A tutorial on Manifold Learning for real data
The Fields Institute Workshop on Manifold and Graph-based learning

Marina Meilă

Department of Statistics
University of Washington

19-20 May, 2022
Outline

1. What is manifold learning good for?

2. Manifolds, Coordinate Charts and Smooth Embeddings

3. Non-linear dimension reduction algorithms
   - Local PCA
   - PCA, Kernel PCA, MDS recap
   - Principal Curves and Surfaces (PCS)
   - Embedding algorithms
   - Heuristic algorithms

4. Metric preserving manifold learning – Riemannian manifolds basics
   - Embedding algorithms introduce distortions
   - Metric Manifold Learning – Intuition
   - Estimating the Riemannian metric

5. Neighborhood radius and other choices
   - What graph? Radius-neighbors vs. k nearest-neighbors
   - What neighborhood radius/kernel bandwidth?
Outline

1 What is manifold learning good for?

2 Manifolds, Coordinate Charts and Smooth Embeddings

3 Non-linear dimension reduction algorithms
   - Local PCA
   - PCA, Kernel PCA, MDS recap
   - Principal Curves and Surfaces (PCS)
   - Embedding algorithms
   - Heuristic algorithms

4 Metric preserving manifold learning – Riemannian manifolds basics
   - Embedding algorithms introduce distortions
   - Metric Manifold Learning – Intuition
   - Estimating the Riemannian metric

5 Neighborhood radius and other choices
   - What graph? Radius-neighbors vs. k nearest-neighbors
   - What neighborhood radius/kernel bandwidth?
What is manifold learning good for?

- Principal Component Analysis (PCA). What is it good for? 
  - High $\rightarrow$ low dim (save space, reduction processing time).
  - Understand $\rightarrow$ more "relevant" features.
What is manifold learning good for?

Spectra of galaxies measured by the Sloan Digital Sky Survey (SDSS)

- Preprocessed by Jacob VanderPlas and Grace Telford
- $n = 675,000$ spectra $\times D = 3750$ dimensions

embedding by James McQueen
What is manifold learning good for?

Molecular configurations

- Data from Molecular Dynamics (MD) simulations of small molecules by [Chmiela et al. 2016]
- \(n \approx 200,000\) configurations \(\times D \sim 20 - 60\) dimensions
What is manifold learning good for?

When to do (non-linear) dimension reduction

- $n = 698$ gray images of faces in
  $D = 64 \times 64$ dimensions
- head moves up/down and right/left
- With only two degrees of freedom, the faces define a 2D manifold in the space of all $64 \times 64$ gray images
Outline

1. What is manifold learning good for?

2. Manifolds, Coordinate Charts and Smooth Embeddings

3. Non-linear dimension reduction algorithms
   - Local PCA
   - PCA, Kernel PCA, MDS recap
   - Principal Curves and Surfaces (PCS)
   - Embedding algorithms
   - Heuristic algorithms

4. Metric preserving manifold learning – Riemannian manifolds basics
   - Embedding algorithms introduce distortions
   - Metric Manifold Learning – Intuition
   - Estimating the Riemannian metric

5. Neighborhood radius and other choices
   - What graph? Radius-neighbors vs. k nearest-neighbors
   - What neighborhood radius/kernel bandwidth?
Manifold. Basic definitions

- **manifold**
  \[ \mathcal{M} = \text{set that can locally be } \cong \mathbb{R}^d \]
  \[ \mathcal{M} \xrightarrow{\text{chart}} V \subset \mathbb{R}^d \]

- **chart**
  \[ U \xrightarrow{x} V \subset \mathbb{R}^d \]

- **atlas**
  \[ \{ \text{all charts } x \} \]

- **d** is called intrinsic dimension of \( \mathcal{M} \)

- If the original data \( p \in \mathbb{R}^D \), call \( D \) the ambient dimension.

- \( x(p) \in \mathbb{R}^d \)
- \( x^{-1} \) exists, differentiable
Intrinsic dimension. Tangent subspace

For $p \in \mathcal{M}$,

$T_p \mathcal{M} \cong \mathbb{R}^d$ vector space

$T\mathcal{M} = \bigcup T_p \mathcal{M}$, $p \in \mathcal{M}$

$tangent$ $bundle$

$T_p \mathcal{M} = \{ tangents \ to \ curves \ in \ \mathcal{M} \}$

$\left\{ \frac{\partial \mathbf{x}}{\partial u}(p), \frac{\partial \mathbf{x}}{\partial v}(p) \right\} \in T_p \mathcal{M}$

basis

$x(p) \in \mathbb{R}^d$

$x = \begin{bmatrix} u \\ v \end{bmatrix}$
Embeddings

- One can circumvent using multiple charts by mapping the data into \( m > d \) dimensions.
- Let \( \phi : \mathcal{M} \rightarrow \mathbb{R}^m \) be a smooth function, and let \( \mathcal{N} = \phi(\mathcal{M}) \).
- \( \phi \) is an embedding if the inverse \( \phi^{-1} : \mathcal{N} \rightarrow \mathcal{M} \) exists and is differentiable (a diffeomorphism).

\[
\text{data } \mathbb{R}^d \xrightarrow{\phi} \mathbb{R}^m
\]

- Whitney’s Embedding Theorem (?) states that any \( d \)-dimensional smooth manifold can be embedded into \( \mathbb{R}^{2d} \).
- Hence, if \( d \ll D \), very significant dimension reductions can be achieved with a single map \( \phi : \mathcal{M} \rightarrow \mathbb{R}^m \).
- Manifold learning algorithms aim to construct maps \( \phi \) like the above from finite data sampled from \( \mathcal{M} \).
Examples of manifolds and coordinate charts

- $\mathbb{R}^d$: $\dim \mathbb{R}^d = d$
- $S^1$: $\dim S^1 = 1$
- $S^d$: sphere of $\dim = d$
- $S^d$ embedded in $\mathbb{R}^{d+1}$, $m \geq d+1$
- $T^2$: torus
- $\dim T^2 = d$
- $m = d+1$
- generated by $2$ circles
- subset of $\mathbb{R}^d$ mapped in $\mathbb{R}^d$
Examples of manifolds and coordinate charts

\[ \xi \in \mathbb{R}^d \quad S^d = \{ \xi \in \mathbb{R}^d : \|\xi\| = 1 \} \]
Examples of manifolds and coordinate charts

**Not manifolds**
- dimension not constant
- unions of manifolds that intersect
- sharp corners (non-smooth)
- many/most neural network embeddings
- manifolds can have border
Outline

1. What is manifold learning good for?

2. Manifolds, Coordinate Charts and Smooth Embeddings

3. Non-linear dimension reduction algorithms
   - Local PCA
   - PCA, Kernel PCA, MDS recap
   - Principal Curves and Surfaces (PCS)
   - Embedding algorithms
   - Heuristic algorithms

4. Metric preserving manifold learning – Riemannian manifolds basics
   - Embedding algorithms introduce distortions
   - Metric Manifold Learning – Intuition
   - Estimating the Riemannian metric

5. Neighborhood radius and other choices
   - What graph? Radius-neighbors vs. k nearest-neighbors
   - What neighborhood radius/kernel bandwidth?
Non-linear dimension reduction: Three principles

Algorithm given \( D = \{ \xi_1, \ldots, \xi_n \} \) from \( M \subset \mathbb{R}^D \), map them by Algorithm \( f \) to \( \{ y_1, \ldots, y_n \} \subset \mathbb{R}^m \).

Assumption if points from \( M \), \( n \to \infty \), \( f \) is embedding of \( M \) (\( f \) “recovers” \( M \) of arbitrary shape).

1. Local (weighted) PCA (lPCA)
2. Principal Curves and Surfaces (PCS)
3. Embedding algorithms (Diffusion Maps/Laplacian Eigenmaps, Isomap, LTSA, MVU, Hessian Eigenmaps,...)
4. [Other, heuristic] t-SNE, UMAP, LLE

What makes the problem hard?

- Intrinsic dimension \( d \)
  - must be estimated (we assume we know it)
  - sample complexity is exponential in \( d \) – NONPARAMETRIC

- non-uniform sampling
- volume of \( M \) (we assume volume finite; larger volume requires more samples)
- injectivity radius/reach of \( M \)
- curvature

- ESSENTIAL smoothness parameter: the neighborhood radius
Non-linear dimension reduction: Three principles

Algorithm given \( D = \{ \xi_1, \ldots, \xi_n \} \) from \( \mathcal{M} \subset \mathbb{R}^D \), map them by Algorithm \( f \) to \( \{ y_1, \ldots, y_n \} \subset \mathbb{R}^m \).

Assumption if points from \( \mathcal{M} \), \( n \to \infty \), \( f \) is embedding of \( \mathcal{M} \) (\( f \) “recovers” \( \mathcal{M} \) of arbitrary shape).

1. Local (weighted) PCA (LPCA)
2. Principal Curves and Surfaces (PCS)
3. Embedding algorithms (Diffusion Maps/Laplacian Eigenmaps, Isomap, LTSA, MVU, Hessian Eigenmaps, . . .)
4. [Other, heuristic] t-SNE, UMAP, LLE

What makes the problem hard?

- Intrinsic dimension \( d \)
  - must be estimated (we assume we know it)
  - sample complexity is exponential in \( d \) – **NONPARAMETRIC**

- **non-uniform sampling**
  - volume of \( \mathcal{M} \) (we assume volume finite; larger volume requires more samples)
  - injectivity radius/reach of \( \mathcal{M} = 2 \)
  - curvature

- **ESSENTIAL** smoothness parameter: the **neighborhood radius**

**Sampling distribution \( p(\cdot) \)**
Parametric vs. non-parametric

An example of density estimation with data $x_{1:n} \in \mathbb{R}$.

1. **Gaussian $N(\mu, \sigma^2)$ parametric.**
   - $\hat{\mu} = \frac{1}{n} \sum_{i=1}^{n} x_i$, $\hat{\sigma}^2 = \frac{1}{n-1} \sum_{i=1}^{n} (x_i - \hat{\mu})^2$
   - Error $\mu - \hat{\mu}$ has mean 0 and standard deviation $\sigma_{\hat{\mu}} = \frac{\sigma}{\sqrt{n}} \propto n^{-1/2}$
   - To increase accuracy $\times 10$, $n$ must increase $\times 10^2 = 100$

2. **Kernel density estimation (KDE), non-parametric**
   \[
p_h(x) = \frac{1}{n} \sum_{i=1}^{n} \frac{1}{h} \kappa \left( \frac{x_i - x}{h} \right)
   \]
   - $\kappa = N(0, 1)$ the kernel, $h > 0$ is the kernel width
   - Accuracy for KDE $\propto n^{-2/5}$
   - To increase accuracy $\times 10$, $n$ must increase $\times 10^{5/2} \approx 316$

<table>
<thead>
<tr>
<th>Model</th>
<th>e.g.</th>
<th>distribution shape</th>
<th>error rate</th>
<th>to decrease err. by 10 we need samples $\times$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Parametric</td>
<td>$N(\mu, \sigma^2)$</td>
<td>fixed</td>
<td>$n^{-1/2}$</td>
<td>$n \times 10^2$</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>100</td>
</tr>
<tr>
<td>Non-parametric</td>
<td>KDE in $\mathbb{R}$</td>
<td>any</td>
<td>$n^{-2/5}$</td>
<td>$n \times 10^{5/2}$</td>
</tr>
<tr>
<td></td>
<td>KDE in $\mathbb{R}^d$</td>
<td>any</td>
<td>$n^{-2/(d+4)}$</td>
<td>$n \times 10^{(d+4)/2}$</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1000 $(d = 2)$</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>3163 $(d = 3)$</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>10,000 $(d = 4)$</td>
</tr>
</tbody>
</table>
Neighborhood graphs

- All ML algorithms start with a **neighborhood graph** over the data points
  - $\text{neigh}_i$ denotes the neighbors of $\xi_i$, and $k_i = |\text{neigh}_i|$.
  - $\Xi_i = [\xi_{i' j}]_{i' \in \text{neigh}_i} \in \mathbb{R}^{D \times k_i}$ contains the coordinates of $\xi_i$’s neighbors
- In the **radius-neighbor** graph, the neighbors of $\xi_i$ are the points within distance $r$ from $\xi_i$, i.e. in the ball $B_r(\xi_i)$.
- In the **k-nearest-neighbor (k-nn)** graph, they are the $k$ nearest-neighbors of $\xi_i$.

k-nn graph has many computational advantages
- constant degree $k$ (or $k - 1$)
- connected for any $k > 1$
- more software available

- but much more difficult to use for consistent estimation of manifolds (see later, and )

Data $\xi_1, \ldots, \xi_n \subset \mathbb{R}^D$

neighborhood graph

$A$ (sparse) matrix of distances between neighbors