Learning & Exploiting Low-Dimensional Structure in High-Dimensional Data

David B. Dunson

Departments of Statistical Science & Mathematics
Duke University

July 29, 2019

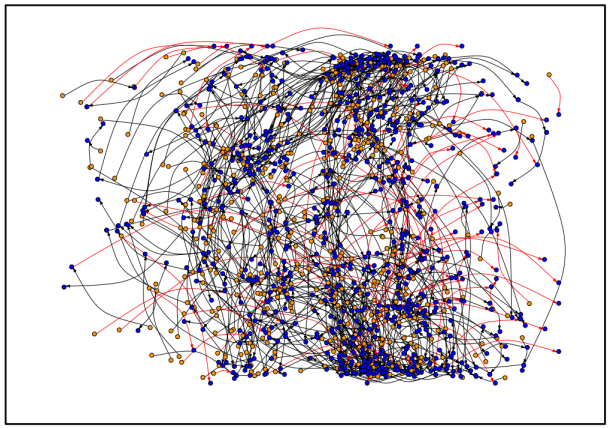
with Didong Li & Minerva Mukhopadhyay

Duke
UNIVERSITY

Overview

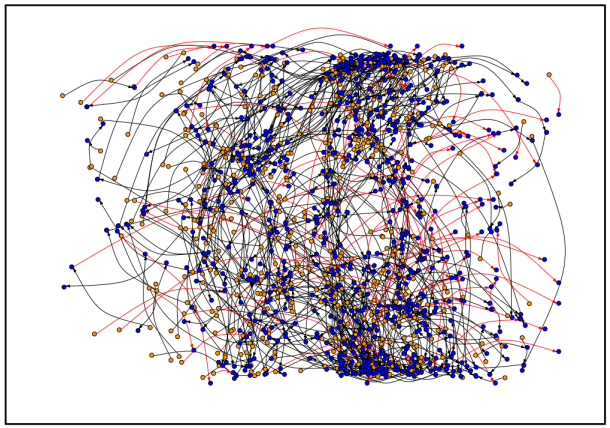
1. Background and Motivation
2. Manifold learning
   - New Dictionary: Spherelets
   - Spherical Principal Component Analysis (SPCA)
   - Algorithm & Examples
3. Density estimation
   - Fisher-Gaussian Kernel
   - Examples
4. Classification
5. Geodesic distance estimation

Motivation



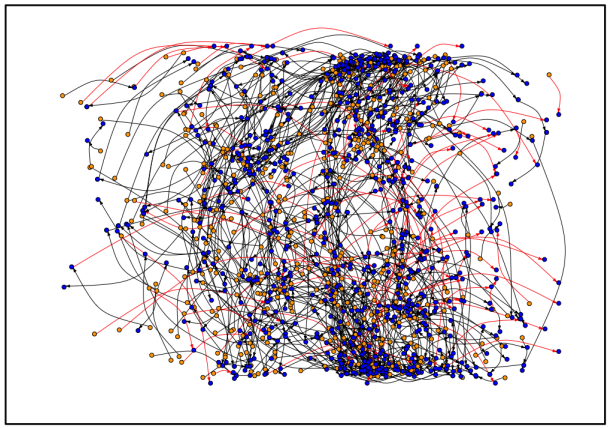
- As an applied statistician, I work with a variety of collaborators - in ecology, neuroscience, epidemiology etc

Motivation



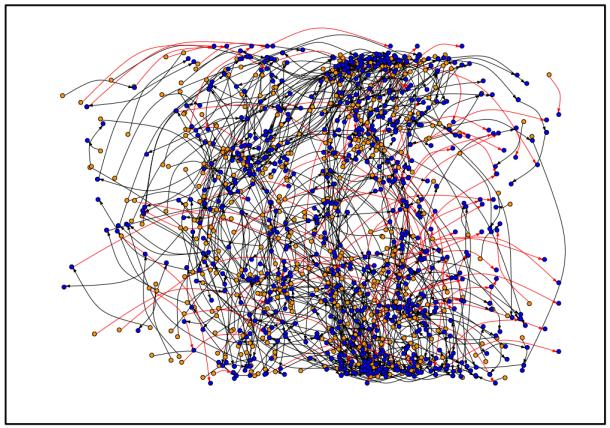
- As an applied statistician, I work with a variety of collaborators - in ecology, neuroscience, epidemiology etc
- It is (of course) typical in modern scientific settings to collect complex high-dimensional data

Motivation



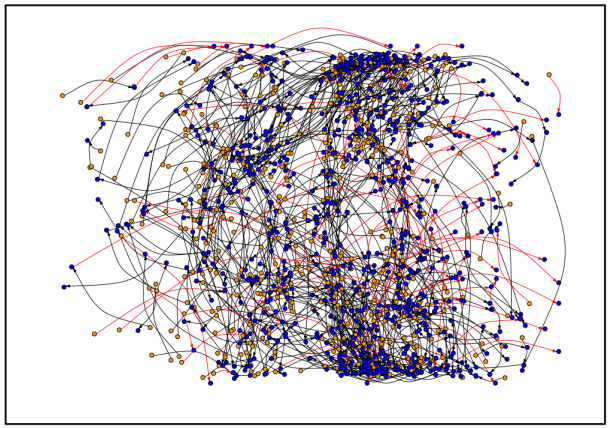
- As an applied statistician, I work with a variety of collaborators - in ecology, neuroscience, epidemiology etc
- It is (of course) typical in modern scientific settings to collect complex high-dimensional data
- Ecology: Occurrences of many different species at distinct spatial locations

Motivation



- As an applied statistician, I work with a variety of collaborators - in ecology, neuroscience, epidemiology etc
- It is (of course) typical in modern scientific settings to collect complex high-dimensional data
- Ecology: Occurrences of many different species at distinct spatial locations
- Neuroscience: Millions of white matter tracts in the human brain

Motivation



- As an applied statistician, I work with a variety of collaborators - in ecology, neuroscience, epidemiology etc
- It is (of course) typical in modern scientific settings to collect complex high-dimensional data
- Ecology: Occurrences of many different species at distinct spatial locations
- Neuroscience: Millions of white matter tracts in the human brain
- Epidemiology: Exposomics - relating many different exposures to health outcomes

Motivation

- Data in all these applications are high-dimensional & complex

Motivation

- Data in all these applications are high-dimensional & complex
- Unlike in many tech applications, often have limited sample size

Motivation

- Data in all these applications are high-dimensional & complex
- Unlike in many tech applications, often have limited sample size
- We don't just want a black box for prediction but want to do science

Motivation

- Data in all these applications are high-dimensional & complex
- Unlike in many tech applications, often have limited sample size
- We don't just want a black box for prediction but want to do science
- Learning of interpretable lower-dimensional structure & relationships in the data

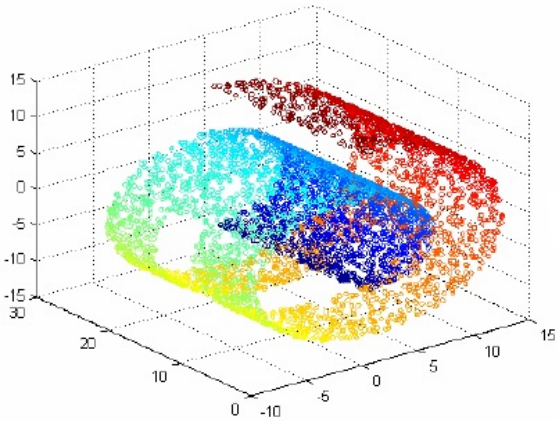
Motivation

- Data in all these applications are high-dimensional & complex
- Unlike in many tech applications, often have limited sample size
- We don't just want a black box for prediction but want to do science
- Learning of interpretable lower-dimensional structure & relationships in the data
- Hypothesis testing, variable selection, statistical inferences

Motivation

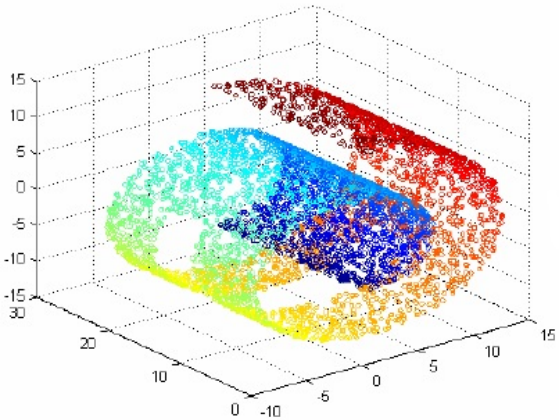
- Data in all these applications are high-dimensional & complex
- Unlike in many tech applications, often have limited sample size
- We don't just want a black box for prediction but want to do science
- Learning of interpretable lower-dimensional structure & relationships in the data
- Hypothesis testing, variable selection, statistical inferences
- Uncertainty quantification is of paramount importance

Dimensionality Reduction



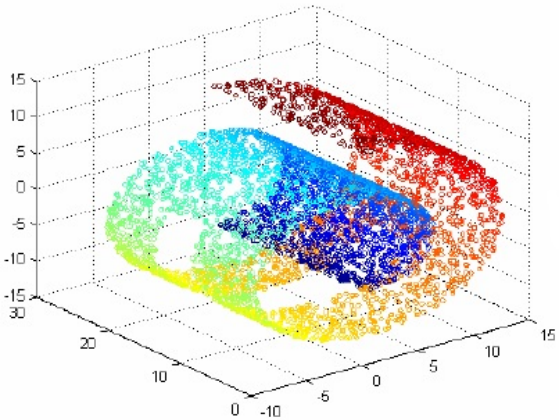
- Certainly to make sense of the data, we need some sort of dimensionality reduction

Dimensionality Reduction



- Certainly to make sense of the data, we need some sort of dimensionality reduction
- Suppose for simplicity (to start) that data $X_i = (X_{i1}, \ldots, X_{ip})' \in \mathbb{R}^p$

Dimensionality Reduction



- Certainly to make sense of the data, we need some sort of dimensionality reduction
- Suppose for simplicity (to start) that data $X_i = (X_{i1}, \ldots, X_{ip})' \in \mathbb{R}^p$
- Then, dimension reduction can be based on the model:

$$X_i = f(\eta_i) + \epsilon_i,$$

where $\eta_i = (\eta_{i1}, \ldots, \eta_{id})'$ are latent factors, $f(.)$ is an unknown function, and ϵ_i is a zero-mean measurement error

Current Strategies for Dimension Reduction

- Supposing $f(.)$ is linear, we can use well established factor analysis & PCA tools

Current Strategies for Dimension Reduction

- Supposing $f(.)$ is linear, we can use well established factor analysis & PCA tools
- But what if the data are intrinsically non-linear?

Current Strategies for Dimension Reduction

- Supposing $f(.)$ is linear, we can use well established factor analysis & PCA tools
- But what if the data are intrinsically non-linear?
- Approach 1: Use a *locally linear* approximation; e.g. cluster data & apply PCA in each neighborhood

Current Strategies for Dimension Reduction

- Supposing $f(.)$ is linear, we can use well established factor analysis & PCA tools
- But what if the data are intrinsically non-linear?
- Approach 1: Use a *locally linear* approximation; e.g. cluster data & apply PCA in each neighborhood
- Approach 2: Use a non-linear factor model, assuming $\eta_i \sim N(0, I)$, with $f(.)$ an unknown non-linear function

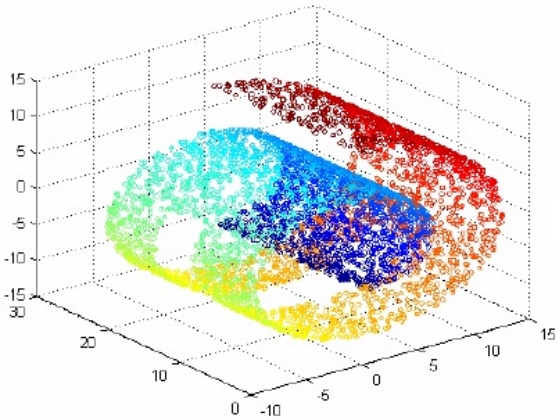
Current Strategies for Dimension Reduction

- Supposing $f(.)$ is linear, we can use well established factor analysis & PCA tools
- But what if the data are intrinsically non-linear?
- Approach 1: Use a *locally linear* approximation; e.g. cluster data & apply PCA in each neighborhood
- Approach 2: Use a non-linear factor model, assuming $\eta_i \sim N(0, I)$, with $f(.)$ an unknown non-linear function
- Approach 2 includes Gaussian process latent variable models (GP-LVMs) & variational auto-encoders, complex black boxes lacking identifiability & interpretability

Current Strategies for Dimension Reduction

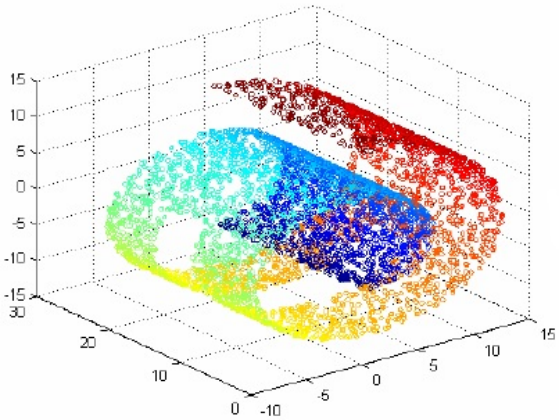
- Supposing $f(.)$ is linear, we can use well established factor analysis & PCA tools
- But what if the data are intrinsically non-linear?
- Approach 1: Use a *locally linear* approximation; e.g. cluster data & apply PCA in each neighborhood
- Approach 2: Use a non-linear factor model, assuming $\eta_i \sim N(0, I)$, with $f(.)$ an unknown non-linear function
- Approach 2 includes Gaussian process latent variable models (GP-LVMs) & variational auto-encoders, complex black boxes lacking identifiability & interpretability
- Our goal: develop simpler approaches including non-linear alternatives to PCA & model-based methods

Manifold learning



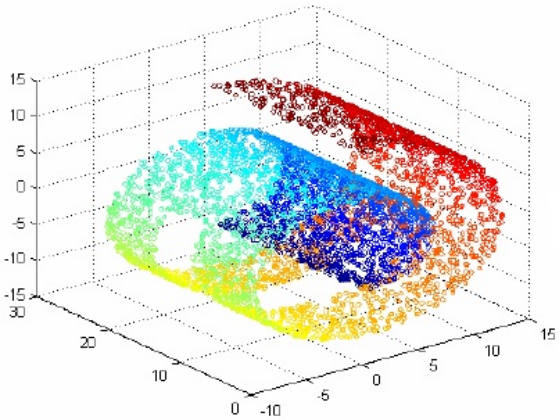
- As a simple toy mathematical model of lower-dimensional non-linear structure, we rely on manifolds

Manifold learning



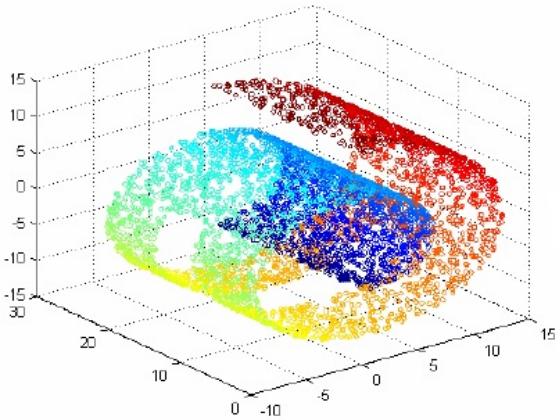
- As a simple toy mathematical model of lower-dimensional non-linear structure, we rely on manifolds
- In particular, in developing methods & studying their properties, useful to suppose data X_i are concentrated near a manifold $\mathcal{M}$

Manifold learning



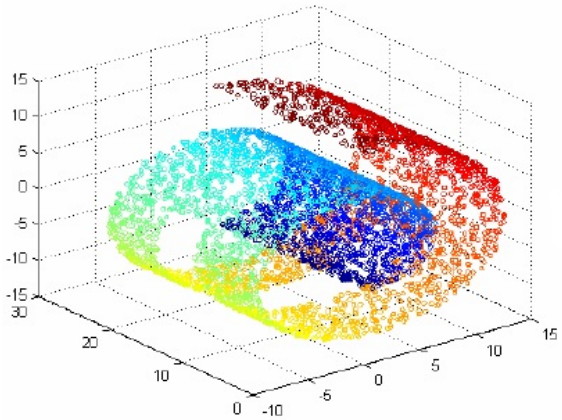
- As a simple toy mathematical model of lower-dimensional non-linear structure, we rely on manifolds
- In particular, in developing methods & studying their properties, useful to suppose data X_i are concentrated near a manifold $\mathcal{M}$
- $\mathcal{M}$ is close to Euclidean in small regions & has dimension $d \ll p$

Manifold learning



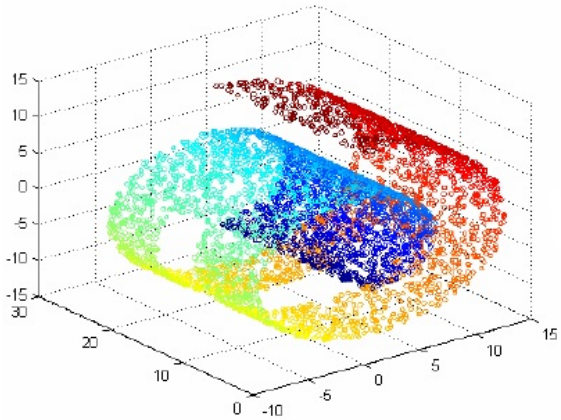
- As a simple toy mathematical model of lower-dimensional non-linear structure, we rely on manifolds
- In particular, in developing methods & studying their properties, useful to suppose data X_i are concentrated near a manifold $\mathcal{M}$
- $\mathcal{M}$ is close to Euclidean in small regions & has dimension $d \ll p$
- An example is the swiss roll (above), which has dimension $d = 2$

Manifold learning



- As a simple toy mathematical model of lower-dimensional non-linear structure, we rely on manifolds
- In particular, in developing methods & studying their properties, useful to suppose data X_i are concentrated near a manifold $\mathcal{M}$
- $\mathcal{M}$ is close to Euclidean in small regions & has dimension $d \ll p$
- An example is the swiss roll (above), which has dimension $d = 2$
- Manifold learning algorithms seek to exploit the manifold structure in mapping $X_i \rightarrow \eta_i$

Manifold learning



- As a simple toy mathematical model of lower-dimensional non-linear structure, we rely on manifolds
- In particular, in developing methods & studying their properties, useful to suppose data X_i are concentrated near a manifold $\mathcal{M}$
- $\mathcal{M}$ is close to Euclidean in small regions & has dimension $d \ll p$
- An example is the swiss roll (above), which has dimension $d = 2$
- Manifold learning algorithms seek to exploit the manifold structure in mapping $X_i \rightarrow \eta_i$
- Misnomer in that manifold estimates are typically not produced - black box for dimensionality reduction

True Manifold Learning

- Algorithms that *actually* “learn” the manifold usually require a *dictionary*

True Manifold Learning

- Algorithms that *actually* "learn" the manifold usually require a *dictionary*
- Dictionaries are made up of simple mathematical pieces used in approximating $\mathcal{M}$

True Manifold Learning

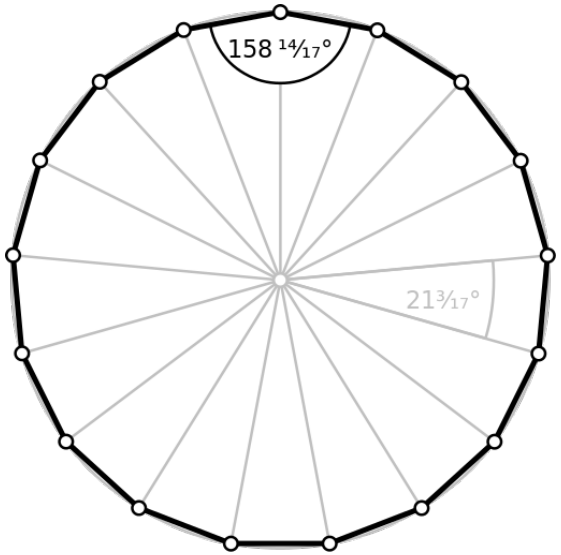
- Algorithms that *actually* "learn" the manifold usually require a *dictionary*
- Dictionaries are made up of simple mathematical pieces used in approximating $\mathcal{M}$
- Almost always these pieces are chosen to be linear

True Manifold Learning

- Algorithms that *actually* "learn" the manifold usually require a *dictionary*
- Dictionaries are made up of simple mathematical pieces used in approximating $\mathcal{M}$
- Almost always these pieces are chosen to be linear
- Using simple linear pieces conceptually nice & appealing computationally

True Manifold Learning

- Algorithms that *actually* "learn" the manifold usually require a *dictionary*
- Dictionaries are made up of simple mathematical pieces used in approximating $\mathcal{M}$
- Almost always these pieces are chosen to be linear
- Using simple linear pieces conceptually nice & appealing computationally
- BUT too many pieces are needed when $\mathcal{M}$ has large curvature



New dictionary - Local quadratic approximations?

- First order $\longrightarrow$ second order: $x^\top Hx + f^\top x + c = 0$.

New dictionary - Local quadratic approximations?

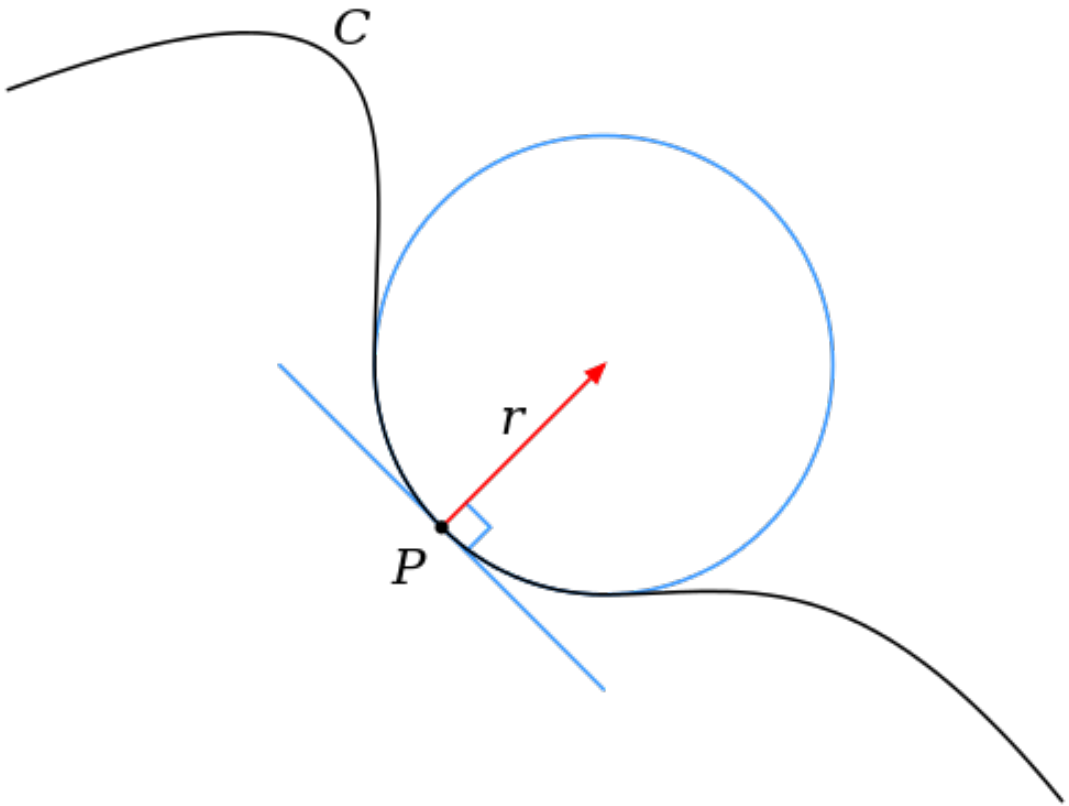
- First order $\longrightarrow$ second order: $x^\top Hx + f^\top x + c = 0$.
- Number of unknown parameters = $\frac{p(p+1)}{2} + p + 1 = O(p^2)$.

New dictionary - Local quadratic approximations?

- First order $\longrightarrow$ second order: $x^\top Hx + f^\top x + c = 0$.
- Number of unknown parameters = $\frac{p(p+1)}{2} + p + 1 = O(p^2)$.
- Trades one problem (*too many pieces*) for another (*too many parameters*)

New dictionary - Local quadratic approximations?

- First order $\longrightarrow$ second order: $x^\top Hx + f^\top x + c = 0$.
- Number of unknown parameters = $\frac{p(p+1)}{2} + p + 1 = O(p^2)$.
- Trades one problem (*too many pieces*) for another (*too many parameters*)
- An alternative is “osculating” circles/spheres



Spherelets - A dictionary for subspaces

- We propose to use pieces of spheres or *spherelets* as a dictionary

Spherelets - A dictionary for subspaces

- We propose to use pieces of spheres or *spherelets* as a dictionary
- Often *many* fewer spheres than planes to obtain the same approximation error

Spherelets - A dictionary for subspaces

- We propose to use pieces of spheres or *spherelets* as a dictionary
- Often *many* fewer spheres than planes to obtain the same approximation error
- Each sphere has few parameters & they are simple geometric objects that are easy to fit

Spherelets - A dictionary for subspaces

- We propose to use pieces of spheres or *spherelets* as a dictionary
- Often *many* fewer spheres than planes to obtain the same approximation error
- Each sphere has few parameters & they are simple geometric objects that are easy to fit
- Before considering algorithms for fitting spherelets, we (briefly) consider their approximation properties

Notation and concepts

- $\mathcal{M}$ is a compact C^3, d-dimensional orientable manifold embedded in $\mathbb{R}^p$

Notation and concepts

- $\mathcal{M}$ is a compact C^3, d-dimensional orientable manifold embedded in $\mathbb{R}^p$
- Trivial to extend our results to a collection of such manifolds

Notation and concepts

- $\mathcal{M}$ is a compact C^3, d-dimensional orientable manifold embedded in $\mathbb{R}^p$
- Trivial to extend our results to a collection of such manifolds
- We want to bound # pieces needed to obtain approximation error ϵ

Notation and concepts

- $\mathcal{M}$ is a compact C^3, d-dimensional orientable manifold embedded in $\mathbb{R}^p$
- Trivial to extend our results to a collection of such manifolds
- We want to bound # pieces needed to obtain approximation error ϵ
- $N_H(\epsilon, \mathcal{M})$ = minimal # hyperplanes, $N_S(\epsilon, \mathcal{M})$ = minimal # spheres

Notation and concepts

- $\mathcal{M}$ is a compact C^3, d-dimensional orientable manifold embedded in $\mathbb{R}^p$
- Trivial to extend our results to a collection of such manifolds
- We want to bound # pieces needed to obtain approximation error ϵ
- $N_H(\epsilon, \mathcal{M})$ = minimal # hyperplanes, $N_S(\epsilon, \mathcal{M})$ = minimal # spheres
- K=max curvature, T=maximum rate of change in curvature, V = Vol($\mathcal{M}$).

Main Theorem

Theorem (Li, Mukhopadhyay, D., 18')

1. *The bound on the hyperplane covering number is*

$$N_H(\epsilon, \mathcal{M}) \leq V\left(\frac{2\epsilon}{K}\right)^{-\frac{d}{2}}$$

Main Theorem

Theorem (Li, Mukhopadhyay, D., 18')

1. *The bound on the hyperplane covering number is*

$$N_H(\epsilon, \mathcal{M}) \leq V\left(\frac{2\epsilon}{K}\right)^{-\frac{d}{2}}$$

2. *Let* $F_\epsilon := \{p \in \mathcal{M} : |k_1(p) - k_d(p)| \leq (\frac{2\epsilon}{K})^{\frac{1}{2}}\}$, *where* $k_1(p)$ *and* $k_d(p)$ *are the max & min principal curvature of* $\mathcal{M}$ *at* p.

Main Theorem

Theorem (Li, Mukhopadhyay, D., 18')

1. *The bound on the hyperplane covering number is*

$$N_H(\epsilon, \mathcal{M}) \leq V\left(\frac{2\epsilon}{K}\right)^{-\frac{d}{2}}$$

2. *Let* $F_\epsilon := \{p \in \mathcal{M} : |k_1(p) - k_d(p)| \leq (\frac{2\epsilon}{K})^{\frac{1}{2}}\}$, *where* $k_1(p)$ *and* $k_d(p)$ *are the max & min principal curvature of* $\mathcal{M}$ *at p. Let*

$$\mathcal{M}_\epsilon := \bigcup_{p \in F_\epsilon} B\left(p, \left(\frac{6\epsilon}{3+T}\right)^{\frac{1}{3}}\right) \text{ and } V_\epsilon := \text{Vol}(\mathcal{M}_\epsilon),$$

Main Theorem

Theorem (Li, Mukhopadhyay, D., 18')

1. *The bound on the hyperplane covering number is*

$$N_H(\epsilon, \mathcal{M}) \leq V\left(\frac{2\epsilon}{K}\right)^{-\frac{d}{2}}$$

2. *Let* $F_\epsilon := \{p \in \mathcal{M} : |k_1(p) - k_d(p)| \leq (\frac{2\epsilon}{K})^{\frac{1}{2}}\}$, *where* $k_1(p)$ *and* $k_d(p)$ *are the max & min principal curvature of* $\mathcal{M}$ *at p. Let* $\mathcal{M}_\epsilon := \bigcup_{p \in F_\epsilon} B\left(p, \left(\frac{6\epsilon}{3+T}\right)^{\frac{1}{3}}\right)$ *and* $V_\epsilon := \text{Vol}(\mathcal{M}_\epsilon)$, *then*

$$N_S(\epsilon, \mathcal{M}) \leq V_\epsilon\left(\frac{6\epsilon}{3+T}\right)^{-\frac{d}{3}} + (V - V_\epsilon)\left(\frac{2\epsilon}{K}\right)^{-\frac{d}{2}}$$

Implications of the Theorem

- Since $\epsilon \approx 0$, $\epsilon^{-d/2}$ is very large showing the *curse of dimensionality*

Implications of the Theorem

- Since $\epsilon \approx 0$, $\epsilon^{-d/2}$ is very large showing the *curse of dimensionality*
- Even if an oracle could perfectly choose the pieces to best approximate $\mathcal{M}$, we need lots of pieces as d increases for small ϵ

Implications of the Theorem

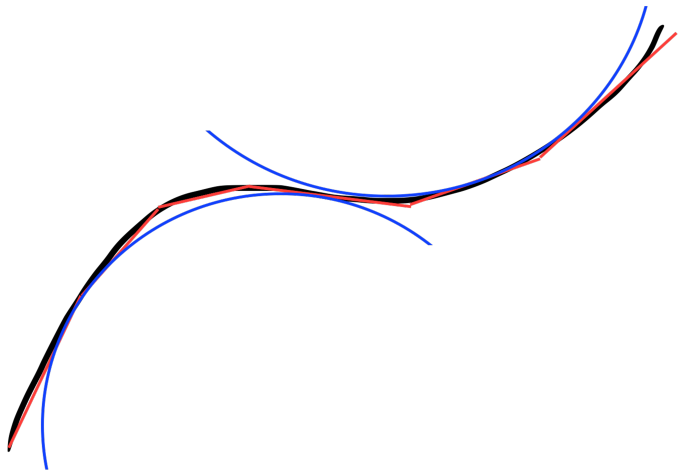
- Since $\epsilon \approx 0$, $\epsilon^{-d/2}$ is very large showing the *curse of dimensionality*
- Even if an oracle could perfectly choose the pieces to best approximate $\mathcal{M}$, we need lots of pieces as d increases for small ϵ
- Spherelets can decrease the impact of the curse to $\epsilon^{-d/3}$ IF

Implications of the Theorem

- Since $\epsilon \approx 0$, $\epsilon^{-d/2}$ is very large showing the *curse of dimensionality*
- Even if an oracle could perfectly choose the pieces to best approximate $\mathcal{M}$, we need lots of pieces as d increases for small ϵ
- Spherelets can decrease the impact of the curse to $\epsilon^{-d/3}$ IF
- There aren't too many locations $p \in \mathcal{M}$ having big changes in principal curvature

Implications of the Theorem

- Since $\epsilon \approx 0$, $\epsilon^{-d/2}$ is very large showing the *curse of dimensionality*
- Even if an oracle could perfectly choose the pieces to best approximate $\mathcal{M}$, we need lots of pieces as d increases for small ϵ
- Spherelets can decrease the impact of the curse to $\epsilon^{-d/3}$ IF
- There aren't too many locations $p \in \mathcal{M}$ having big changes in principal curvature



Spherical principal component analysis (SPCA)

Definition

$X \in \mathbb{R}^{n \times p}$,

Spherical principal component analysis (SPCA)

Definition

$X \in \mathbb{R}^{n \times p}$, $d \ll p$,

Spherical principal component analysis (SPCA)

Definition

$X \in \mathbb{R}^{n \times p}$, $d \ll p$, $Y_i = \bar{X} + \widehat{V}\widehat{V}^\top (X_i - \bar{X})$,

Spherical principal component analysis (SPCA)

Definition

$X \in \mathbb{R}^{n \times p}$, $d \ll p$, $Y_i = \bar{X} + \widehat{V}\widehat{V}^\top (X_i - \bar{X})$, $\widehat{V} = (v_1, \cdots, v_{d+1})$, $v_i = \mathrm{evec}_i\{(X - 1\bar{X}^\top)^\top (X - 1\bar{X}^\top)\}$, where $\mathrm{evec}_i(S)$ is the ith eigenvector of S in decreasing order.

Spherical principal component analysis (SPCA)

Definition

$X \in \mathbb{R}^{n \times p}$, $d \ll p$, $Y_i = \bar{X} + \widehat{V}\widehat{V}^\top (X_i - \bar{X})$, $\widehat{V} = (v_1, \cdots, v_{d+1})$, $v_i = \mathrm{evec}_i\{(X - 1\bar{X}^\top)^\top (X - 1\bar{X}^\top)\}$, where $\mathrm{evec}_i(S)$ is the ith eigenvector of S in decreasing order. $Z_i = \widehat{c} + \frac{\hat{r}}{\|Y_i - \widehat{c}\|}(Y_i - \widehat{c})$ is the d-dimensional spherical component of X,

Spherical principal component analysis (SPCA)

Definition

$X \in \mathbb{R}^{n \times p}$, $d \ll p$, $Y_i = \bar{X} + \widehat{V}\widehat{V}^\top (X_i - \bar{X})$, $\widehat{V} = (v_1, \cdots, v_{d+1})$, $v_i = \mathrm{evec}_i\{(X - 1\bar{X}^\top)^\top (X - 1\bar{X}^\top)\}$, where $\mathrm{evec}_i(S)$ is the ith eigenvector of S in decreasing order. $Z_i = \widehat{c} + \frac{\hat{r}}{\|Y_i - \widehat{c}\|}(Y_i - \widehat{c})$ is the d-dimensional spherical component of X, where $\hat{r} = \frac{1}{n}\sum_{i=1}^n \|Y_i - \widehat{c}\|$,

$$\widehat{c} = -\frac{1}{2}\left(\sum_{i=1}^n (\bar{Y} - Y_i)(\bar{Y} - Y_i)^\top\right)^{-1} \sum_{i=1}^n \left(\|Y_i^\top Y_i\| - \frac{1}{n}\sum_{j=1}^n \|Y_j^\top Y_j\|\right)(\bar{Y} - Y_i).$$

Spherical principal component analysis (SPCA)

Definition

$X \in \mathbb{R}^{n \times p}$, $d \ll p$, $Y_i = \bar{X} + \widehat{V}\widehat{V}^\top (X_i - \bar{X})$, $\widehat{V} = (v_1, \cdots, v_{d+1})$, $v_i = \mathrm{evec}_i\{(X - 1\bar{X}^\top)^\top (X - 1\bar{X}^\top)\}$, where $\mathrm{evec}_i(S)$ is the ith eigenvector of S in decreasing order. $Z_i = \widehat{c} + \frac{\hat{r}}{\|Y_i - \widehat{c}\|}(Y_i - \widehat{c})$ is the d-dimensional spherical component of X, where $\hat{r} = \frac{1}{n}\sum_{i=1}^n \|Y_i - \widehat{c}\|$,

$$\widehat{c} = -\frac{1}{2}\left(\sum_{i=1}^n (\bar{Y} - Y_i)(\bar{Y} - Y_i)^\top\right)^{-1} \sum_{i=1}^n \left(\|Y_i^\top Y_i\| - \frac{1}{n}\sum_{j=1}^n \|Y_j^\top Y_j\|\right)(\bar{Y} - Y_i).$$

- d-PSPCA = the projection of X to the "best" d dimensional sphere centered at c with radius r

Spherical principal component analysis (SPCA)

Definition

$X \in \mathbb{R}^{n \times p}$, $d \ll p$, $Y_i = \bar{X} + \widehat{V}\widehat{V}^\top (X_i - \bar{X})$, $\widehat{V} = (v_1, \cdots, v_{d+1})$, $v_i = \mathrm{evec}_i\{(X - 1\bar{X}^\top)^\top (X - 1\bar{X}^\top)\}$, where $\mathrm{evec}_i(S)$ is the ith eigenvector of S in decreasing order. $Z_i = \widehat{c} + \frac{\hat{r}}{\|Y_i - \widehat{c}\|}(Y_i - \widehat{c})$ is the d-dimensional spherical component of X, where $\hat{r} = \frac{1}{n}\sum_{i=1}^n \|Y_i - \widehat{c}\|$,

$$\widehat{c} = -\frac{1}{2}\left(\sum_{i=1}^n (\bar{Y} - Y_i)(\bar{Y} - Y_i)^\top\right)^{-1} \sum_{i=1}^n \left(\|Y_i^\top Y_i\| - \frac{1}{n}\sum_{j=1}^n \|Y_j^\top Y_j\|\right)(\bar{Y} - Y_i).$$

- d-PSPCA = the projection of X to the "best" d dimensional sphere centered at c with radius r
- Simple analytic form & can be trivially used in place of PCA in broad settings

Loss function

- SPCA minimizes the loss function

$$\sum_{i=1}^{n}(X_i^\top X_i + f^\top X_i + b)^2$$

where $\hat{f} = -2\hat{c}$ and $\hat{b} = \|\hat{c}\|^2 - \hat{r}^2$.

Loss function

- SPCA minimizes the loss function

$$\sum_{i=1}^{n}(X_i^\top X_i + f^\top X_i + b)^2$$

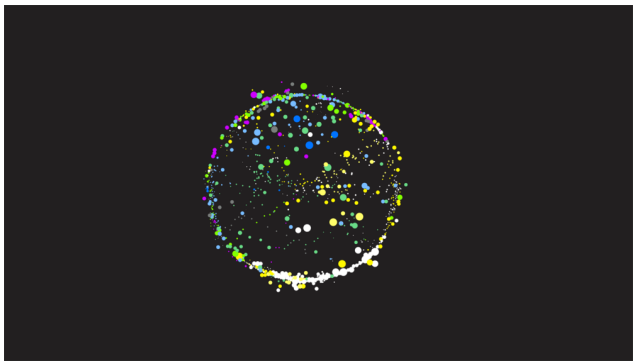
where $\hat{f} = -2\hat{c}$ and $\hat{b} = \|\hat{c}\|^2 - \hat{r}^2$.

- PCA minimizes the loss function

$$\sum_{i=1}^{n}(f^\top X_i + b)^2,$$

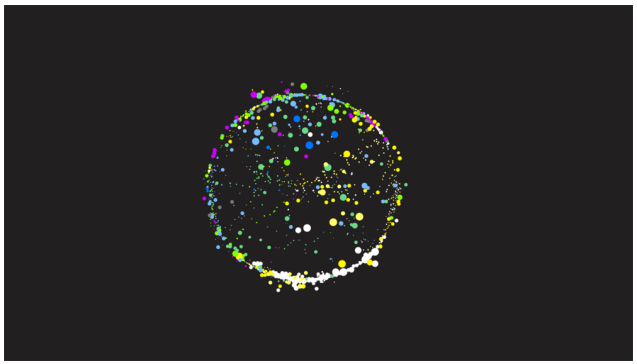
where $\hat{f}$ is the unit normal vector of the best d-dimensional affine subspace, or the eigenvector of covariance matrix corresponding to the smallest eigenvalue.

Spherical projection



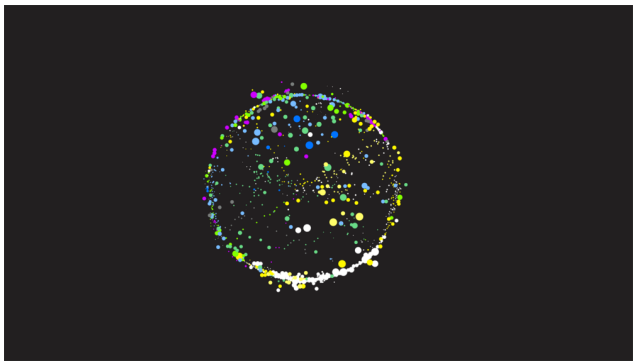
- $\widehat{\mathrm{Proj}}_n(x) := \hat{c} + \frac{\hat{r}}{\|\widehat{V}\widehat{V}^\top(x-\hat{c})\|}\widehat{V}\widehat{V}^\top(x-\hat{c})$ is the spherical projection to $S_{\widehat{V}}(\hat{c},\hat{r})$

Spherical projection



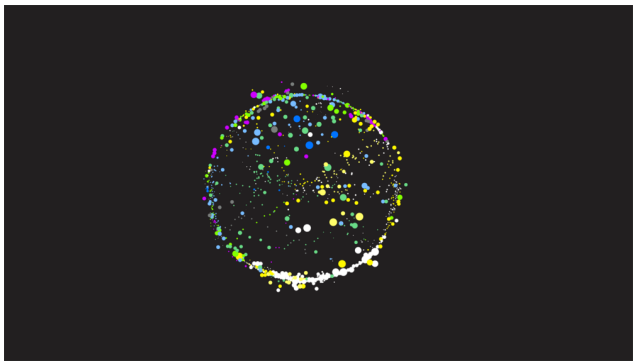
- $\widehat{\mathrm{Proj}}_n(x) := \hat{c} + \frac{\hat{r}}{\|\widehat{V}\widehat{V}^\top(x-\hat{c})\|}\widehat{V}\widehat{V}^\top(x-\hat{c})$ is the spherical projection to $S_{\widehat{V}}(\hat{c},\hat{r})$
- $\mathrm{Proj}^*(x) := c^* + \frac{r^*}{\|V^*V^{*\top}(x-c^*)\|}V^*V^{*\top}(x-c^*)$ is the population version

Spherical projection



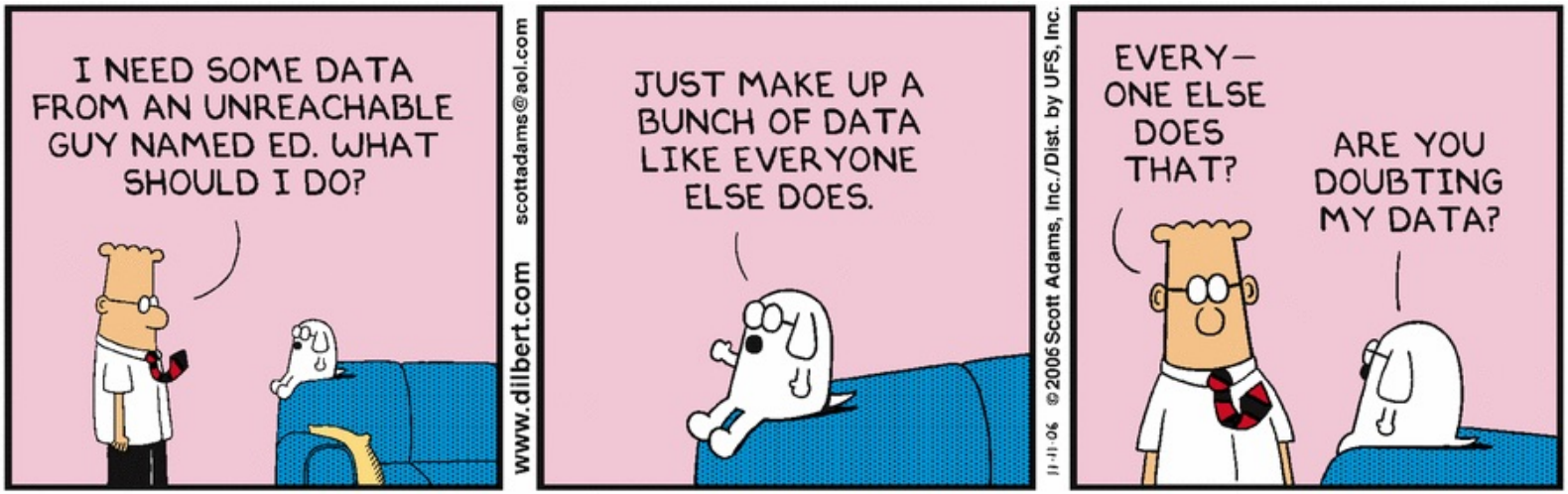
- $\widehat{\mathrm{Proj}}_n(x) := \hat{c} + \frac{\hat{r}}{\|\widehat{V}\widehat{V}^\top(x-\hat{c})\|}\widehat{V}\widehat{V}^\top(x-\hat{c})$ is the spherical projection to $S_{\widehat{V}}(\hat{c},\hat{r})$
- $\mathrm{Proj}^*(x) := c^* + \frac{r^*}{\|V^*V^{*\top}(x-c^*)\|}V^*V^{*\top}(x-c^*)$ is the population version
- $\widehat{\mathrm{Proj}}_n$ converges to Proj^* in probability under some mild conditions

Spherical projection



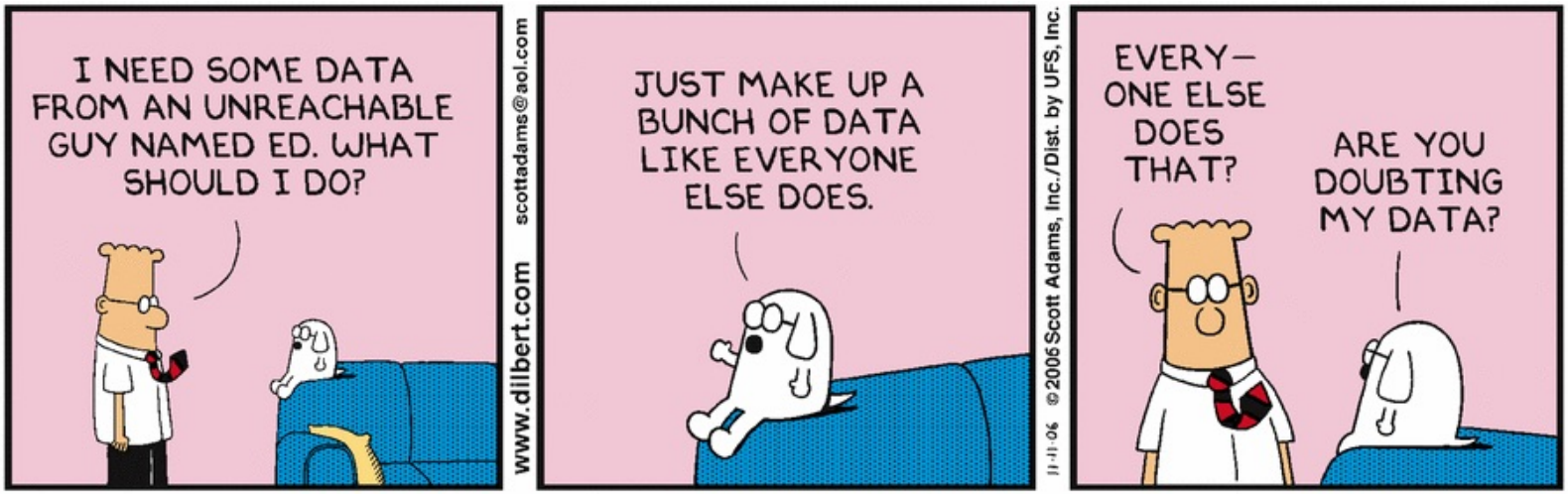
- $\widehat{\mathrm{Proj}}_n(x) := \hat{c} + \frac{\hat{r}}{\|\widehat{V}\widehat{V}^\top(x-\hat{c})\|}\widehat{V}\widehat{V}^\top(x-\hat{c})$ is the spherical projection to $S_{\widehat{V}}(\hat{c},\hat{r})$
- $\mathrm{Proj}^*(x) := c^* + \frac{r^*}{\|V^*V^{*\top}(x-c^*)\|}V^*V^{*\top}(x-c^*)$ is the population version
- $\widehat{\mathrm{Proj}}_n$ converges to Proj^* in probability under some mild conditions
- We also provide concentration inequalities etc in our paper

Analyzing data using spherelets



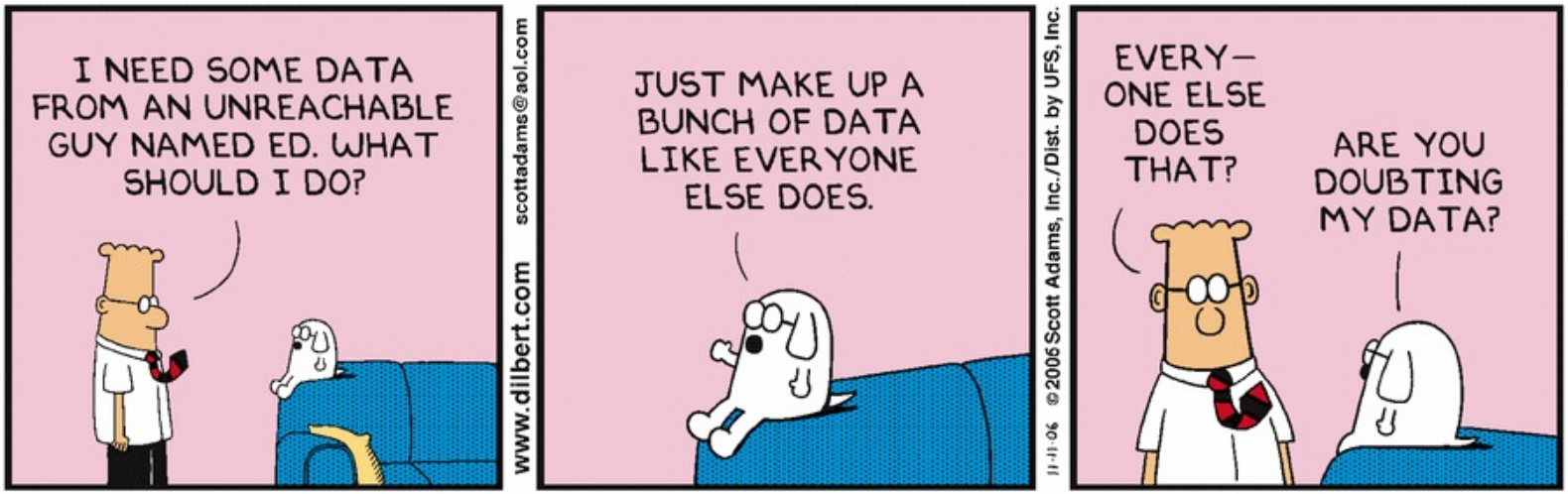
- The main theorem suggests that we should see big gains in practical performance

Analyzing data using spherelets



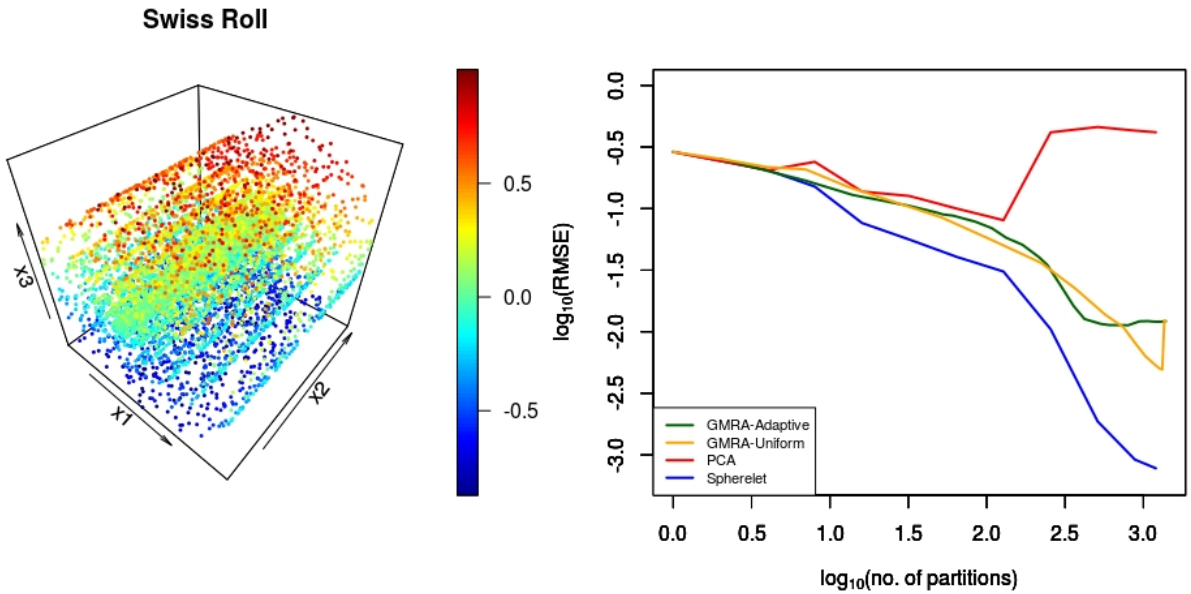
- The main theorem suggests that we should see big gains in practical performance
- For any (locally) linear algorithm, we can replace PCA by spherical PCA & get the spherical version

Analyzing data using spherelets

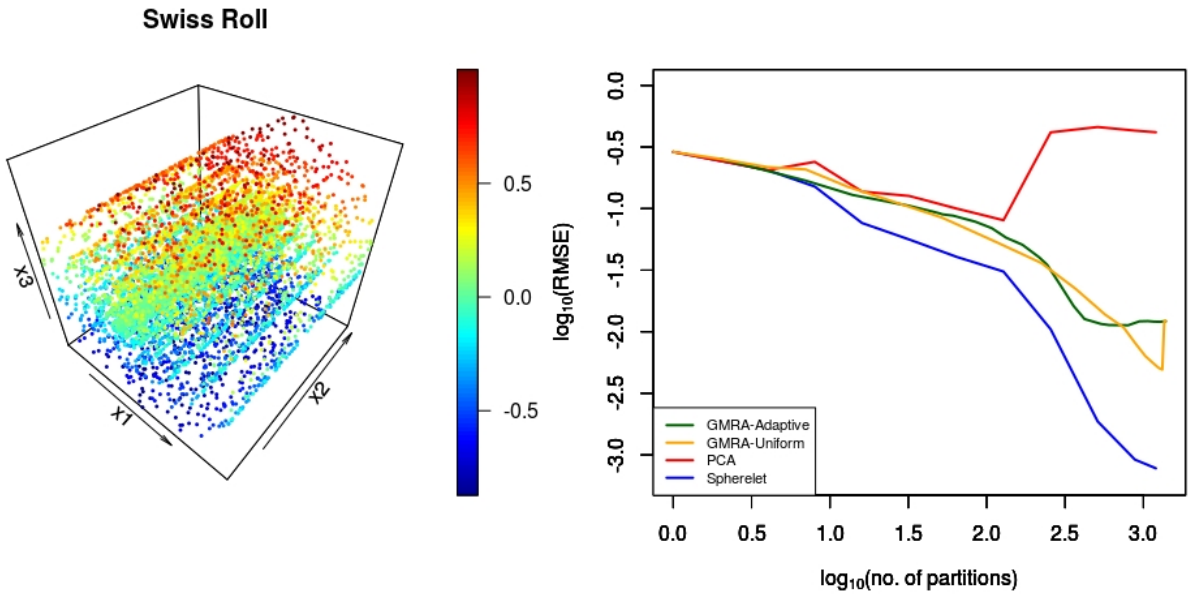


- The main theorem suggests that we should see big gains in practical performance
- For any (locally) linear algorithm, we can replace PCA by spherical PCA & get the spherical version
- Initially develop simple local SPCA algorithm & compared to local PCA

Analyzing data using spherelets

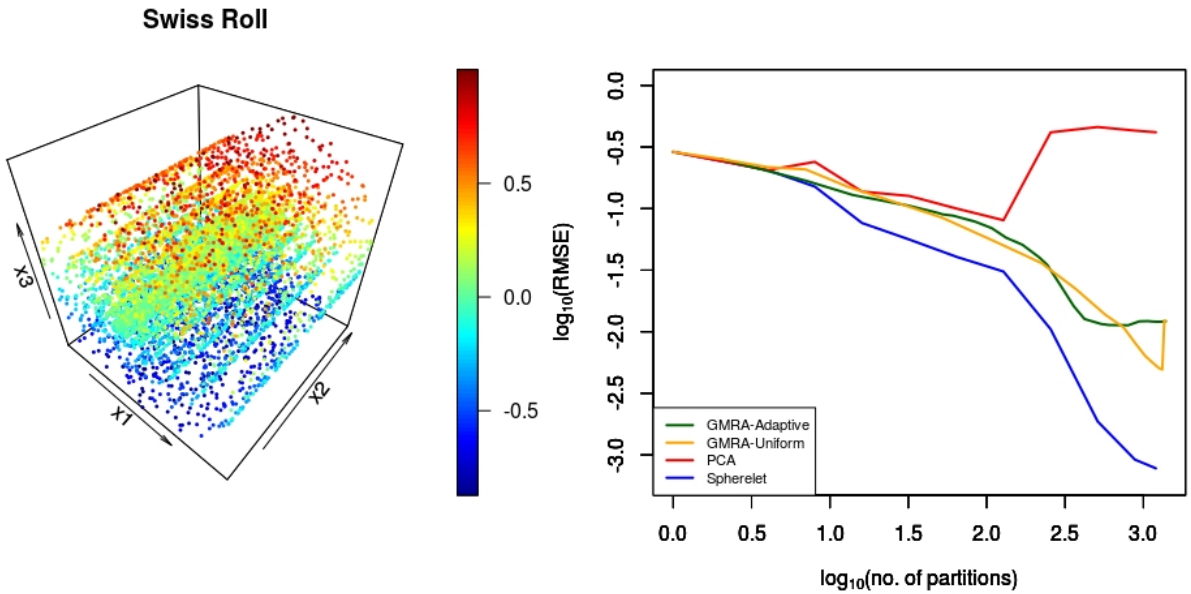


Analyzing data using spherelets



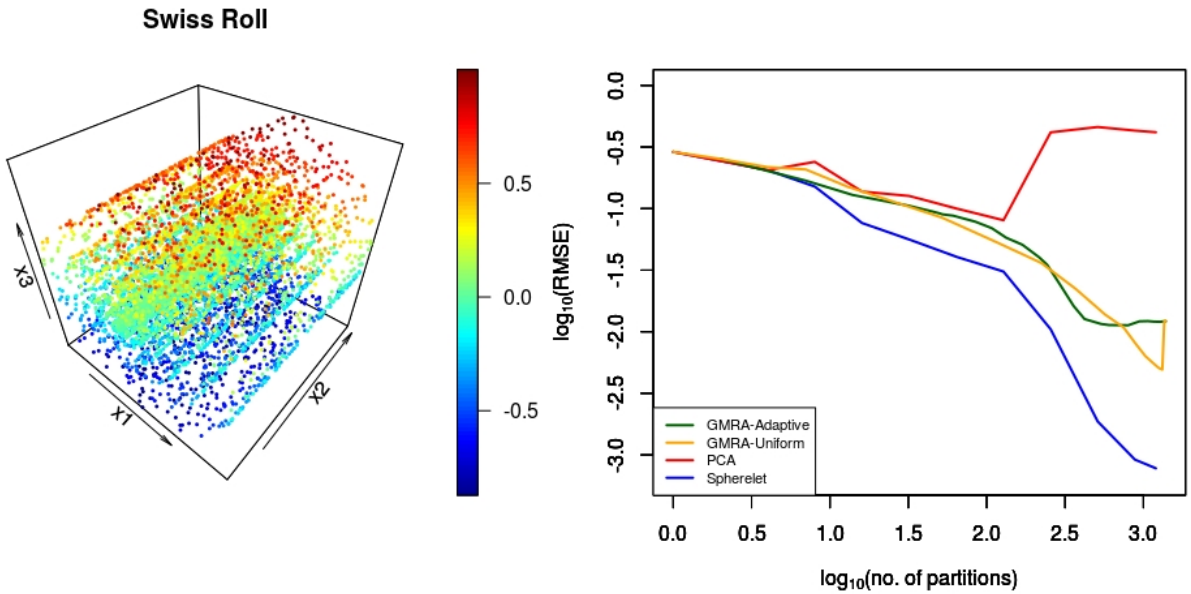
- Analyzed many different toy & real data sets using local SPCA

Analyzing data using spherelets



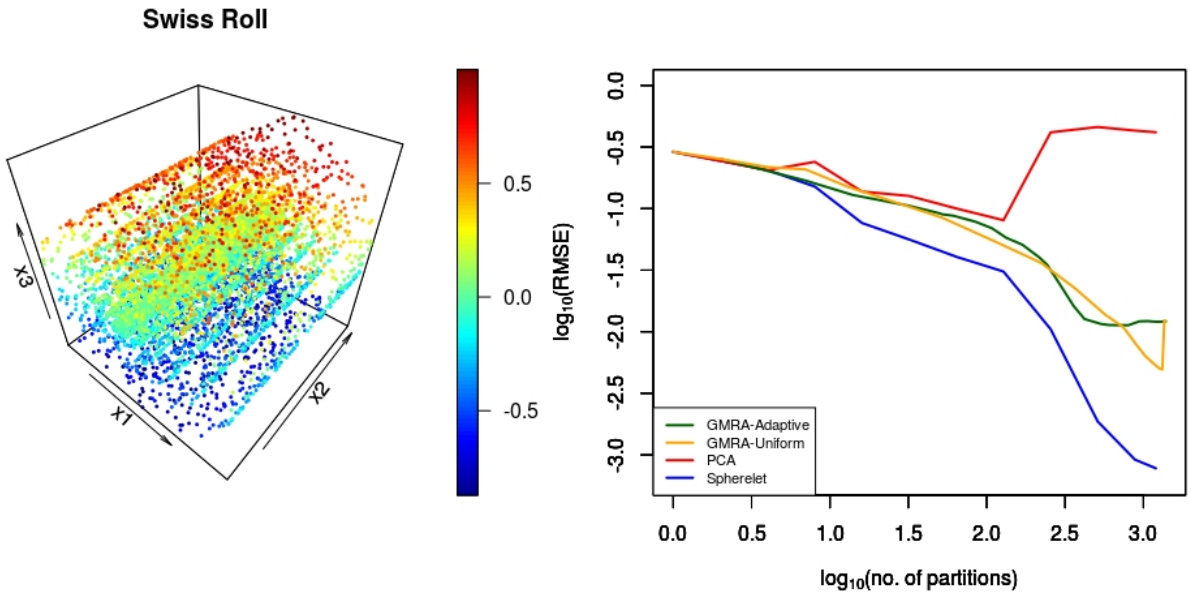
- Analyzed many different toy & real data sets using local SPCA
- Compared with local PCA & GMRA

Analyzing data using spherelets



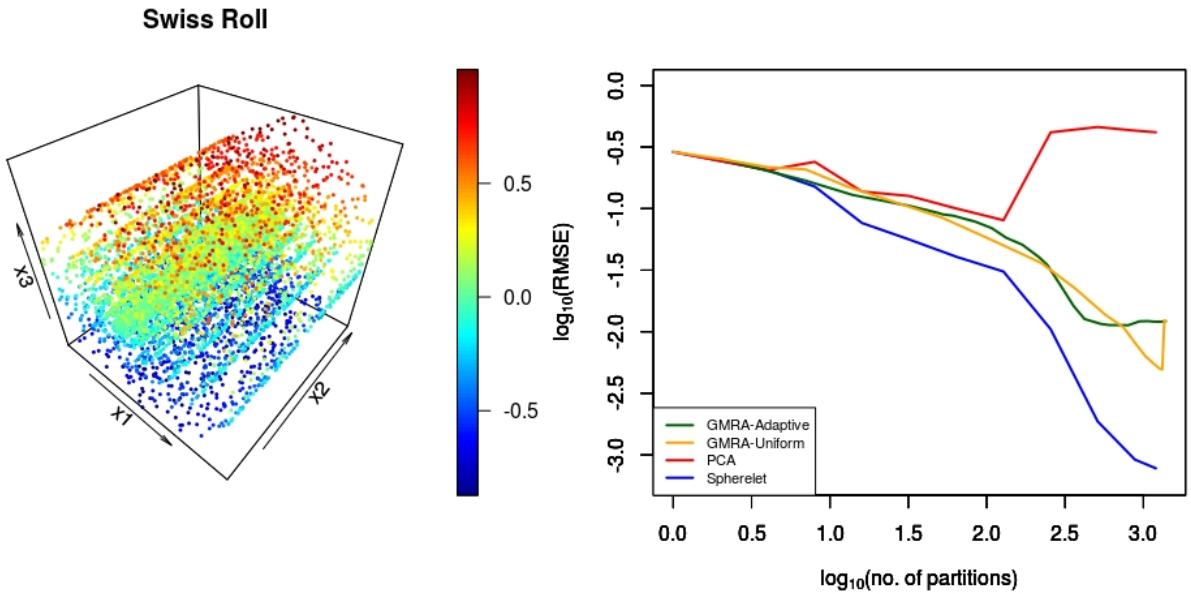
- Analyzed many different toy & real data sets using local SPCA
- Compared with local PCA & GMRA
- Datasets randomly split into training & test (1/2 each)

Analyzing data using spherelets



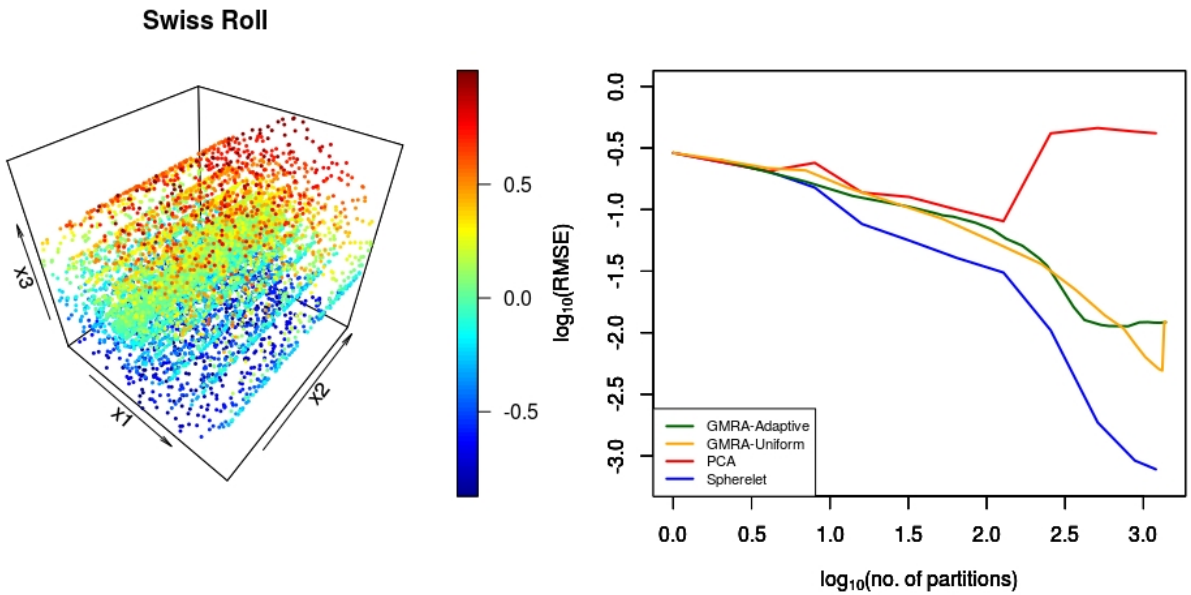
- Analyzed many different toy & real data sets using local SPCA
- Compared with local PCA & GMRA
- Datasets randomly split into training & test (1/2 each)
- Measured RMSE on test set vs # pieces

Analyzing data using spherelets



- Analyzed many different toy & real data sets using local SPCA
- Compared with local PCA & GMRA
- Datasets randomly split into training & test (1/2 each)
- Measured RMSE on test set vs # pieces
- Result: spherelets produces dramatically lower RMSE curves

Analyzing data using spherelets



- Analyzed many different toy & real data sets using local SPCA
- Compared with local PCA & GMRA
- Datasets randomly split into training & test (1/2 each)
- Measured RMSE on test set vs # pieces
- Result: spherelets produces dramatically lower RMSE curves
- Spirals, olympic rings, swiss roll, armadillo, dragon & real datasets (Iris, EUStockMarkets, Seals, banknotes, quakes)

Denoising

- Manifold Blurring Mean Shift (MBMS) is designed for denoising data which are assumed to lie in some lower dimensional manifold.

Denoising

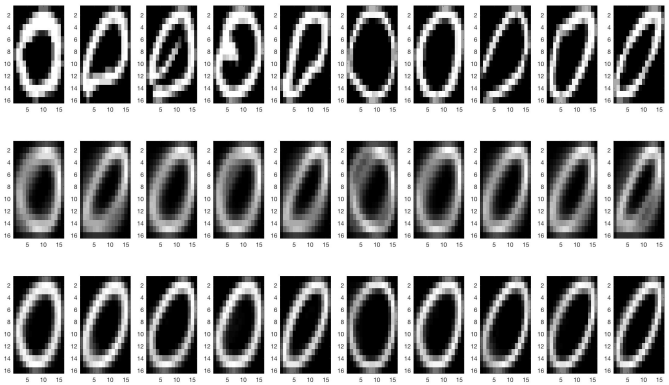
- Manifold Blurring Mean Shift (MBMS) is designed for denoising data which are assumed to lie in some lower dimensional manifold.
- MBMS first shifts the original data toward their local mean, then the shifted data are projected to the tangent space by standard PCA.

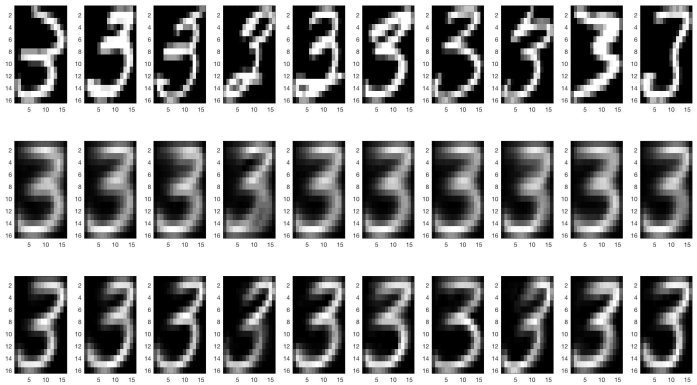
Denoising

- Manifold Blurring Mean Shift (MBMS) is designed for denoising data which are assumed to lie in some lower dimensional manifold.
- MBMS first shifts the original data toward their local mean, then the shifted data are projected to the tangent space by standard PCA.
- We modify MBMS by applying spherelets to provide a locally spherical projection instead of linear by replacing local PCA by local SPCA.

Denoising

Application on USPS Hand written digits images ($D = 256$).





The first row: original data;
The second row: denoised data by MBMS;
The third row: denoised data by SMBMS.

Spherelets - What have we done & where to next?

- Spherelets/SPCA provides a useful alternative to PCA that incorporates curvature

Spherelets - What have we done & where to next?

- Spherelets/SPCA provides a useful alternative to PCA that incorporates curvature
- Has a simple analytic form & can be used in place of linear PCA in broad settings

Spherelets - What have we done & where to next?

- Spherelets/SPCA provides a useful alternative to PCA that incorporates curvature
- Has a simple analytic form & can be used in place of linear PCA in broad settings
- Other non-linear generalizations of PCA tend to be complicated & not as interpretable

Spherelets - What have we done & where to next?

- Spherelets/SPCA provides a useful alternative to PCA that incorporates curvature
- Has a simple analytic form & can be used in place of linear PCA in broad settings
- Other non-linear generalizations of PCA tend to be complicated & not as interpretable
- Motivated by the success of SPCA, interesting to develop model-based approaches

Spherelets - What have we done & where to next?

- Spherelets/SPCA provides a useful alternative to PCA that incorporates curvature
- Has a simple analytic form & can be used in place of linear PCA in broad settings
- Other non-linear generalizations of PCA tend to be complicated & not as interpretable
- Motivated by the success of SPCA, interesting to develop model-based approaches
- How to model the distribution of multivariate data concentrated close to some non-linear support??

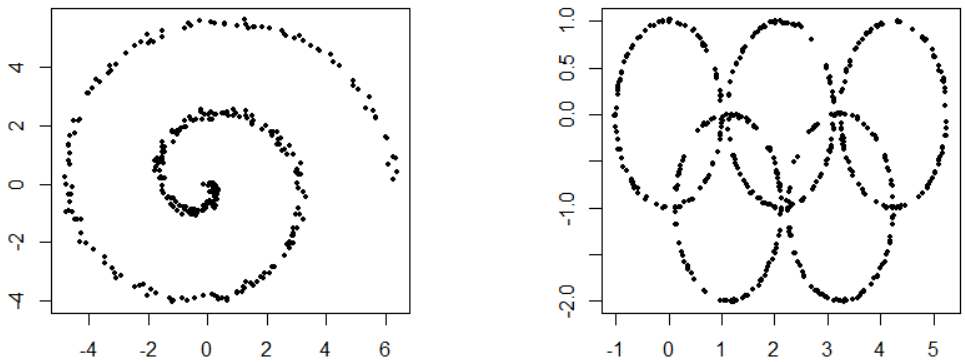
Spherelets - What have we done & where to next?

- Spherelets/SPCA provides a useful alternative to PCA that incorporates curvature
- Has a simple analytic form & can be used in place of linear PCA in broad settings
- Other non-linear generalizations of PCA tend to be complicated & not as interpretable
- Motivated by the success of SPCA, interesting to develop model-based approaches
- How to model the distribution of multivariate data concentrated close to some non-linear support??
- In particular, without resorting to complex/non-identifiable black-boxes (GP-LVMs, VAEs)?

Spherelets - What have we done & where to next?

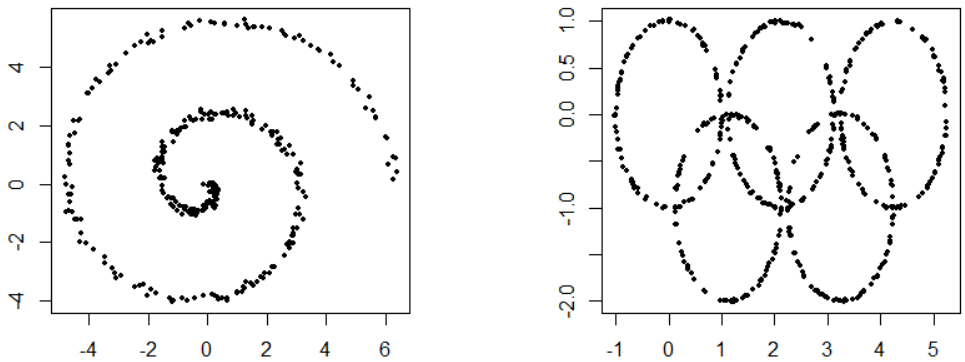
- Spherelets/SPCA provides a useful alternative to PCA that incorporates curvature
- Has a simple analytic form & can be used in place of linear PCA in broad settings
- Other non-linear generalizations of PCA tend to be complicated & not as interpretable
- Motivated by the success of SPCA, interesting to develop model-based approaches
- How to model the distribution of multivariate data concentrated close to some non-linear support??
- In particular, without resorting to complex/non-identifiable black-boxes (GP-LVMs, VAEs)?
- Also, we want to incorporate uncertainty quantification & allow easy extensions to more complex settings

Density estimation with non-linear support



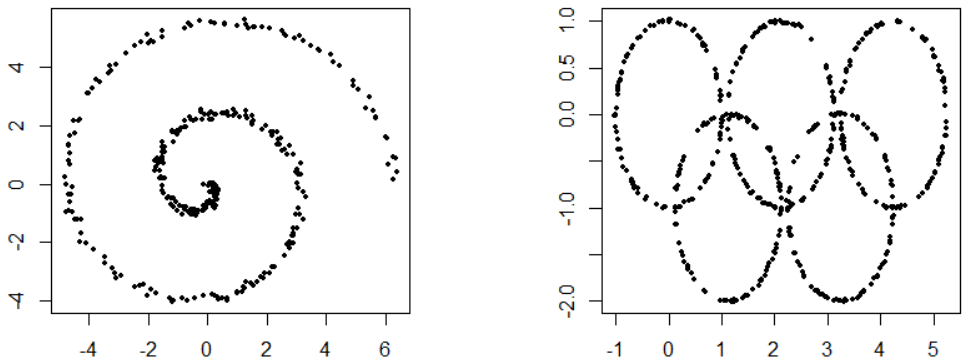
- Consider the above plots - data are concentrated near a non-linear support

Density estimation with non-linear support



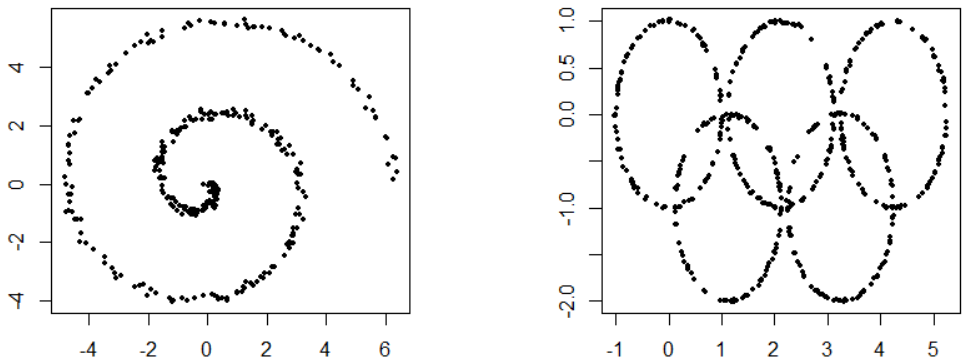
- Consider the above plots - data are concentrated near a non-linear support
- Standard toy examples in the manifold learning literature – serve as useful motivation

Density estimation with non-linear support



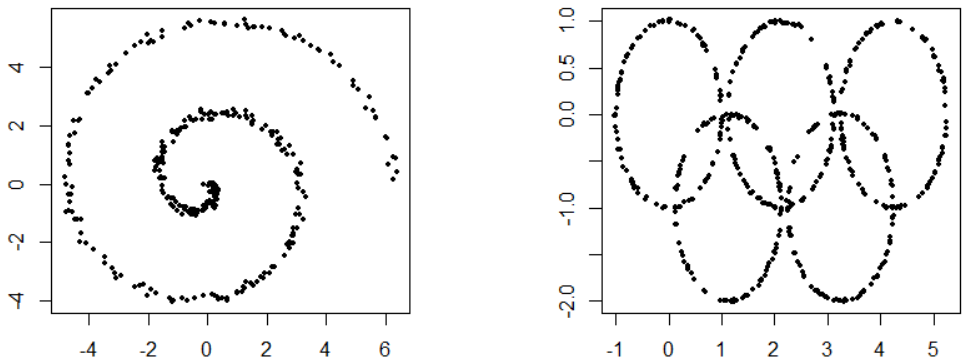
- Consider the above plots - data are concentrated near a non-linear support
- Standard toy examples in the manifold learning literature – serve as useful motivation
- What if we had non-linear structure like this in our data?

Density estimation with non-linear support



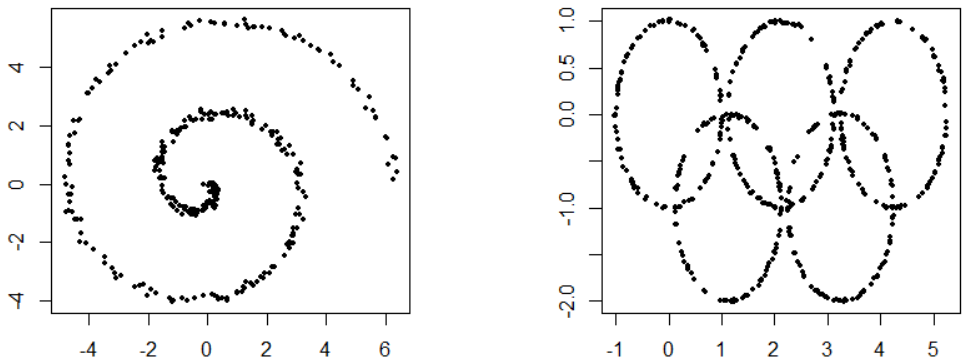
- Consider the above plots - data are concentrated near a non-linear support
- Standard toy examples in the manifold learning literature – serve as useful motivation
- What if we had non-linear structure like this in our data?
- How to model or estimate the density of the data?

Density estimation with non-linear support



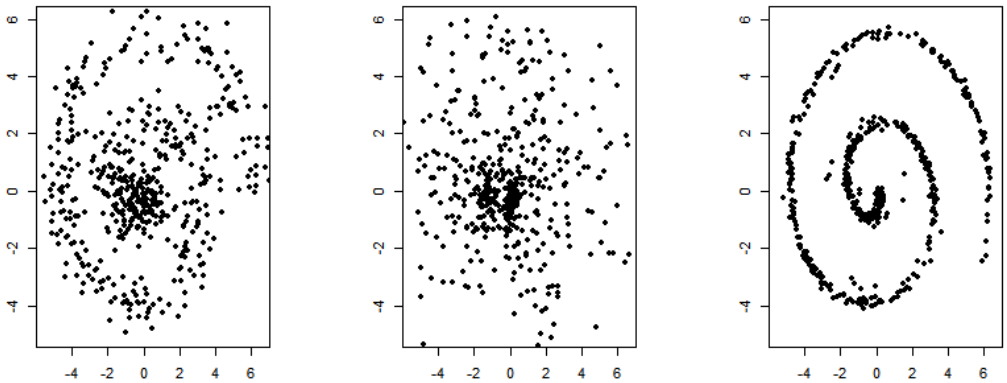
- Consider the above plots - data are concentrated near a non-linear support
- Standard toy examples in the manifold learning literature – serve as useful motivation
- What if we had non-linear structure like this in our data?
- How to model or estimate the density of the data?
- Real-world data very often have non-linear relationships among variables

Density estimation with non-linear support



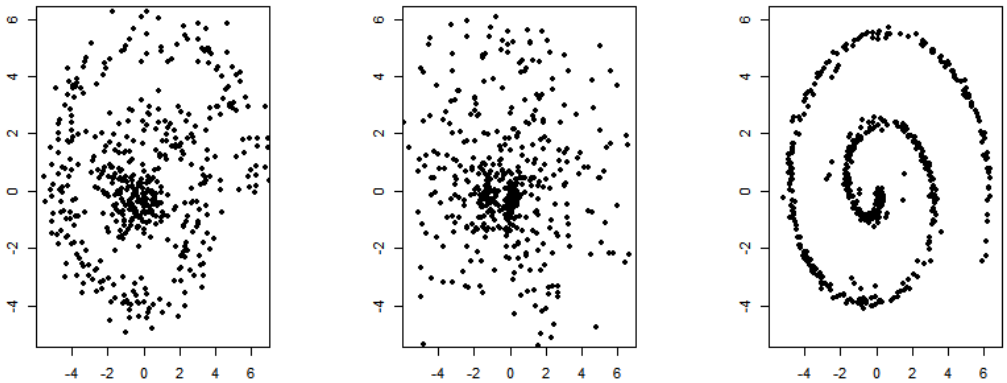
- Consider the above plots - data are concentrated near a non-linear support
- Standard toy examples in the manifold learning literature – serve as useful motivation
- What if we had non-linear structure like this in our data?
- How to model or estimate the density of the data?
- Real-world data very often have non-linear relationships among variables
- Are current density estimation methods adequate in the presence of such relationships?

Current density estimation methods



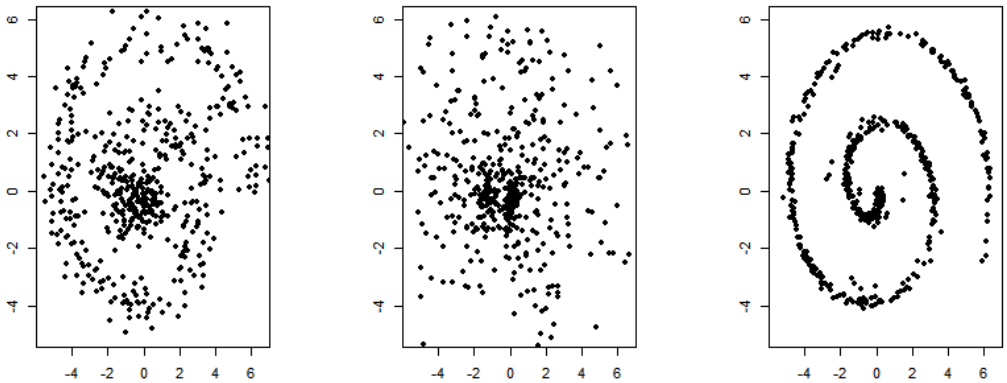
- Approach 1: nonlinear latent factor models: GP-LVMs, VAEs etc

Current density estimation methods



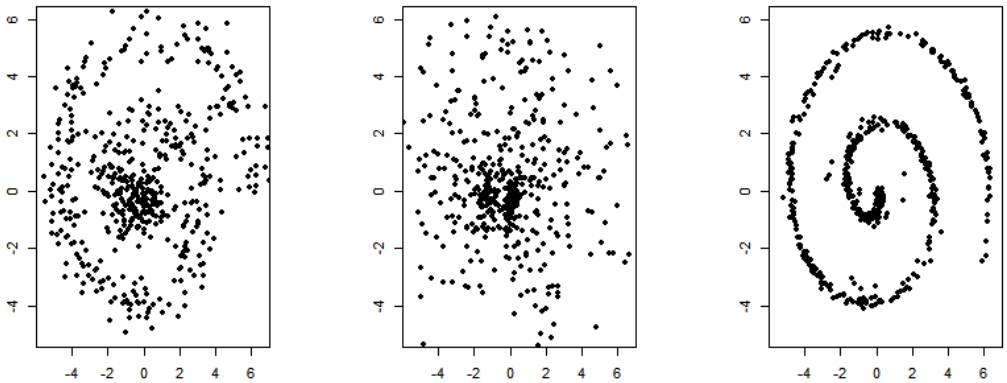
- Approach 1: nonlinear latent factor models: GP-LVMs, VAEs etc
- Approach 2: frequentist kernel density estimation (KDE)

Current density estimation methods



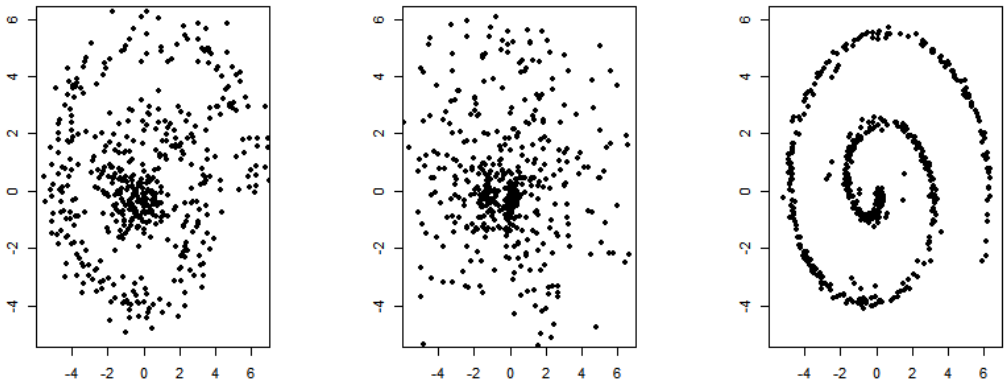
- Approach 1: nonlinear latent factor models: GP-LVMs, VAEs etc
- Approach 2: frequentist kernel density estimation (KDE)
- Approach 3: kernel mixture models (KMM), $f(y) = \sum_h \pi_h \mathcal{K}(y; \theta_h)$

Current density estimation methods



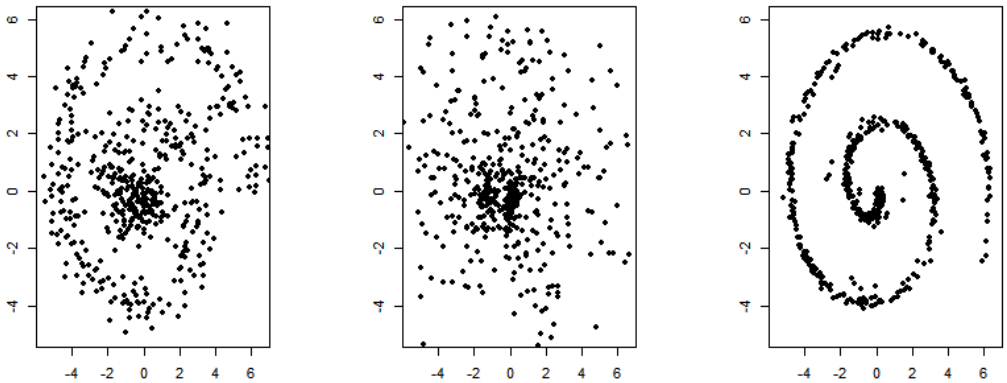
- Approach 1: nonlinear latent factor models: GP-LVMs, VAEs etc
- Approach 2: frequentist kernel density estimation (KDE)
- Approach 3: kernel mixture models (KMM), $f(y) = \sum_h \pi_h \mathcal{K}(y; \theta_h)$
- Approach 1 doesn't actually give an expression for the density so we rule it out

Current density estimation methods



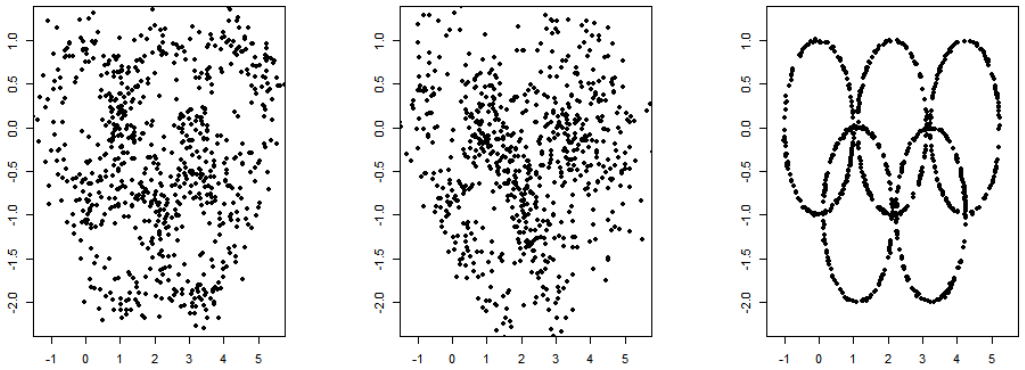
- Approach 1: nonlinear latent factor models: GP-LVMs, VAEs etc
- Approach 2: frequentist kernel density estimation (KDE)
- Approach 3: kernel mixture models (KMM), $f(y) = \sum_h \pi_h \mathcal{K}(y; \theta_h)$
- Approach 1 doesn't actually give an expression for the density so we rule it out
- LEFT (above) = samples from density estimated with KDE (for Euler spiral)

Current density estimation methods



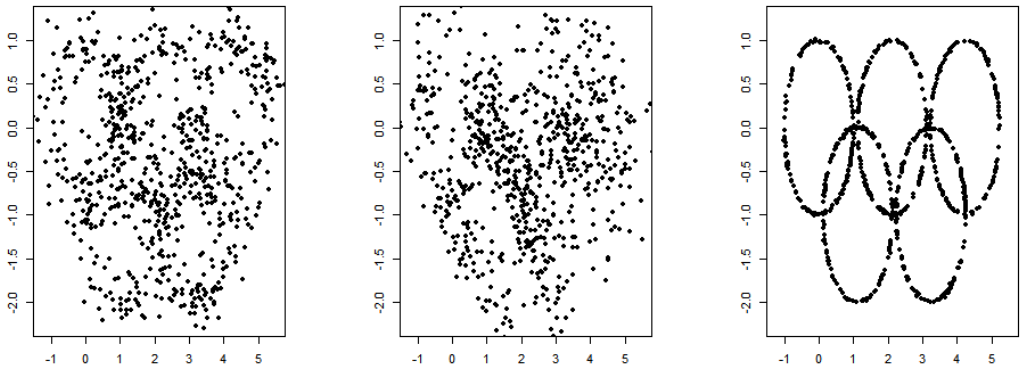
- Approach 1: nonlinear latent factor models: GP-LVMs, VAEs etc
- Approach 2: frequentist kernel density estimation (KDE)
- Approach 3: kernel mixture models (KMM), $f(y) = \sum_h \pi_h \mathcal{K}(y; \theta_h)$
- Approach 1 doesn't actually give an expression for the density so we rule it out
- LEFT (above) = samples from density estimated with KDE (for Euler spiral)
- MIDDLE = samples from density estimated with KMM; using Dirichlet process (DP) mixture of multivariate Gaussians

Current density estimation methods



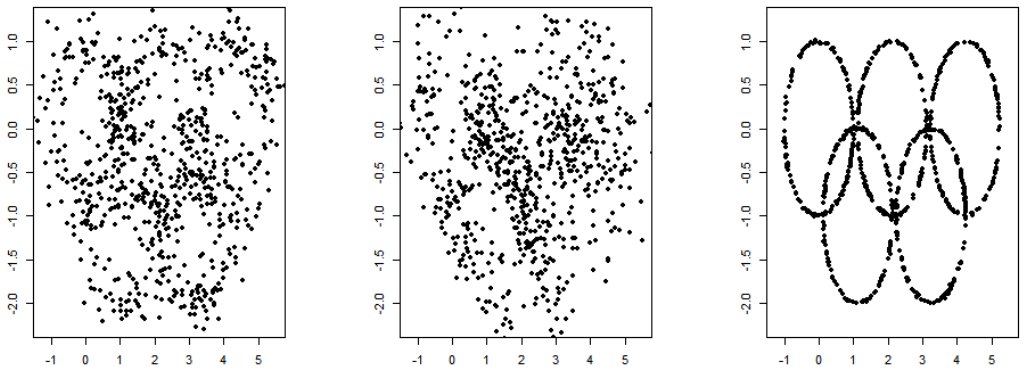
- KDE does poorly in this and other examples we considered of data concentrated near a non-linear support

Current density estimation methods



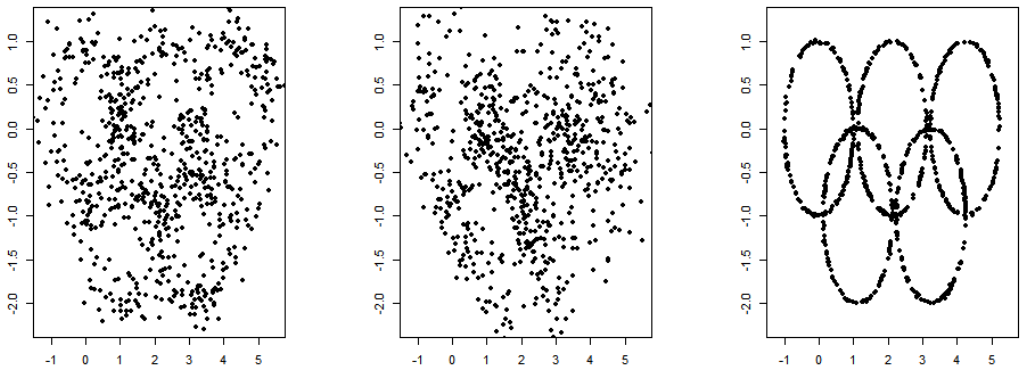
- KDE does poorly in this and other examples we considered of data concentrated near a non-linear support
- Sample size isn't small & we use current approaches for bandwidth selection in popular packages

Current density estimation methods



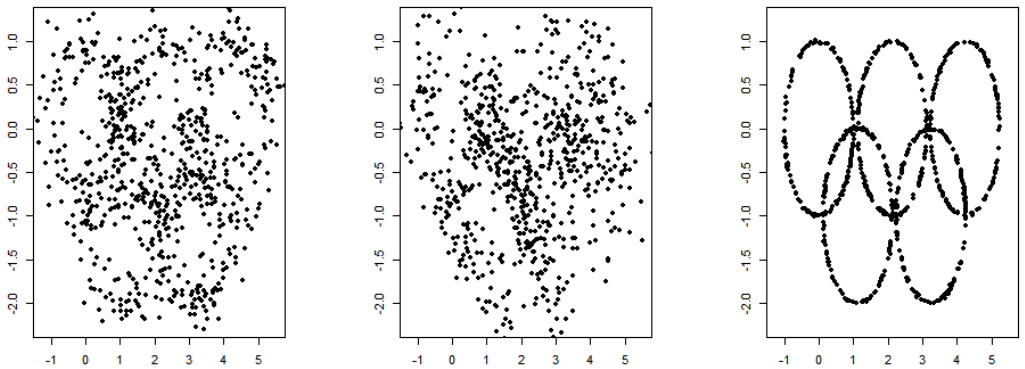
- KDE does poorly in this and other examples we considered of data concentrated near a non-linear support
- Sample size isn't small & we use current approaches for bandwidth selection in popular packages
- We expected Dirichlet process mixtures to do better but they actually do worse

Current density estimation methods



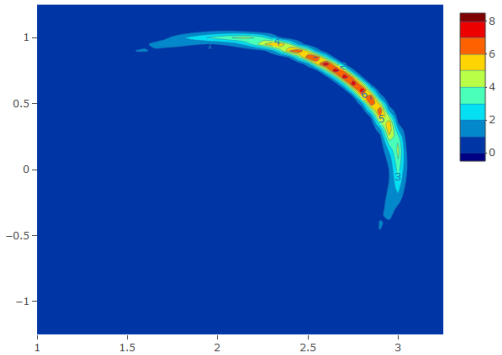
- KDE does poorly in this and other examples we considered of data concentrated near a non-linear support
- Sample size isn't small & we use current approaches for bandwidth selection in popular packages
- We expected Dirichlet process mixtures to do better but they actually do worse
- For multivariate Gaussian kernels, need tons of clusters/components of the mixture model

Current density estimation methods



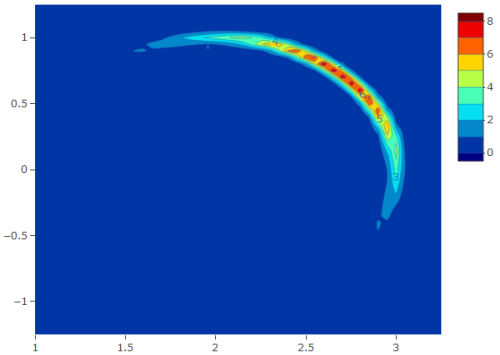
- KDE does poorly in this and other examples we considered of data concentrated near a non-linear support
- Sample size isn't small & we use current approaches for bandwidth selection in popular packages
- We expected Dirichlet process mixtures to do better but they actually do worse
- For multivariate Gaussian kernels, need tons of clusters/components of the mixture model
- MCMC mixing becomes a significant issue, as posterior sampling algorithms struggle here

Choosing a better kernel



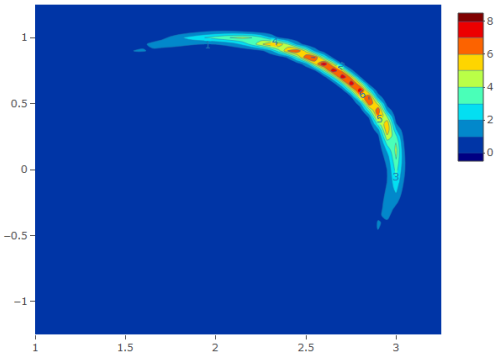
- Why does the Gaussian kernel suck so bad?

Choosing a better kernel



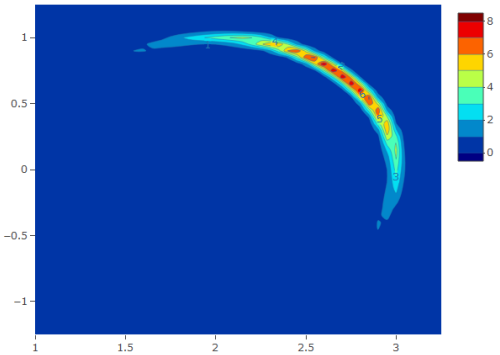
- Why does the Gaussian kernel suck so bad?
- Answer: Because it is essentially a locally linear approximation

Choosing a better kernel



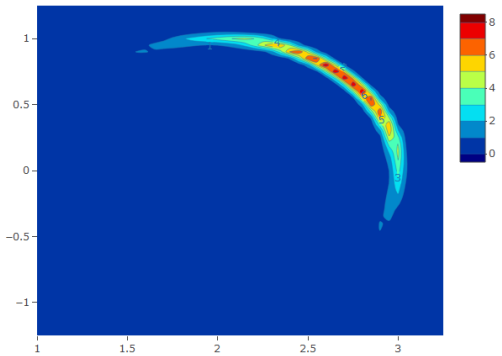
- Why does the Gaussian kernel suck so bad?
- Answer: Because it is essentially a locally linear approximation
- We have seen that locally linear algorithms often require many components

Choosing a better kernel



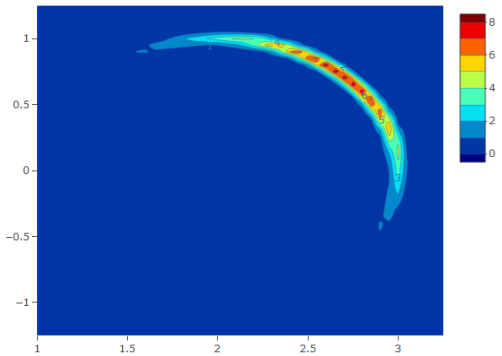
- Why does the Gaussian kernel suck so bad?
- Answer: Because it is essentially a locally linear approximation
- We have seen that locally linear algorithms often require many components
- This was the motivation for spherelets/SPCA

Choosing a better kernel



- Why does the Gaussian kernel suck so bad?
- Answer: Because it is essentially a locally linear approximation
- We have seen that locally linear algorithms often require many components
- This was the motivation for spherelets/SPCA
- So we need instead a kernel/pdf that accommodates curvature

Choosing a better kernel



- Why does the Gaussian kernel suck so bad?
- Answer: Because it is essentially a locally linear approximation
- We have seen that locally linear algorithms often require many components
- This was the motivation for spherelets/SPCA
- So we need instead a kernel/pdf that accommodates curvature
- There is a huge literature on kernels/multivariate pdfs but mostly on skewed, heavy tails etc

Fisher-Gaussian kernels [Mukhopadhyay, Li, D., 19']

- Couldn't find an appropriate kernel so we invented one

Fisher-Gaussian kernels [Mukhopadhyay, Li, D., 19']

- Couldn't find an appropriate kernel so we invented one
- The idea is very simple and inspired by spherelets/SPCA

Fisher-Gaussian kernels [Mukhopadhyay, Li, D., 19']

- Couldn't find an appropriate kernel so we invented one
- The idea is very simple and inspired by spherelets/SPCA
- First generate an observation y_i from a von Mises-Fisher (vMF) distribution on a sphere, with center c & radius r

Fisher-Gaussian kernels [Mukhopadhyay, Li, D., 19']

- Couldn't find an appropriate kernel so we invented one
- The idea is very simple and inspired by spherelets/SPCA
- First generate an observation y_i from a von Mises-Fisher (vMF) distribution on a sphere, with center c & radius r
- The vMF has a location (μ) and scale parameter (τ) & is essentially a spherical Gaussian constrained to the sphere

Fisher-Gaussian kernels [Mukhopadhyay, Li, D., 19']

- Couldn't find an appropriate kernel so we invented one
- The idea is very simple and inspired by spherelets/SPCA
- First generate an observation y_i from a von Mises-Fisher (vMF) distribution on a sphere, with center c & radius r
- The vMF has a location (μ) and scale parameter (τ) & is essentially a spherical Gaussian constrained to the sphere
- We then add a Gaussian measurement error $\epsilon_i \sim N(0, \sigma^2 I)$

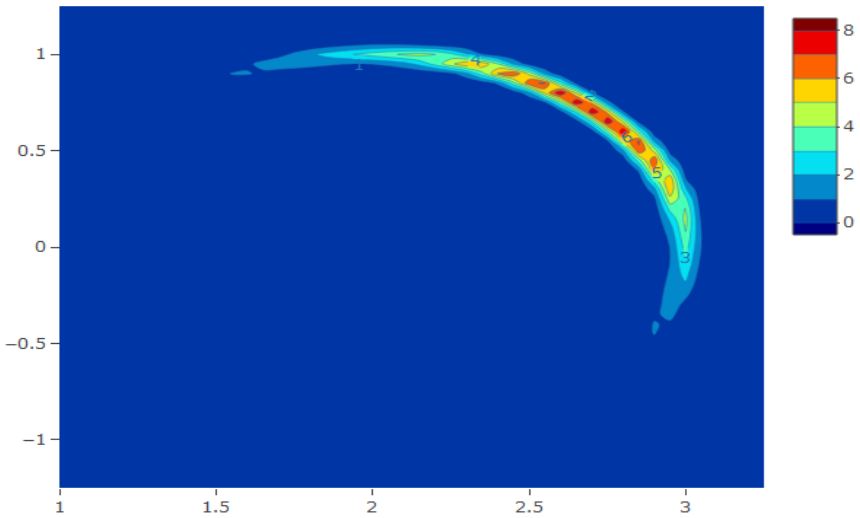
Fisher-Gaussian kernels [Mukhopadhyay, Li, D., 19']

- Couldn't find an appropriate kernel so we invented one
- The idea is very simple and inspired by spherelets/SPCA
- First generate an observation y_i from a von Mises-Fisher (vMF) distribution on a sphere, with center c & radius r
- The vMF has a location (μ) and scale parameter (τ) & is essentially a spherical Gaussian constrained to the sphere
- We then add a Gaussian measurement error $\epsilon_i \sim N(0, \sigma^2 I)$
- The resulting FG density has parameters $\rho = (c, r, \mu, \tau)$:

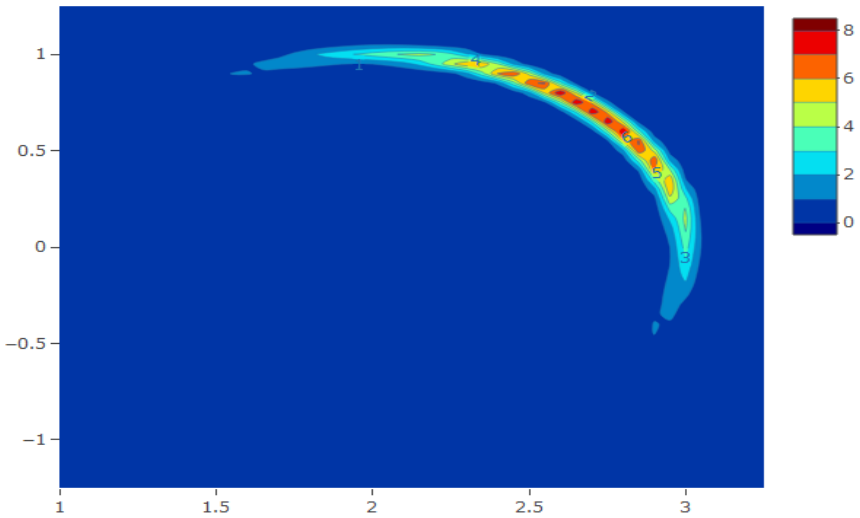
$$FG_\sigma(x \mid \rho) = \frac{c(\tau)(2\pi\sigma^2)^{-d/2}}{c\left(\|\tau\mu + r(x-c)x/\sigma^2\|\right)} \exp\left\{-\frac{1}{2\sigma^2}\left(\|x-c\|^2 + r^2\right)\right\},$$

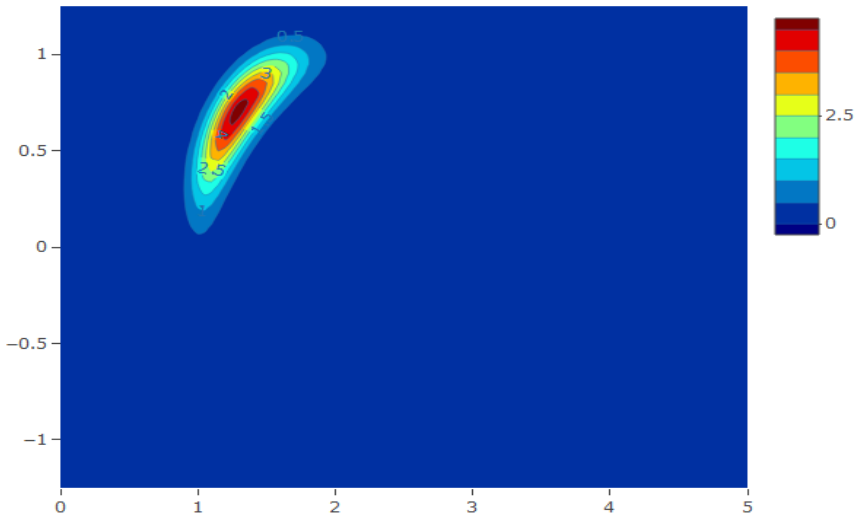
with $c(\tau) = \frac{\tau^{d/2-1}}{(2\pi)^{d/2} I_{d/2-1}(\tau)}$ & I_ν the modified Bessel function of the first kind of order ν

Examples of FG kernels

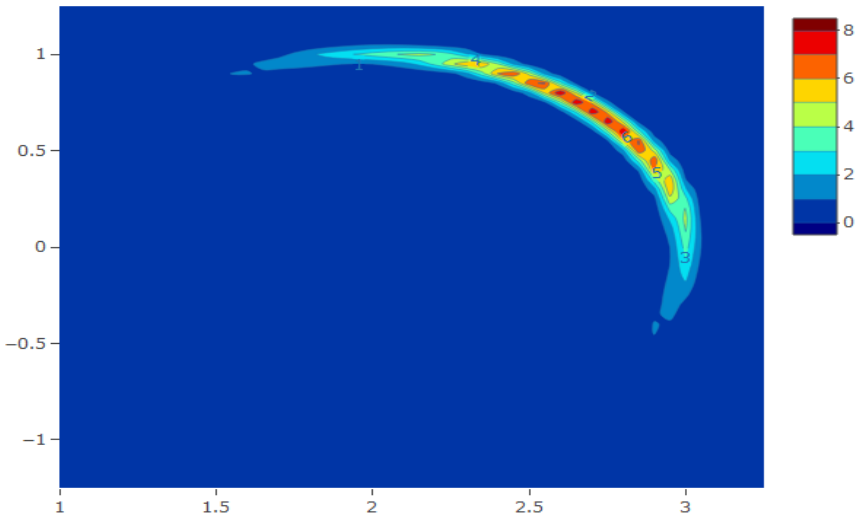


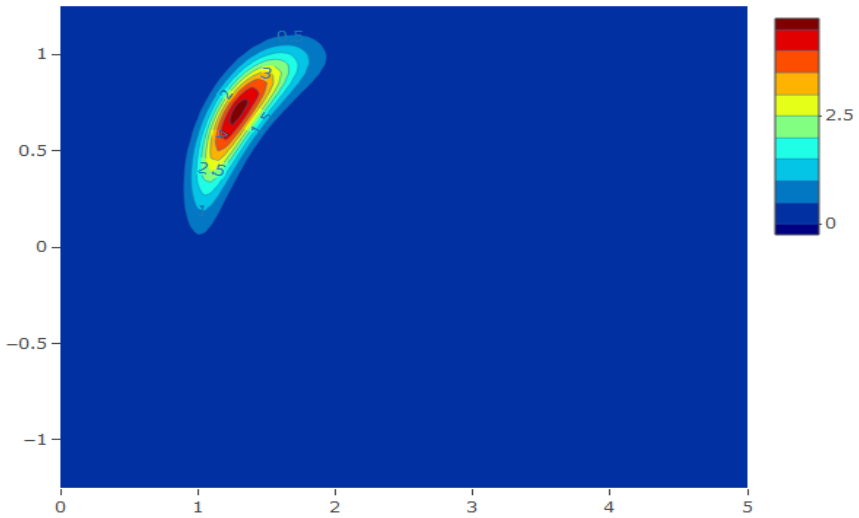
Examples of FG kernels

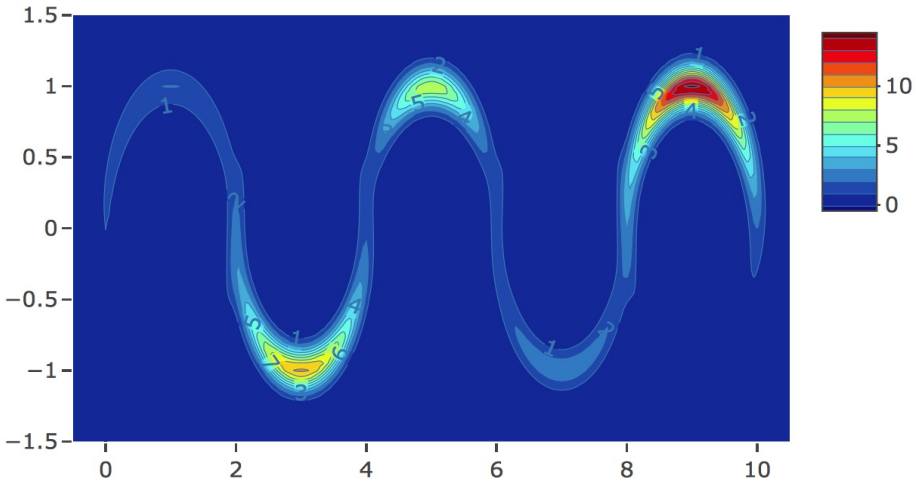




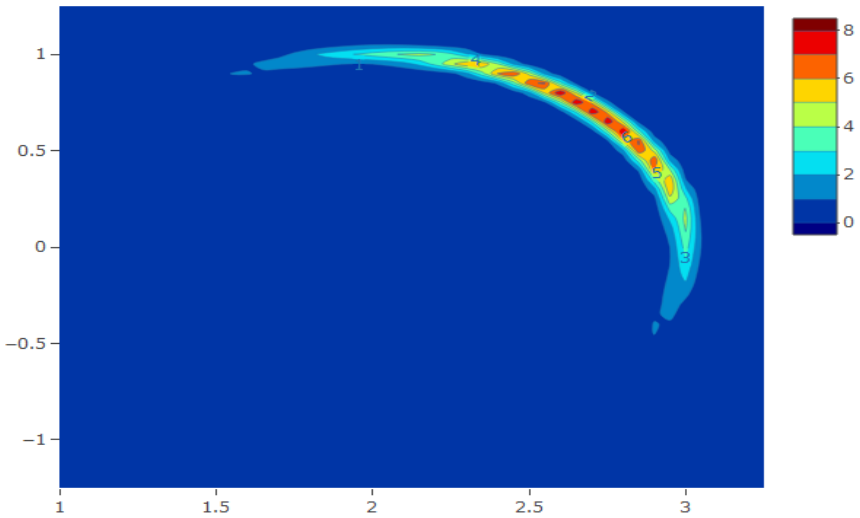
Examples of FG kernels

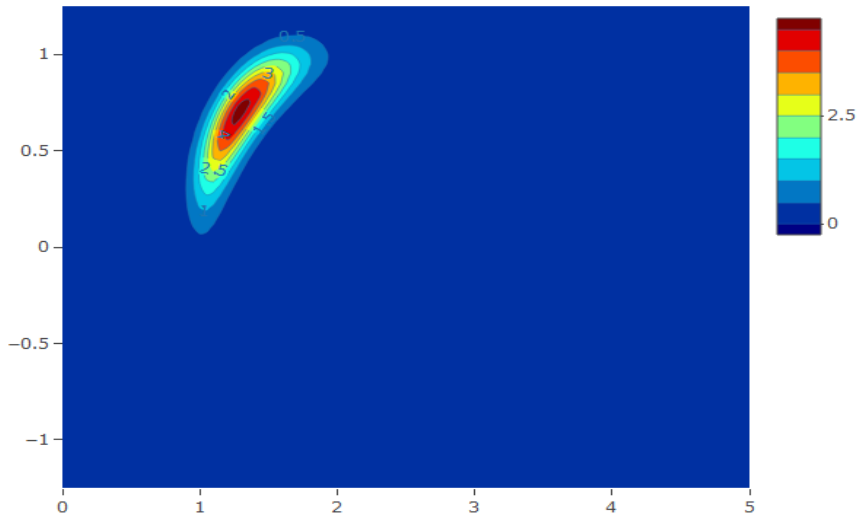


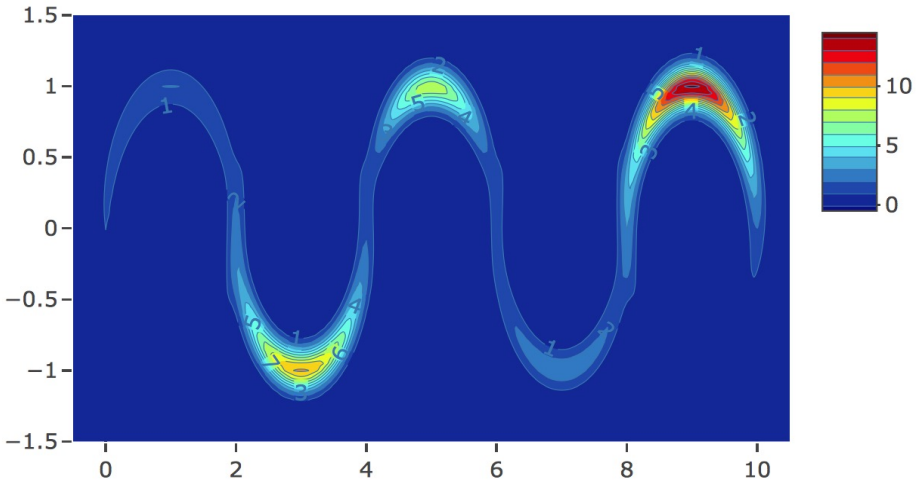


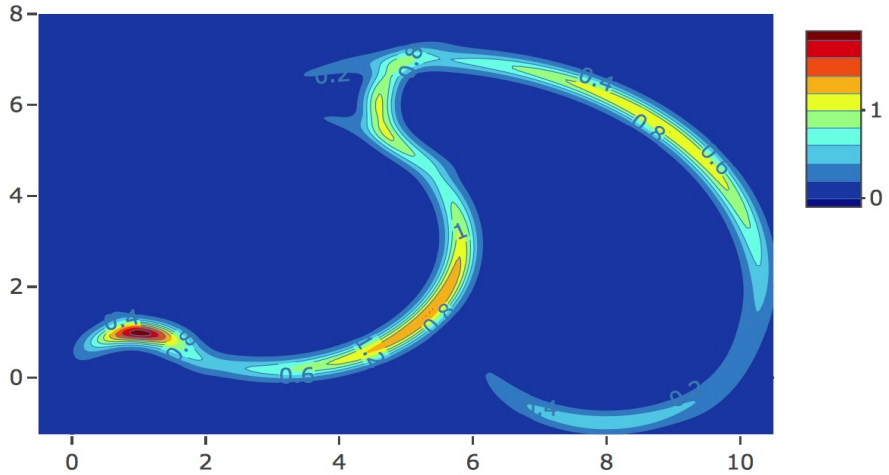


Examples of FG kernels

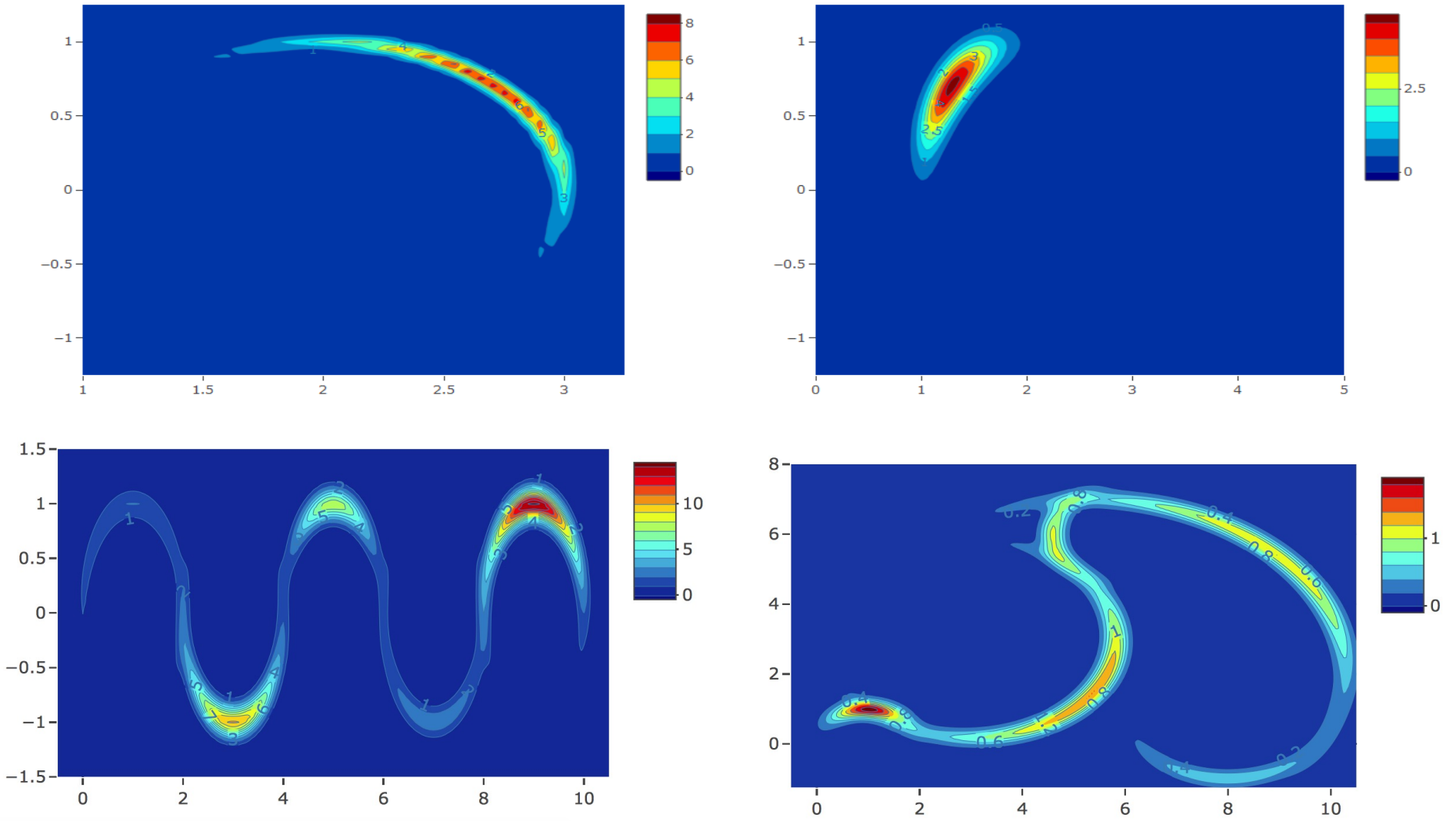






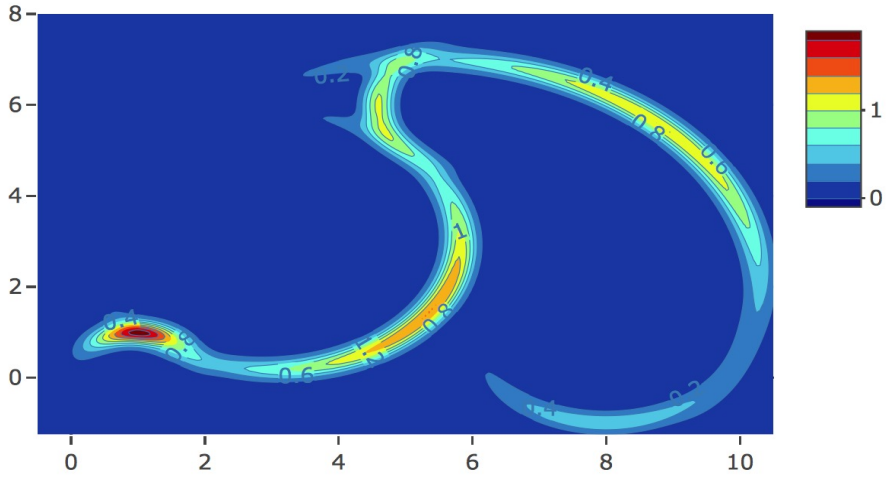


Examples of FG kernels



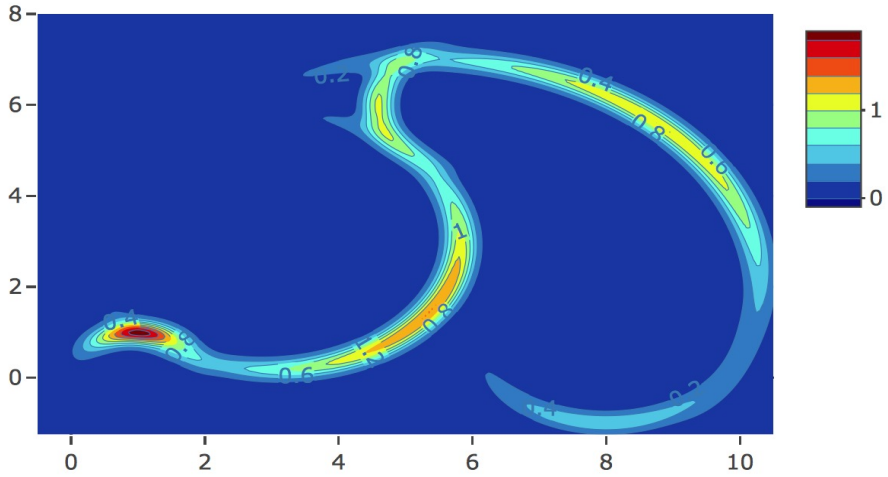
Two single FG kernels and two mixtures of five FG kernels.

FG Mixtures & Implementation



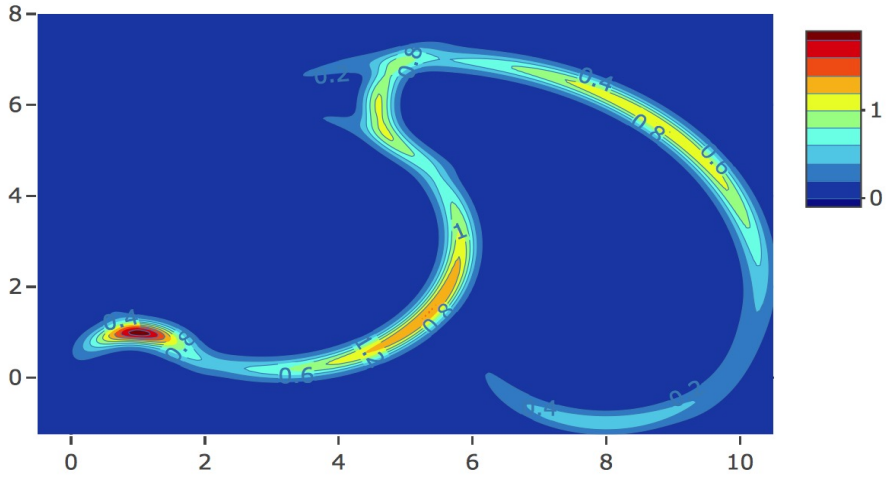
- We implemented FG kernel mixtures in a Bayesian analysis

FG Mixtures & Implementation



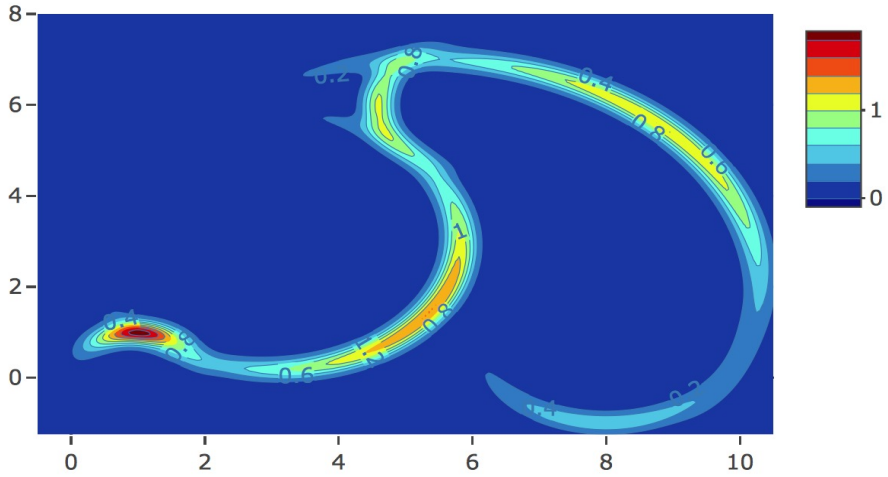
- We implemented FG kernel mixtures in a Bayesian analysis
- We used a two-layer formulation to allow multiple vMF components on each sphere

FG Mixtures & Implementation



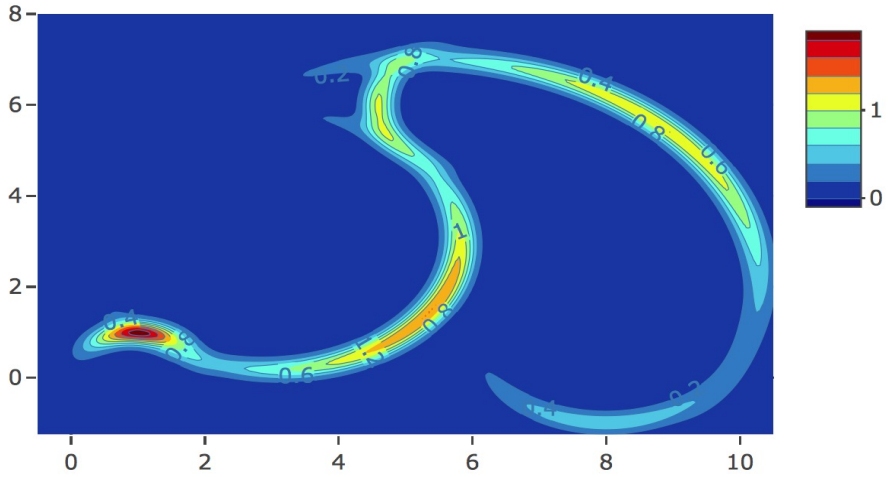
- We implemented FG kernel mixtures in a Bayesian analysis
- We used a two-layer formulation to allow multiple vMF components on each sphere
- Dirichlet process for the spheres and finite mixture on each sphere

FG Mixtures & Implementation



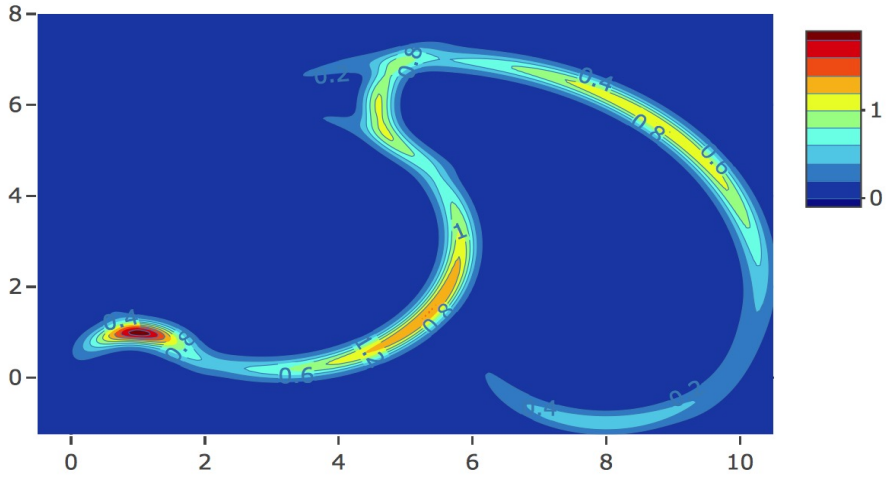
- We implemented FG kernel mixtures in a Bayesian analysis
- We used a two-layer formulation to allow multiple vMF components on each sphere
- Dirichlet process for the spheres and finite mixture on each sphere
- Large support & posterior consistency can be shown

FG Mixtures & Implementation



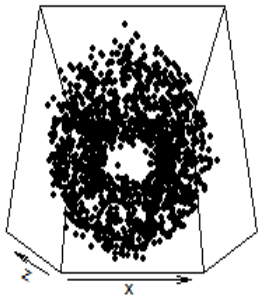
- We implemented FG kernel mixtures in a Bayesian analysis
- We used a two-layer formulation to allow multiple vMF components on each sphere
- Dirichlet process for the spheres and finite mixture on each sphere
- Large support & posterior consistency can be shown
- Conditionally conjugate priors can be chosen leading to a Gibbs sampler with one MH step

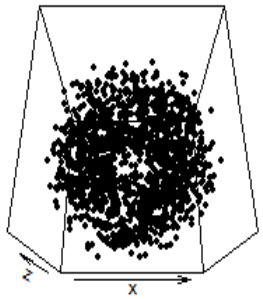
FG Mixtures & Implementation

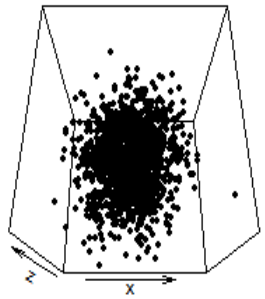


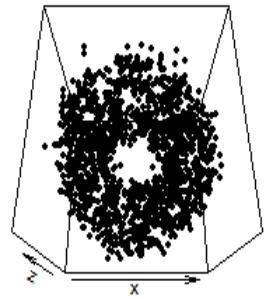
- We implemented FG kernel mixtures in a Bayesian analysis
- We used a two-layer formulation to allow multiple vMF components on each sphere
- Dirichlet process for the spheres and finite mixture on each sphere
- Large support & posterior consistency can be shown
- Conditionally conjugate priors can be chosen leading to a Gibbs sampler with one MH step
- Let's see how it works in practice!

Predictive samples

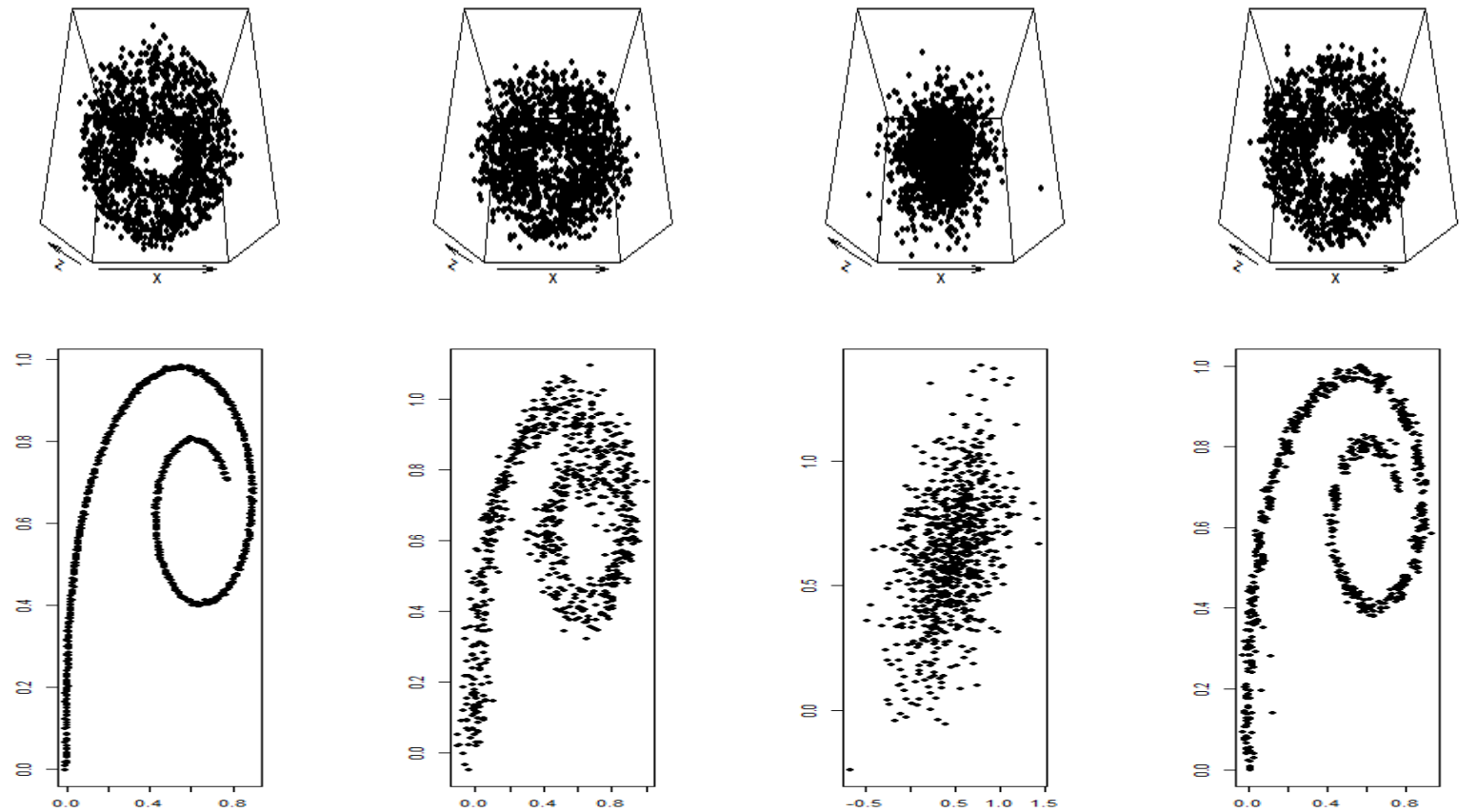






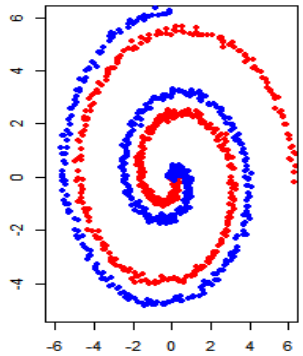


Predictive samples

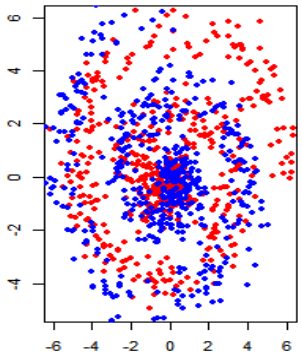


(a) Observations (b) KDE (c) DPMG (d) DPMFG

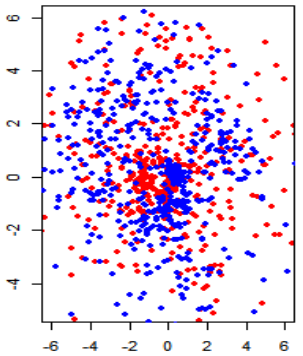
Two Spiral - Density based classification



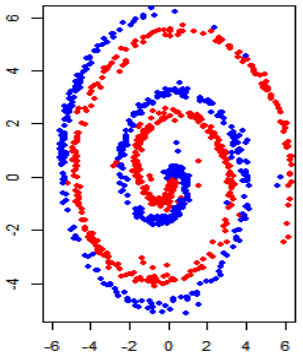
(a) Observations



(b) KDE

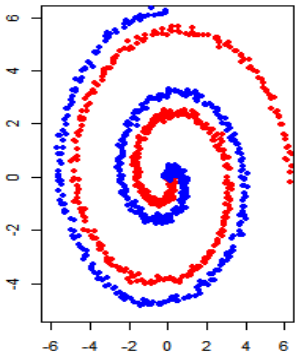


(c) DPMG

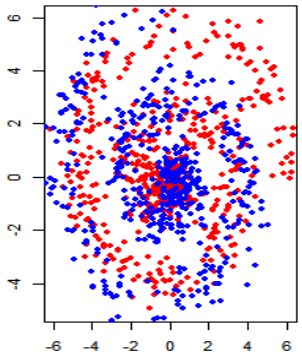


(d) DPMFG

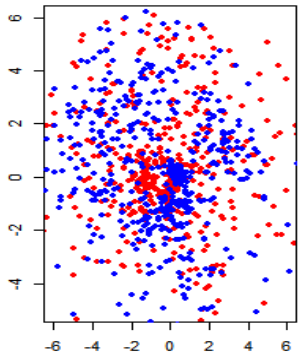
Two Spiral - Density based classification



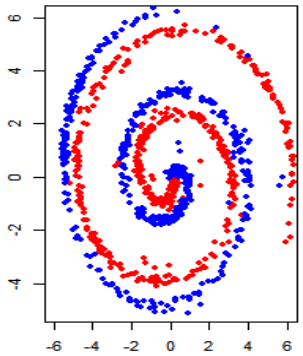
(a) Observations



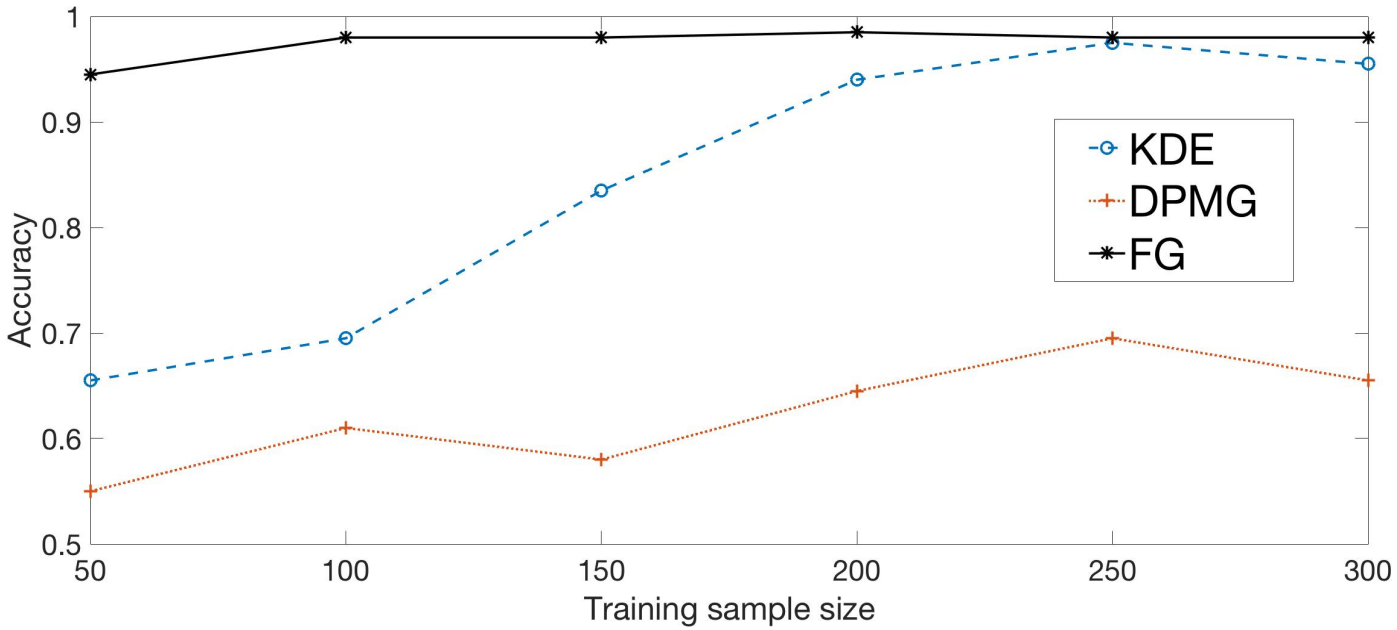
(b) KDE



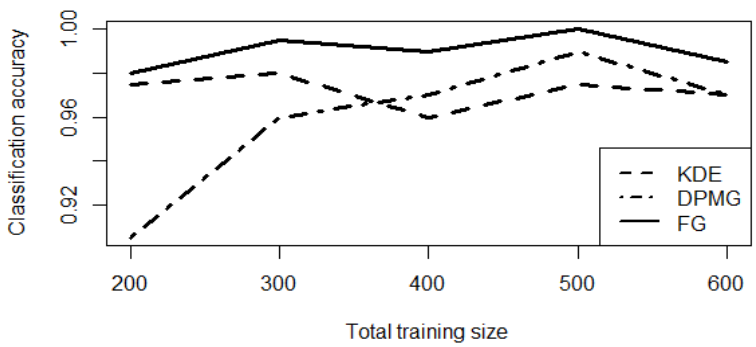
(c) DPMG

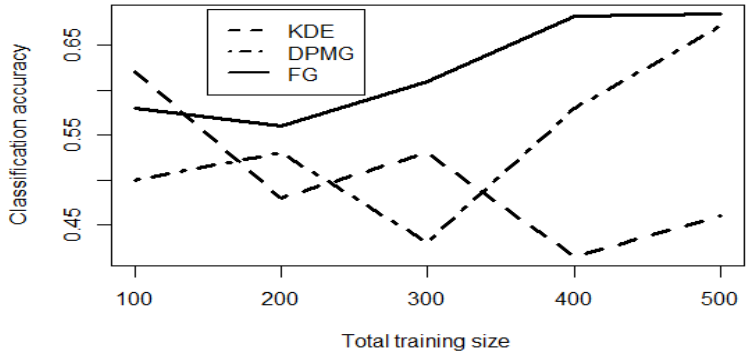


(d) DPMFG



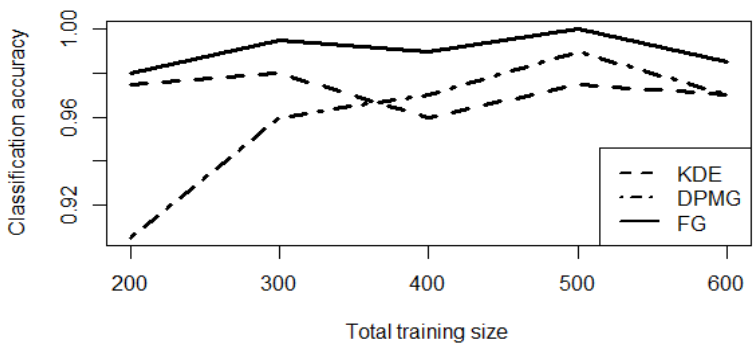
Galaxy Zoo and Balance scale datasets

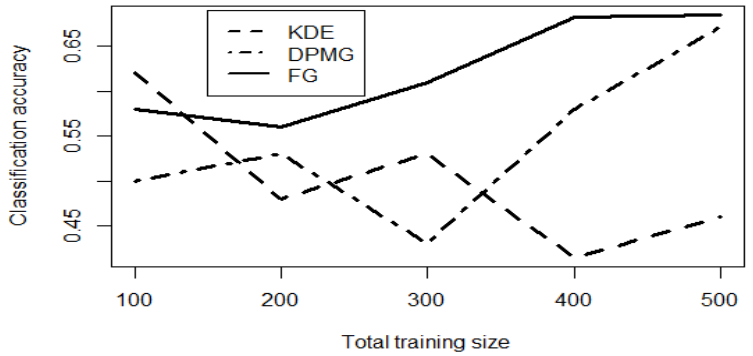




Accuracies for the Galaxy (left panel) and Balance scale data (right panel).

Galaxy Zoo and Balance scale datasets

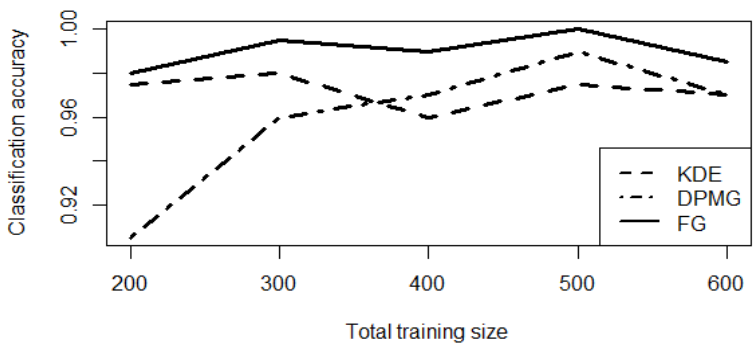


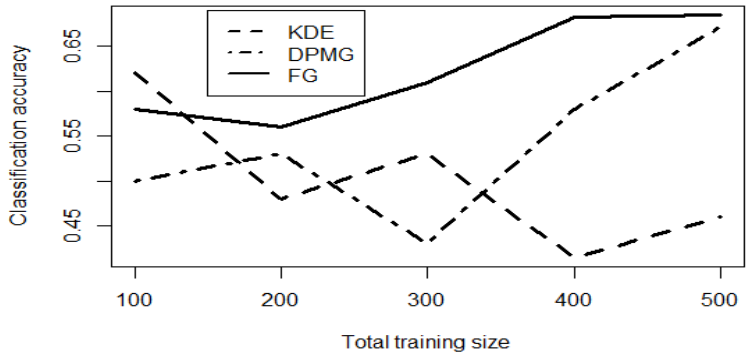


Accuracies for the Galaxy (left panel) and Balance scale data (right panel).

- Galaxy Zoo project: volunteers classified images of Sloan Digital Sky Survey galaxies as spiral, elliptical or uncertain

Galaxy Zoo and Balance scale datasets

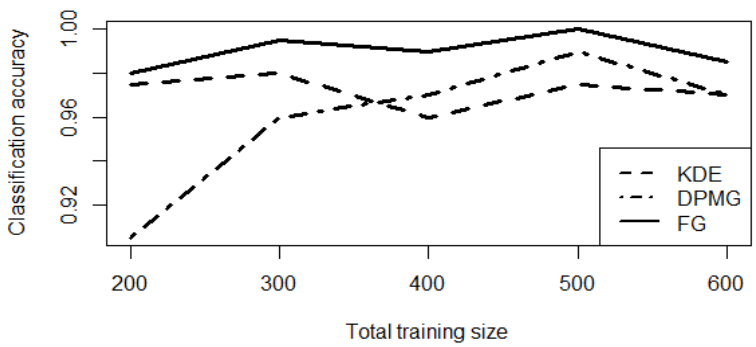


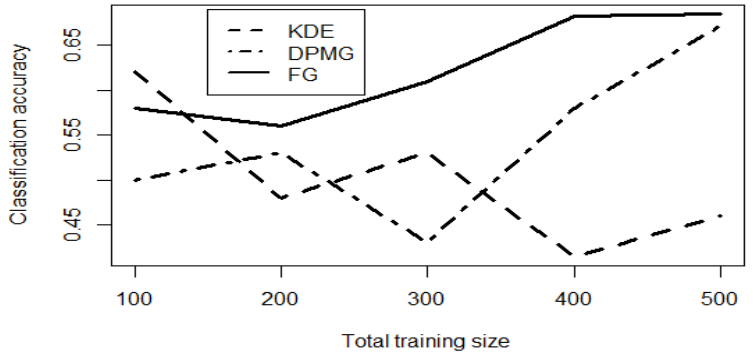


Accuracies for the Galaxy (left panel) and Balance scale data (right panel).

- Galaxy Zoo project: volunteers classified images of Sloan Digital Sky Survey galaxies as spiral, elliptical or uncertain
- Balance scale data: from psychological experiments. Each example is classified as having the balance scale tip to the right, left, or balanced

Galaxy Zoo and Balance scale datasets





Accuracies for the Galaxy (left panel) and Balance scale data (right panel).

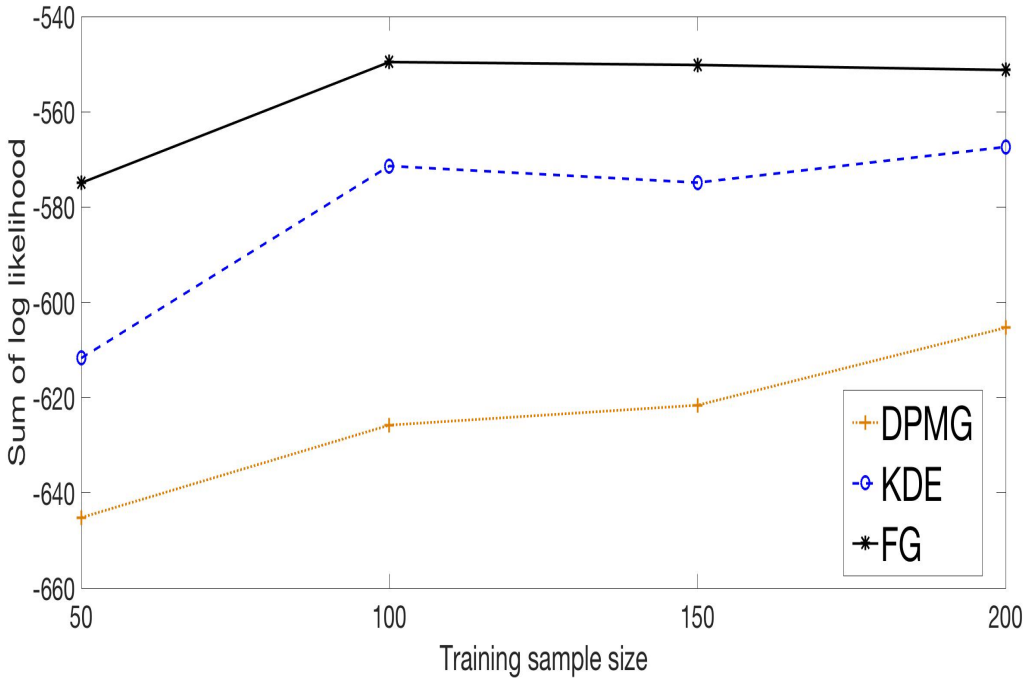
- Galaxy Zoo project: volunteers classified images of Sloan Digital Sky Survey galaxies as spiral, elliptical or uncertain
- Balance scale data: from psychological experiments. Each example is classified as having the balance scale tip to the right, left, or balanced
- Gesture phase data: 32 features from 7 videos of people gesticulating, aimed at studying gesture phase segmentation. FG had much better classification accuracy (results in paper)

Density estimation

- We also consider the sum of log likelihood of the test data based on the estimated density (the larger the better)

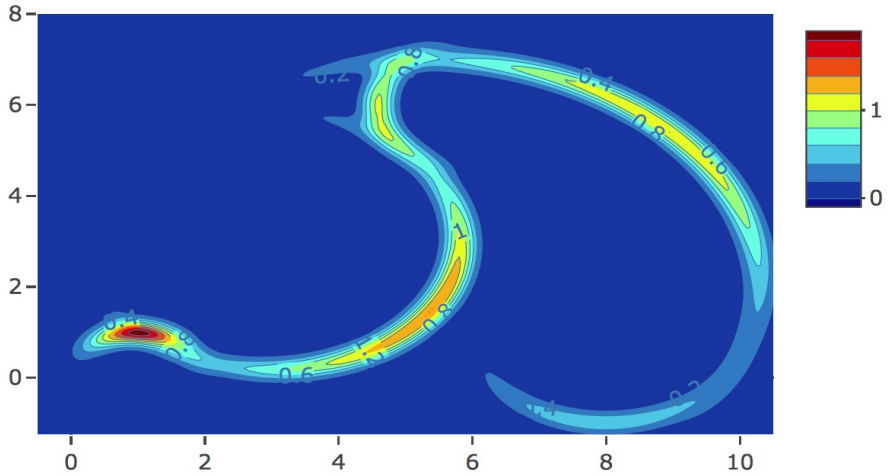
Density estimation

- We also consider the sum of log likelihood of the test data based on the estimated density (the larger the better)



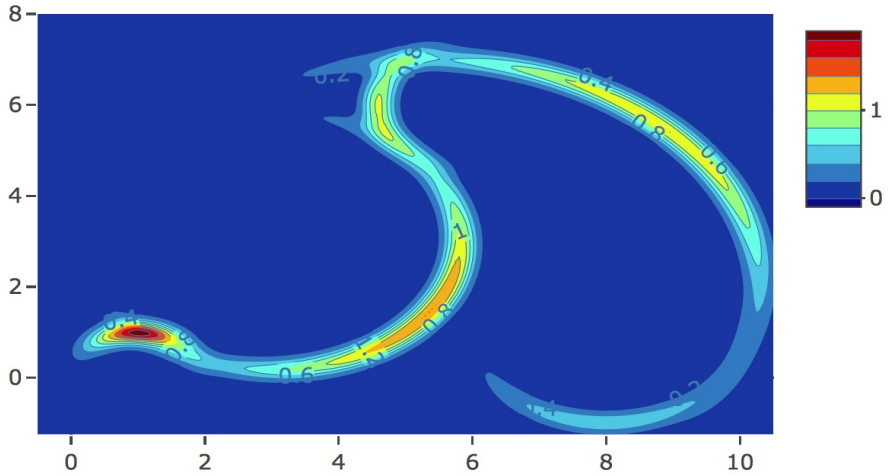
Sum of log likelihood for the Balance scale.

FG mixtures - comments & where to go now?



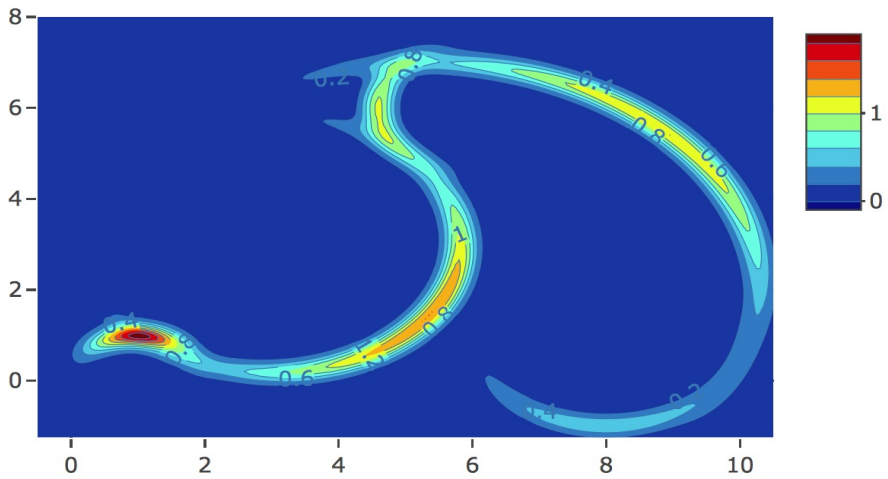
- Fisher-Gaussian distribution seems to be a useful addition to the literature

FG mixtures - comments & where to go now?



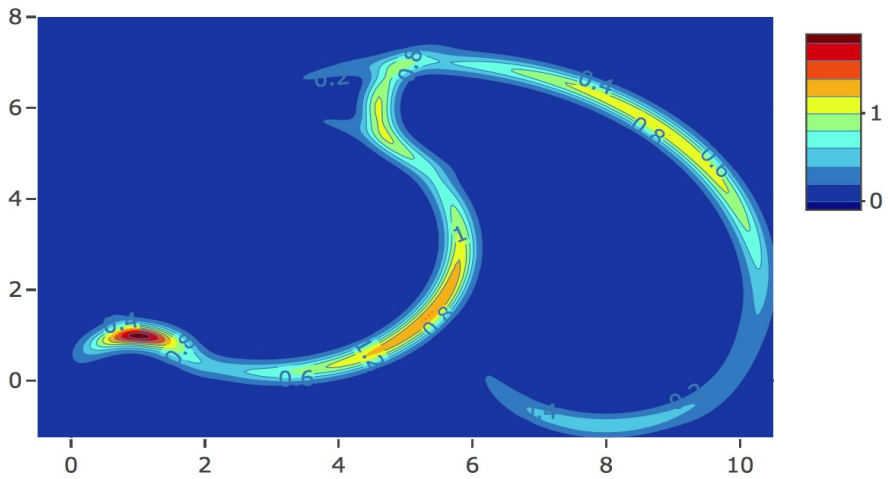
- Fisher-Gaussian distribution seems to be a useful addition to the literature
- Can be used for multivariate modeling & in other settings

FG mixtures - comments & where to go now?



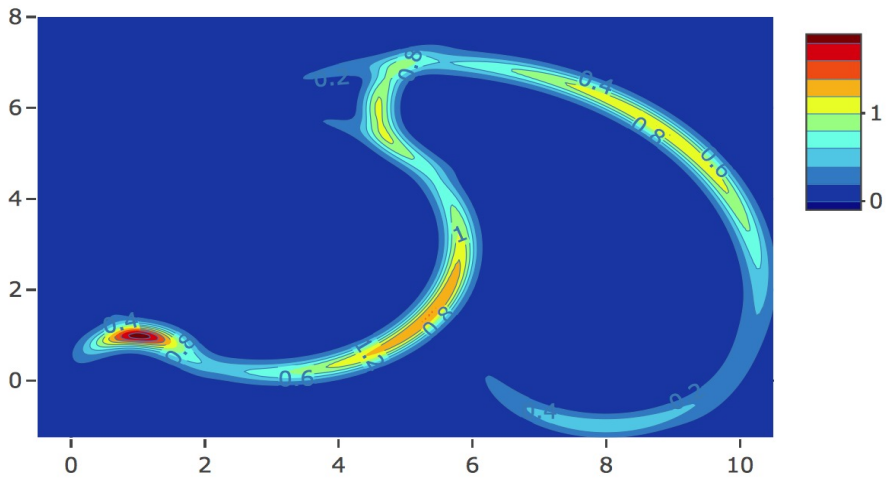
- Fisher-Gaussian distribution seems to be a useful addition to the literature
- Can be used for multivariate modeling & in other settings
- Interesting to further generalize it but one may loose the simplicity

FG mixtures - comments & where to go now?



- Fisher-Gaussian distribution seems to be a useful addition to the literature
- Can be used for multivariate modeling & in other settings
- Interesting to further generalize it but one may loose the simplicity
- FG kernels were inspired by spherelets/SPCA

FG mixtures - comments & where to go now?



- Fisher-Gaussian distribution seems to be a useful addition to the literature
- Can be used for multivariate modeling & in other settings
- Interesting to further generalize it but one may loose the simplicity
- FG kernels were inspired by spherelets/SPCA
- In the remainder, I will discuss two additional methods inspired by spherelets - for classification & distance estimation

Changing gears - A new approach for classification

- Given labeled data $\{(x_i, y_i)\}_{i=1}^n$, where $x_i \in \mathbb{R}^p$, $y_i \in \{1, 2, \cdots, L\}$

Changing gears - A new approach for classification

- Given labeled data $\{(x_i, y_i)\}_{i=1}^n$, where $x_i \in \mathbb{R}^p$, $y_i \in \{1, 2, \cdots, L\}$
- Classifier: takes features x as input & outputs a class label y

Changing gears - A new approach for classification

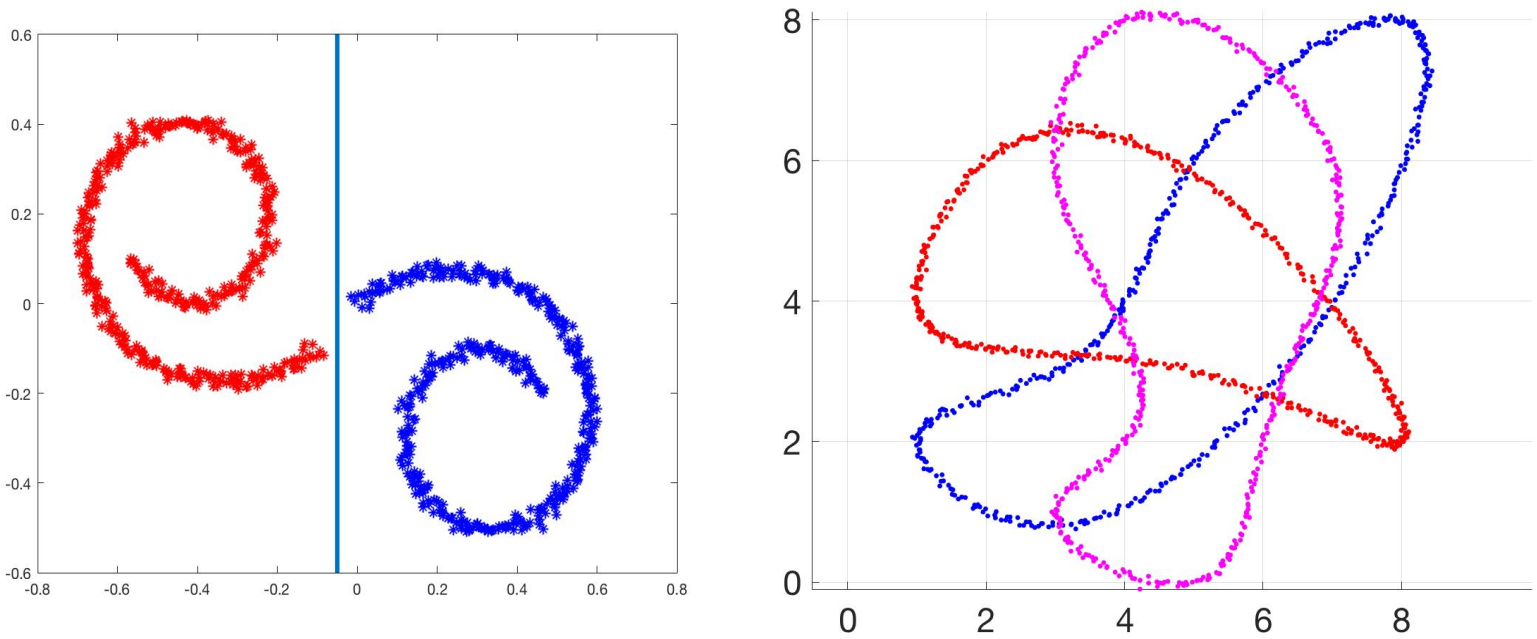
- Given labeled data $\{(x_i, y_i)\}_{i=1}^n$, where $x_i \in \mathbb{R}^p$, $y_i \in \{1, 2, \cdots, L\}$
- Classifier: takes features x as input & outputs a class label y
- There is a huge literature, but methods struggle when feature distributions have non-linear, intersecting and 'entangled' supports

Changing gears - A new approach for classification

- Given labeled data $\{(x_i, y_i)\}_{i=1}^n$, where $x_i \in \mathbb{R}^p$, $y_i \in \{1, 2, \cdots, L\}$
- Classifier: takes features x as input & outputs a class label y
- There is a huge literature, but methods struggle when feature distributions have non-linear, intersecting and 'entangled' supports
- Neural networks can potentially deal with this IF sample size n is HUGE

Changing gears - A new approach for classification

- Given labeled data $\{(x_i, y_i)\}_{i=1}^n$, where $x_i \in \mathbb{R}^p$, $y_i \in \{1, 2, \cdots, L\}$
- Classifier: takes features x as input & outputs a class label y
- There is a huge literature, but methods struggle when feature distributions have non-linear, intersecting and 'entangled' supports
- Neural networks can potentially deal with this IF sample size n is HUGE



Separable and inseparable cases.

Local Manifold Approximation (LOMA) Classification

- Assume points with label j are close to manifold M_j

Local Manifold Approximation (LOMA) Classification

- Assume points with label j are close to manifold M_j
- Assign x to the class j having the smallest distance $d(x, M_j)$

Local Manifold Approximation (LOMA) Classification

- Assume points with label j are close to manifold M_j
- Assign x to the class j having the smallest distance $d(x, M_j)$
- M_j is unknown & there is typically measurement error

Local Manifold Approximation (LOMA) Classification

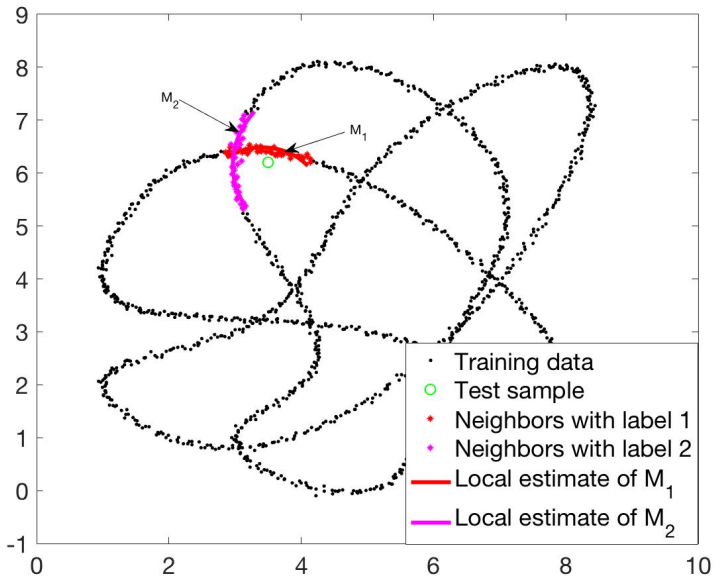
- Assume points with label j are close to manifold M_j
- Assign x to the class j having the smallest distance $d(x, M_j)$
- M_j is unknown & there is typically measurement error
- Assign x to the class with the smallest distance $d(x, \widehat{M}_j)$

Local Manifold Approximation (LOMA) Classification

- Assume points with label j are close to manifold M_j
- Assign x to the class j having the smallest distance $d(x, M_j)$
- M_j is unknown & there is typically measurement error
- Assign x to the class with the smallest distance $d(x, \widehat{M}_j)$
- $d(x, \hat{M}_j)$ only depends on *local* estimation of M_j

Local Manifold Approximation (LOMA) Classification

- Assume points with label j are close to manifold M_j
- Assign x to the class j having the smallest distance $d(x, M_j)$
- M_j is unknown & there is typically measurement error
- Assign x to the class with the smallest distance $d(x, \widehat{M}_j)$
- $d(x, \hat{M}_j)$ only depends on *local* estimation of M_j



SPherical Approximation (SPA) classifier

- To obtain the local estimates of M_j within a neighborhood of x, we use a sphere

SPherical Approximation (SPA) classifier

- To obtain the local estimates of M_j within a neighborhood of x, we use a sphere
- Special case of LOMA classifier - using other local manifold estimates produces slightly different classifiers

SPherical Approximation (SPA) classifier

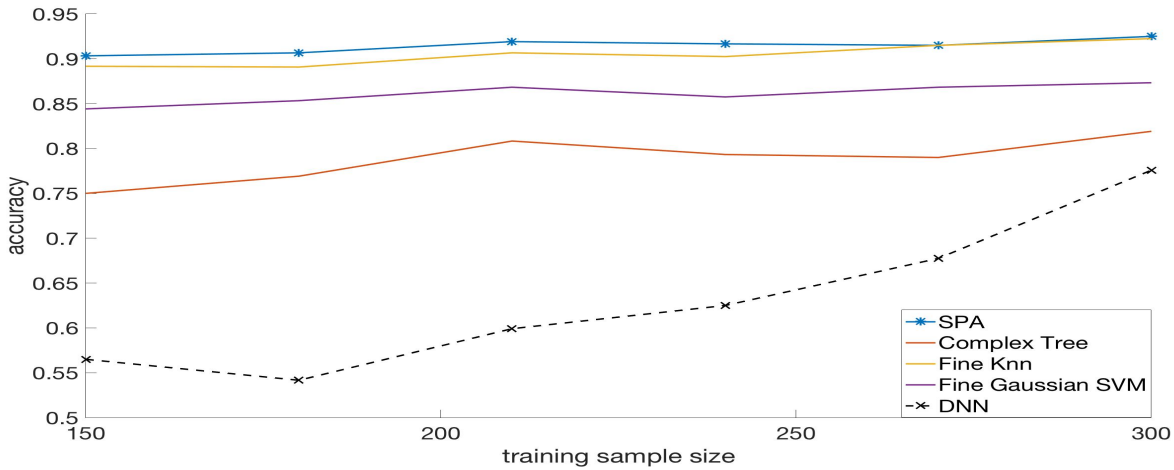
- To obtain the local estimates of M_j within a neighborhood of x, we use a sphere
- Special case of LOMA classifier - using other local manifold estimates produces slightly different classifiers
- Sphere is convenient computationally & has much better performance than linear approximations

SPherical Approximation (SPA) classifier

- To obtain the local estimates of M_j within a neighborhood of x, we use a sphere
- Special case of LOMA classifier - using other local manifold estimates produces slightly different classifiers
- Sphere is convenient computationally & has much better performance than linear approximations
- We refer to this simple algorithm as the SPherical Approximation (SPA) classifier

SPherical Approximation (SPA) classifier

- To obtain the local estimates of M_j within a neighborhood of x, we use a sphere
- Special case of LOMA classifier - using other local manifold estimates produces slightly different classifiers
- Sphere is convenient computationally & has much better performance than linear approximations
- We refer to this simple algorithm as the SPherical Approximation (SPA) classifier



Funky curves data, sample size v.s. accuracy plot

Asymptotic accuracy bound

- Data are often noisy due to measurement error

Asymptotic accuracy bound

- Data are often noisy due to measurement error
- Assume $x_i = z_i + \epsilon_i$, where $\epsilon_i \sim N(0, \sigma^2 I_p)$ is the noise

Asymptotic accuracy bound

- Data are often noisy due to measurement error
- Assume $x_i = z_i + \epsilon_i$, where $\epsilon_i \sim N(0, \sigma^2 I_p)$ is the noise
- $\{z_i\}_{i=1}^n \overset{iid}{\sim} \rho$ and $\text{supp}(\rho) = M_1 \cup M_2$, $z_i \in M_{y_i}$

Asymptotic accuracy bound

- Data are often noisy due to measurement error
- Assume $x_i = z_i + \epsilon_i$, where $\epsilon_i \sim N(0, \sigma^2 I_p)$ is the noise
- $\{z_i\}_{i=1}^n \overset{iid}{\sim} \rho$ and $\text{supp}(\rho) = M_1 \cup M_2$, $z_i \in M_{y_i}$
- Let y be the true label, $\widehat{y}_n$ be the predictive label given by SPA

Asymptotic accuracy bound

- Data are often noisy due to measurement error
- Assume $x_i = z_i + \epsilon_i$, where $\epsilon_i \sim N(0, \sigma^2 I_p)$ is the noise
- $\{z_i\}_{i=1}^n \overset{iid}{\sim} \rho$ and $\text{supp}(\rho) = M_1 \cup M_2$, $z_i \in M_{y_i}$
- Let y be the true label, $\widehat{y}_n$ be the predictive label given by SPA

Theorem (Li, D., 19')

Let $\delta > 0$ and $B_\delta(M) := \{x \mid d(x, M) < \delta\}$, then $\lim_{n\to\infty} \mathbb{P}(y \neq \widehat{y}_n)$ is upper bounded by

$$\rho(B_\delta(M_1) \cap B_\delta(M_2)) + \exp\left\{-\frac{\delta^2}{8\sigma^2} + \frac{p}{2}\log\left(\frac{\delta^2}{4\sigma^2}\right) - \frac{p}{2}(\log(p) - 1)\right\}.$$

Asymptotic accuracy bound

- Data are often noisy due to measurement error
- Assume $x_i = z_i + \epsilon_i$, where $\epsilon_i \sim N(0, \sigma^2 I_p)$ is the noise
- $\{z_i\}_{i=1}^n \overset{iid}{\sim} \rho$ and $\text{supp}(\rho) = M_1 \cup M_2$, $z_i \in M_{y_i}$
- Let y be the true label, $\widehat{y}_n$ be the predictive label given by SPA

Theorem (Li, D., 19')

Let $\delta > 0$ and $B_\delta(M) := \{x \mid d(x, M) < \delta\}$, then $\lim_{n\to\infty} \mathbb{P}(y \neq \widehat{y}_n)$ is upper bounded by

$$\rho(B_\delta(M_1) \cap B_\delta(M_2)) + \exp\left\{-\frac{\delta^2}{8\sigma^2} + \frac{p}{2}\log\left(\frac{\delta^2}{4\sigma^2}\right) - \frac{p}{2}(\log(p) - 1)\right\}.$$

- As $\sigma \to 0$, the upper bound degenerates to $\rho(M_1 \cap M_2)$.

Image data

- For image data, p is very large so n may need to be enormous for good performance

Image data

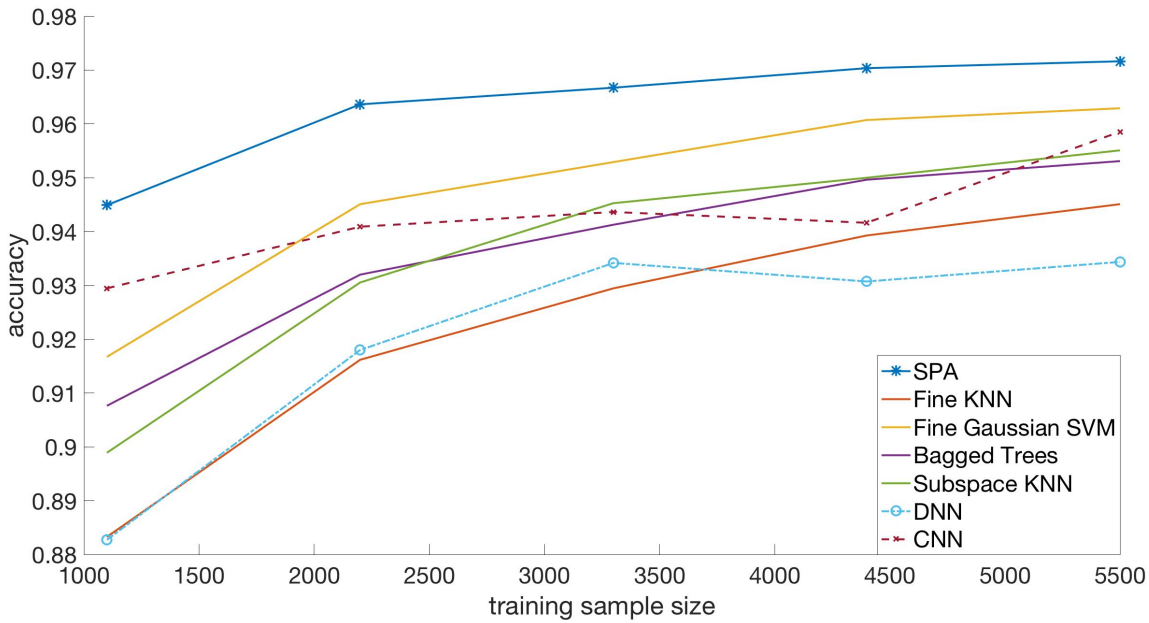
- For image data, p is very large so n may need to be enormous for good performance
- Convolutional Neural Networks (CNNs) great when n is HUGE

Image data

- For image data, p is very large so n may need to be enormous for good performance
- Convolutional Neural Networks (CNNs) great when n is HUGE
- SPA does much better for small to moderate n in many cases

Image data

- For image data, p is very large so n may need to be enormous for good performance
- Convolutional Neural Networks (CNNs) great when n is HUGE
- SPA does much better for small to moderate n in many cases



USPS hand written digits data set

Wrapping up Classification & on to Distance estimation

- *Labeled* data: LOMA and SPA work well when supports are entangled and sample size is small-to-moderate

Wrapping up Classification & on to Distance estimation

- *Labeled* data: LOMA and SPA work well when supports are entangled and sample size is small-to-moderate
- Promising new type of nearest neighbors classifier; quite different from current competitors

Wrapping up Classification & on to Distance estimation

- *Labeled* data: LOMA and SPA work well when supports are entangled and sample size is small-to-moderate
- Promising new type of nearest neighbors classifier; quite different from current competitors
- As the last topic, switch gears: classification $\rightarrow$ clustering

Wrapping up Classification & on to Distance estimation

- *Labeled* data: LOMA and SPA work well when supports are entangled and sample size is small-to-moderate
- Promising new type of nearest neighbors classifier; quite different from current competitors
- As the last topic, switch gears: classification $\rightarrow$ clustering
- Many clustering algorithms rely on a pairwise distance matrix

Wrapping up Classification & on to Distance estimation

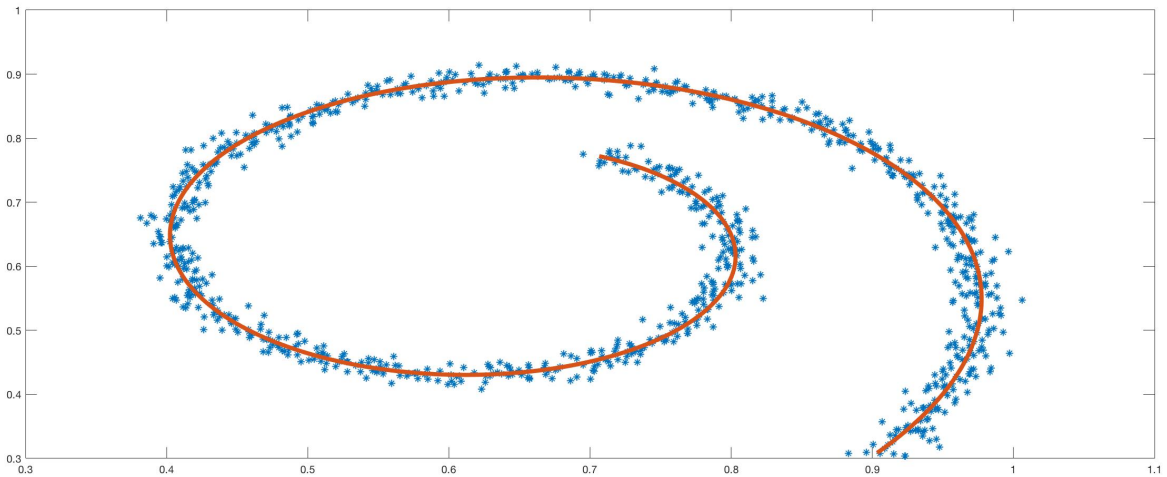
- *Labeled* data: LOMA and SPA work well when supports are entangled and sample size is small-to-moderate
- Promising new type of nearest neighbors classifier; quite different from current competitors
- As the last topic, switch gears: classification $\rightarrow$ clustering
- Many clustering algorithms rely on a pairwise distance matrix
- Choice of distance between data points plays a critical role in many statistical procedures

Wrapping up Classification & on to Distance estimation

- *Labeled* data: LOMA and SPA work well when supports are entangled and sample size is small-to-moderate
- Promising new type of nearest neighbors classifier; quite different from current competitors
- As the last topic, switch gears: classification $\rightarrow$ clustering
- Many clustering algorithms rely on a pairwise distance matrix
- Choice of distance between data points plays a critical role in many statistical procedures
- When support is non-linear, Euclidean $\rightarrow$ GEODESIC distance

Wrapping up Classification & on to Distance estimation

- *Labeled* data: LOMA and SPA work well when supports are entangled and sample size is small-to-moderate
- Promising new type of nearest neighbors classifier; quite different from current competitors
- As the last topic, switch gears: classification $\rightarrow$ clustering
- Many clustering algorithms rely on a pairwise distance matrix
- Choice of distance between data points plays a critical role in many statistical procedures
- When support is non-linear, Euclidean $\rightarrow$ GEODESIC distance



Geodesic distance

- Geodesic distance: $d_M(x, y)$ = length of shortest path on manifold connecting x and y

Geodesic distance

- Geodesic distance: $d_M(x, y)$ = length of shortest path on manifold connecting x and y
- Geodesic distance is only computable for very simple known manifolds (planes, spheres)

Geodesic distance

- Geodesic distance: $d_M(x, y)$ = length of shortest path on manifold connecting x and y
- Geodesic distance is only computable for very simple known manifolds (planes, spheres)
- Current algorithms combine local (within neighborhoods) & global estimation

Geodesic distance

- Geodesic distance: $d_M(x, y)$ = length of shortest path on manifold connecting x and y
- Geodesic distance is only computable for very simple known manifolds (planes, spheres)
- Current algorithms combine local (within neighborhoods) & global estimation
- Euclidean used within neighborhoods

Geodesic distance

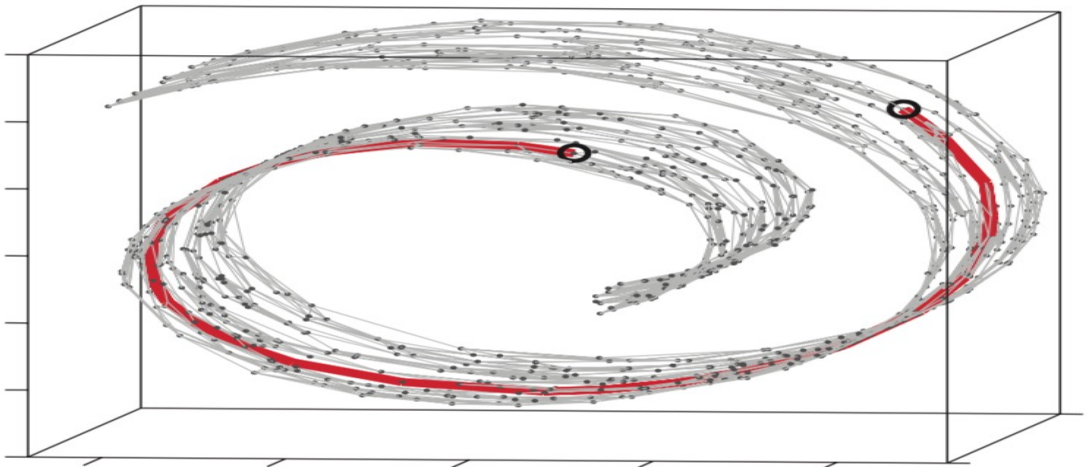
- Geodesic distance: $d_M(x, y)$ = length of shortest path on manifold connecting x and y
- Geodesic distance is only computable for very simple known manifolds (planes, spheres)
- Current algorithms combine local (within neighborhoods) & global estimation
- Euclidean used within neighborhoods
- Graph distance is then used to connect the neighborhoods

Geodesic distance

- Geodesic distance: $d_M(x, y)$ = length of shortest path on manifold connecting x and y
- Geodesic distance is only computable for very simple known manifolds (planes, spheres)
- Current algorithms combine local (within neighborhoods) & global estimation
- Euclidean used within neighborhoods
- Graph distance is then used to connect the neighborhoods
- Our contribution: replace locally Euclidean w/ locally spherical

Geodesic distance

- Geodesic distance: $d_M(x, y)$ = length of shortest path on manifold connecting x and y
- Geodesic distance is only computable for very simple known manifolds (planes, spheres)
- Current algorithms combine local (within neighborhoods) & global estimation
- Euclidean used within neighborhoods
- Graph distance is then used to connect the neighborhoods
- Our contribution: replace locally Euclidean w/ locally spherical

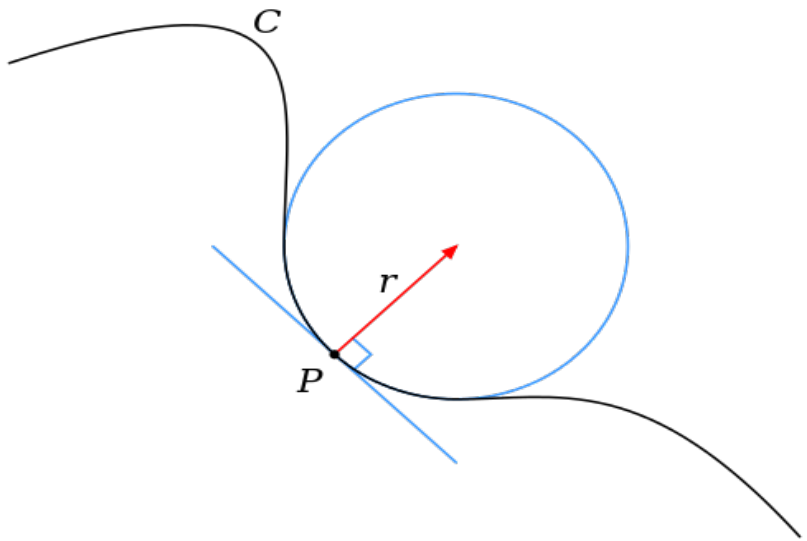


Spherical distance

- On a sphere centered at c with radius r, geodesic distance: $d_S(x, y) = r \arccos\left(\frac{(x-c)^\top (y-c)}{r^2}\right)$.

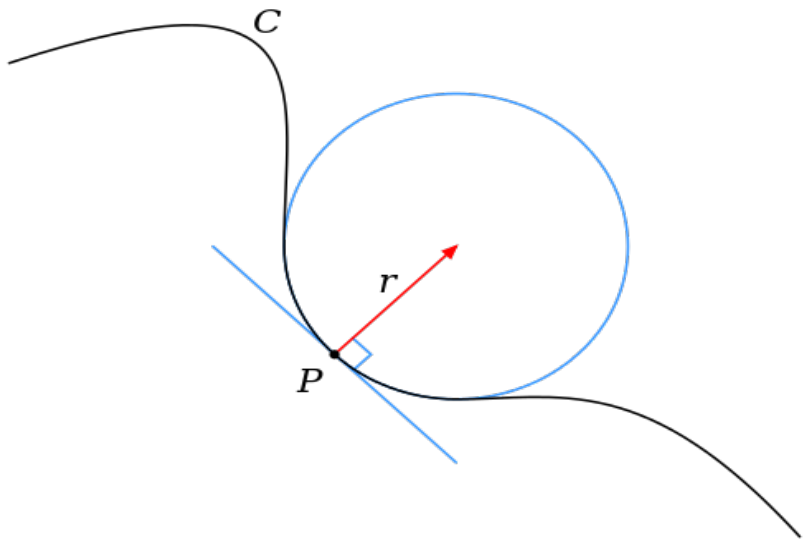
Spherical distance

- On a sphere centered at c with radius r, geodesic distance: $d_S(x, y) = r \arccos\left(\frac{(x-c)^\top (y-c)}{r^2}\right)$.



Spherical distance

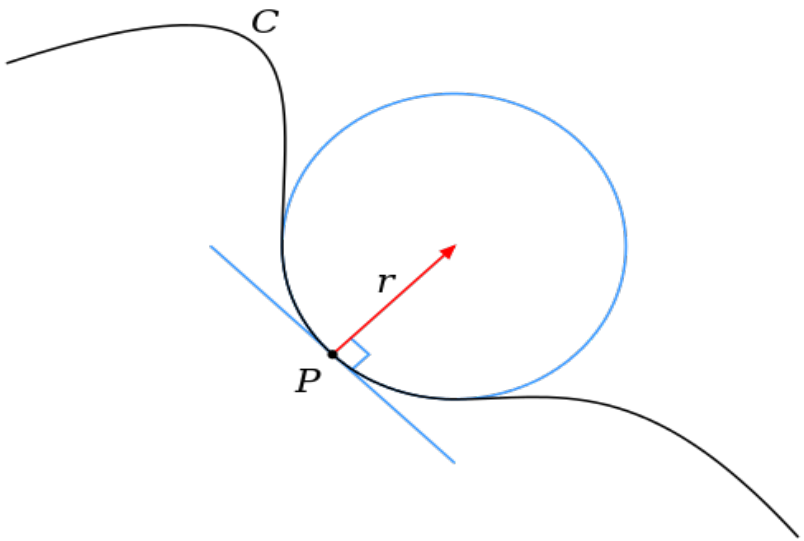
- On a sphere centered at c with radius r, geodesic distance: $d_S(x, y) = r \arccos\left(\frac{(x-c)^\top (y-c)}{r^2}\right)$.



- Let $\hat{x}, \hat{y} \in S(c, r)$ be the projection of $x, y \in M$ to the "best" sphere $S(c, r)$ and estimate $d_M(x, y)$ by the spherical distance $d_S(\hat{x}, \hat{y})$.

Spherical distance

- On a sphere centered at c with radius r, geodesic distance: $d_S(x, y) = r \arccos\left(\frac{(x-c)^\top (y-c)}{r^2}\right)$.



- Let $\widehat{x}, \widehat{y} \in S(c, r)$ be the projection of $x, y \in M$ to the "best" sphere $S(c, r)$ and estimate $d_M(x, y)$ by the spherical distance $d_S(\widehat{x}, \widehat{y})$.
- The error is in general smaller than the Euclidean estimation.

Local error

Theorem (Li, D., 19')

Let γ be a curve and $d_M(x, y) = s$ for $x, y \in \gamma$. Then

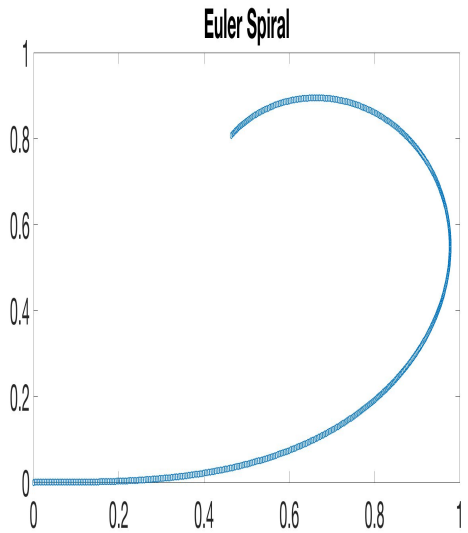
$$\|x - y\| = s + O(s^3), \quad d_S(x, y) = s + O(s^4).$$

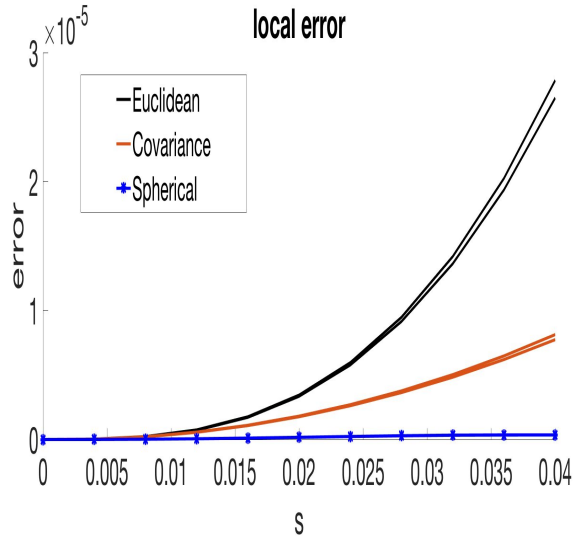
Local error

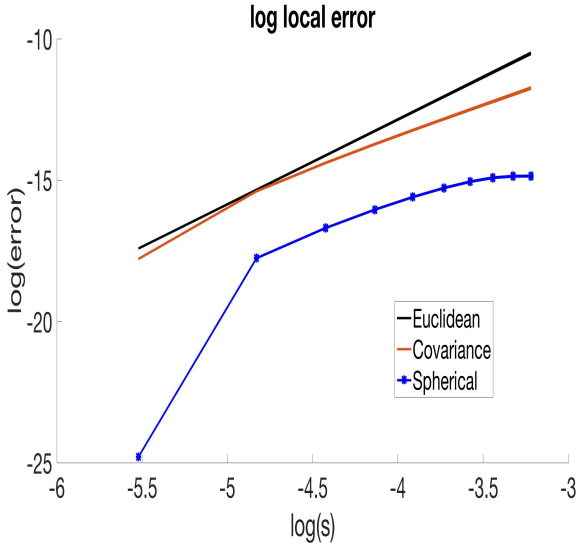
Theorem (Li, D., 19')

Let γ be a curve and $d_M(x, y) = s$ for $x, y \in \gamma$. Then

$$\|x - y\| = s + O(s^3), \quad d_S(x, y) = s + O(s^4).$$





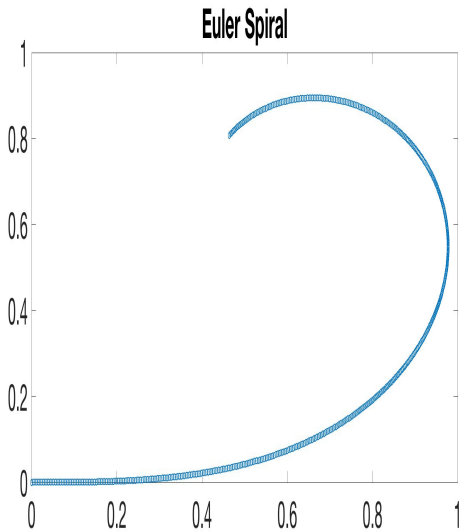


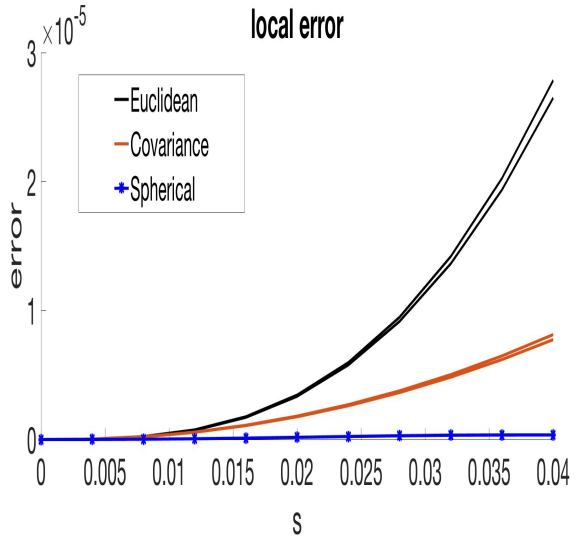
Local error

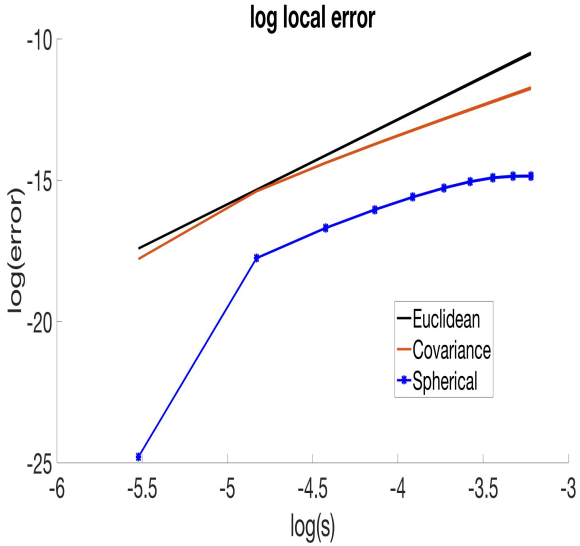
Theorem (Li, D., 19')

Let γ be a curve and $d_M(x, y) = s$ for $x, y \in \gamma$. Then

$$\|x - y\| = s + O(s^3), \quad d_S(x, y) = s + O(s^4).$$







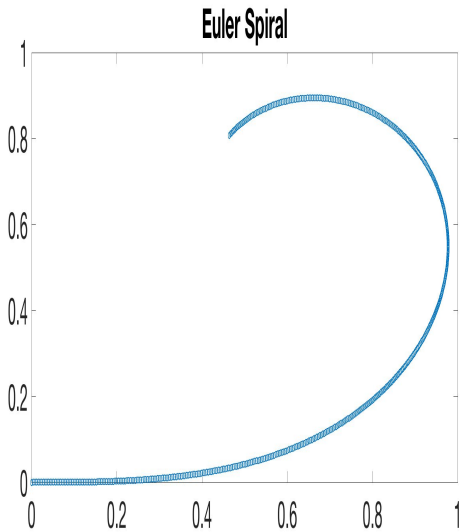
- Spherical distance has better error rate.

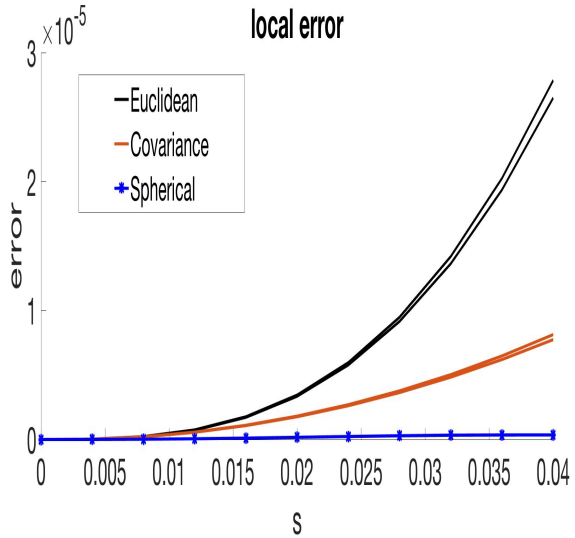
Local error

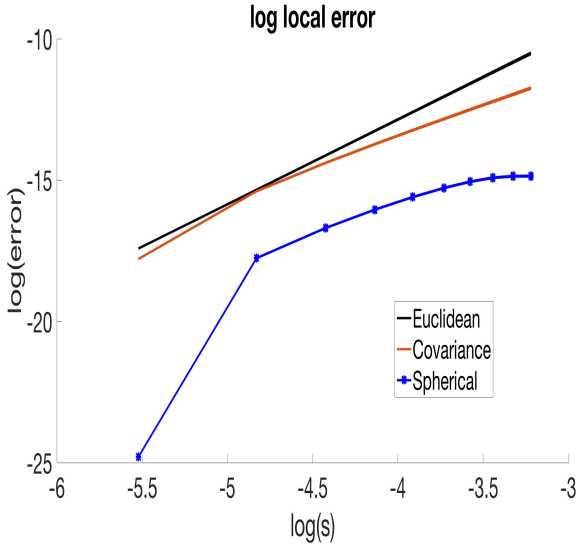
Theorem (Li, D., 19')

Let γ be a curve and $d_M(x, y) = s$ for $x, y \in \gamma$. Then

$$\|x - y\| = s + O(s^3), \quad d_S(x, y) = s + O(s^4).$$







- Spherical distance has better error rate.
- Simulation results match our theory.

Clustering

- k-medoids: minimize distance between points in each group & group centers.

Clustering

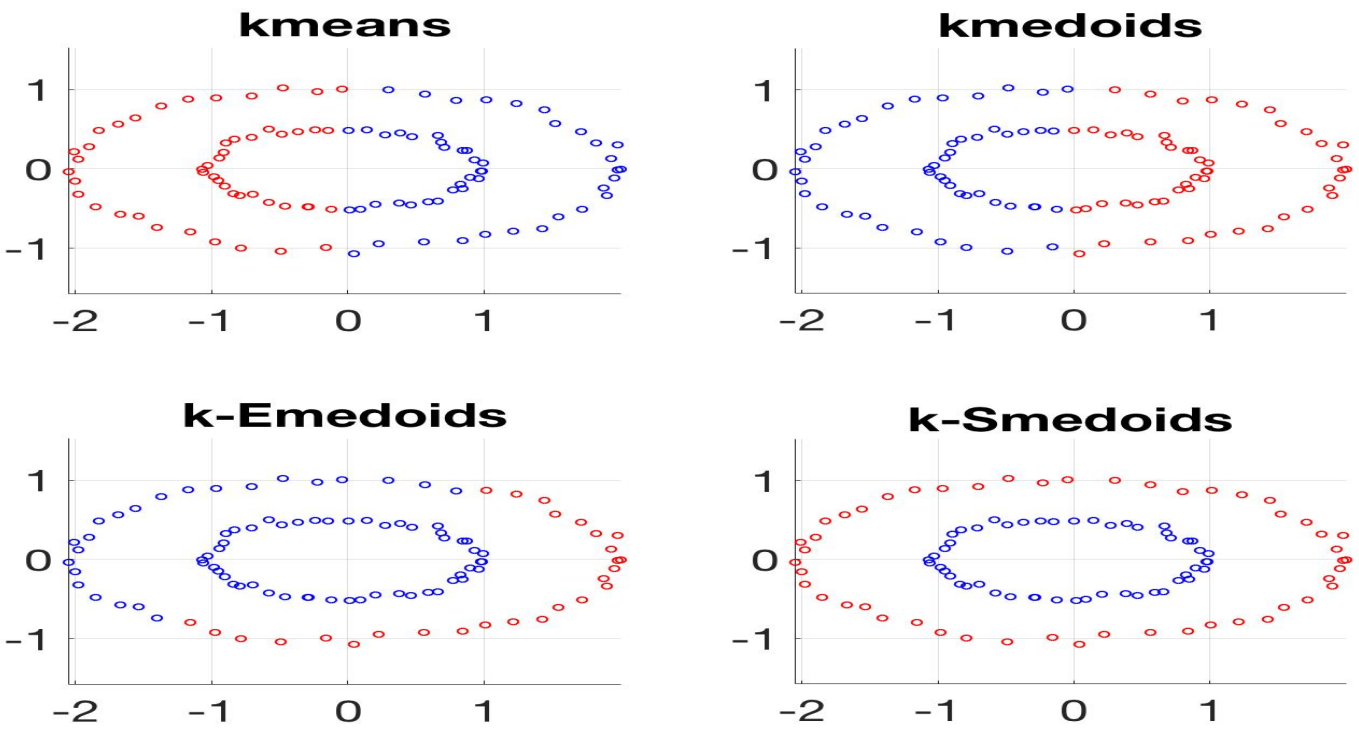
- k-medoids: minimize distance between points in each group & group centers.
- Input is pairwise distance matrix between data points.

Clustering

- k-medoids: minimize distance between points in each group & group centers.
- Input is pairwise distance matrix between data points.
- Some choices: global Euclidean (k-medoids), graph Euclidean(k-Emedoids) and graph spherical (k-Smedoids).

Clustering

- *k*-medoids: minimize distance between points in each group & group centers.
- Input is pairwise distance matrix between data points.
- Some choices: global Euclidean (*k*-medoids), graph Euclidean(*k*-Emedoids) and graph spherical (*k*-Smedoids).



Banknote

- Banknote dataset. $n = 1372$, $p = 4$, $d = 1$, $K = 2$.

Banknote

- Banknote dataset. $n = 1372$, $p = 4$, $d = 1$, $K = 2$.
- Six measurements of the clustering performance: Adjusted Rand Index (ARI), Mutual Information Based Scores (MIBS), HOMogeneity (HOM), COMpleteness (COM), V-Measure (VM) and Fowlkes-Mallows Scores (FMS): the larger the better.

Banknote

- Banknote dataset. $n = 1372$, $p = 4$, $d = 1$, $K = 2$.
- Six measurements of the clustering performance: Adjusted Rand Index (ARI), Mutual Information Based Scores (MIBS), HOMogeneity (HOM), COMpleteness (COM), V-Measure (VM) and Fowlkes-Mallows Scores (FMS): the larger the better.

Table: Clustering performance for Galaxy Zoo

	k-medoids	k-Emedoids	k-Smedoids
ARI	0.744	0.805	**0.954**
MIBS	0.6402	0.702	**0.900**
HOM	0.712	0.763	**0.919**
COM	0.641	0.702	**0.900**
VM	0.674	0.731	**0.909**
FMS	0.899	0.923	**0.983**

Galaxy Zoo

- Galaxy Zoo dataset. $n = 1000$, $p = 6$, $d = 1$, $K = 2$.

Galaxy Zoo

- Galaxy Zoo dataset. $n = 1000$, $p = 6$, $d = 1$, $K = 2$.
- Two groups: spiral galaxy and elliptical galaxy.

Galaxy Zoo

- Galaxy Zoo dataset. $n = 1000$, $p = 6$, $d = 1$, $K = 2$.
- Two groups: spiral galaxy and elliptical galaxy.

Table: Clustering performance for Banknote

	k-medoids	k-Emedoids	k-Smedoids
ARI	0.059	0.004	**0.452**
MIBS	0.041	0.008	**0.439**
HOM	0.0416	0.009	**0.439**
COM	0.041	0.138	**0.508**
VM	0.0415	0.0163	**0.471**
FMS	0.533	0.707	**0.754**

Discussion

- Over-arching theme: use spheres for local approximation instead of planes

Discussion

- Over-arching theme: use spheres for local approximation instead of planes
- SPCA – PCA

Discussion

- Over-arching theme: use spheres for local approximation instead of planes
- SPCA – PCA
- Fisher Gaussian kernel – Gaussian kernel

Discussion

- Over-arching theme: use spheres for local approximation instead of planes
- SPCA – PCA
- Fisher Gaussian kernel – Gaussian kernel
- Spherical distance – Euclidean distance

Discussion

- Over-arching theme: use spheres for local approximation instead of planes
- SPCA – PCA
- Fisher Gaussian kernel – Gaussian kernel
- Spherical distance – Euclidean distance
- Many applications: manifold learning, denoising, visualization, density estimation, classification, geodesic distance estimation, clustering.

Discussion

- Over-arching theme: use spheres for local approximation instead of planes
- SPCA – PCA
- Fisher Gaussian kernel – Gaussian kernel
- Spherical distance – Euclidean distance
- Many applications: manifold learning, denoising, visualization, density estimation, classification, geodesic distance estimation, clustering.
- Interesting future areas:
  - To further generalize the spherelets to ellipsoid or quadratic surface.

Discussion

- Over-arching theme: use spheres for local approximation instead of planes
- SPCA – PCA
- Fisher Gaussian kernel – Gaussian kernel
- Spherical distance – Euclidean distance
- Many applications: manifold learning, denoising, visualization, density estimation, classification, geodesic distance estimation, clustering.
- Interesting future areas:
  - To further generalize the spherelets to ellipsoid or quadratic surface.
  - To scale up to large p and large n case, like sparse SPCA.

Discussion

- Over-arching theme: use spheres for local approximation instead of planes
- SPCA – PCA
- Fisher Gaussian kernel – Gaussian kernel
- Spherical distance – Euclidean distance
- Many applications: manifold learning, denoising, visualization, density estimation, classification, geodesic distance estimation, clustering.
- Interesting future areas:
  - To further generalize the spherelets to ellipsoid or quadratic surface.
  - To scale up to large p and large n case, like sparse SPCA.
  - To address data not in $\mathbb{R}^p$, like categorical data or functional data.

Key References

- D. Li, M. Mukhopadhyay and D. Dunson, Efficient Manifold and Subspace Approximations with Spherelets, arXiv: 1706.08263, 2018.
- M. Mukhopadhyay, D. Li and D. Dunson, Estimating densities with nonlinear support using Fisher-Gaussian kernels, arXiv:1907.05918, 2019.
- D. Li and D. Dunson, Classification via local manifold approximation, arXiv:1903.00985, 2019.
- D. Li and D. Dunson, Geodesic Distance Estimation with Spherelets, arXiv:1907.00296, 2019.

Brilliant collaborators who did everything important