Nonlinear Dimensionality Reduction

Yunpeng Shi

October 9, 2017

Table of Contents

What is Dimension Reduction?

Dimension Reduction

Figure: High dimensional data usually have low dimensional structure

Assume that there are n data points $x_i \in \mathbb{R}^p$, we want to find a map $\Phi(x) \mapsto y$ such that $y \in \mathbb{R}^d$ where $d < p$ or even $d \ll p$. Namely, we are finding a map, either linear or nonlinear, that projects the high dimensional data points into lower dimensional one.

Principal Component Analysis (PCA)

- Seeks an optimal low dimensional vector space that gives smallest projection distance for the input data

Principal Component Analysis (PCA)

- Seeks an optimal low dimensional vector space that gives smallest projection distance for the input data
- The mapping is linear

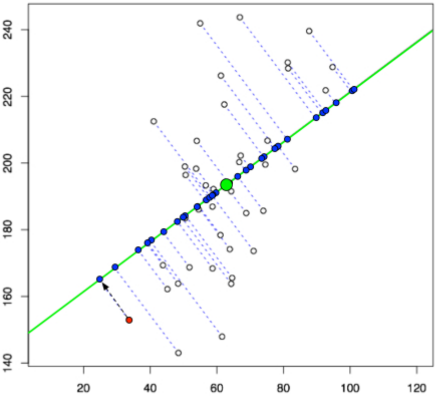
Principal Component Analysis (PCA)

- Seeks an optimal low dimensional vector space that gives smallest projection distance for the input data
- The mapping is linear

The problem is formulated as follows:
Given a set of high dimensional data $\{x_i\}_{i=1}^n$ where $x_i \in \mathbb{R}^p$, we want to find a $d-$dimensional linear subspace L such that:

$$L = \text{argmin}_L \sum_{i=1}^{n} dist^2(x_i, L) \tag{1}$$

PCA 2D case



PCA

1. Center the data set to origin.

PCA

1. Center the data set to origin.
2. Do eigenvalue decomposition for Gram matrix $X^T X$ where $X = [x_1, x_2, ..., x_n]$
   Namely, $K = X^T X = U \Lambda U^T$ where U is orthogonal matrix.
   Equivalently U can also be obtained by directly applying SVD to X.

PCA

1. Center the data set to origin.
2. Do eigenvalue decomposition for Gram matrix $X^T X$ where $X = [x_1, x_2, ..., x_n]$
   Namely, $K = X^T X = U \Lambda U^T$ where U is orthogonal matrix.
   Equivalently U can also be obtained by directly applying SVD to X.
3. Let $U = [U_1, U_2, ...U_n]$. then $L = \text{span}\{U_1, U_2..., U_d\}$

PCA

1. Center the data set to origin.
2. Do eigenvalue decomposition for Gram matrix $X^T X$ where $X = [x_1, x_2, ..., x_n]$
   Namely, $K = X^T X = U \Lambda U^T$ where U is orthogonal matrix.
   Equivalently U can also be obtained by directly applying SVD to X.
3. Let $U = [U_1, U_2, ...U_n]$. then $L = \text{span}\{U_1, U_2..., U_d\}$
4. Project the centered data to L, we have $y_i = [U_1, U_2..., U_d]^T x_i$.

PCA

1. Center the data set to origin.
2. Do eigenvalue decomposition for Gram matrix $X^T X$ where $X = [x_1, x_2, ..., x_n]$
   Namely, $K = X^T X = U \Lambda U^T$ where U is orthogonal matrix.
   Equivalently U can also be obtained by directly applying SVD to X.
3. Let $U = [U_1, U_2, ...U_n]$. then $L = \text{span}\{U_1, U_2 ..., U_d\}$
4. Project the centered data to L, we have $y_i = [U_1, U_2 ..., U_d]^T x_i$.

PCA

Good-of-Fit Measure

$$\frac{\sum_{i=1}^{d} \lambda_i}{\sum_{i=1}^{n} \lambda_i}$$

Nonlinear Dimension Reduction and Manifold Learning

What if data structure is intrinsically nonlinear?

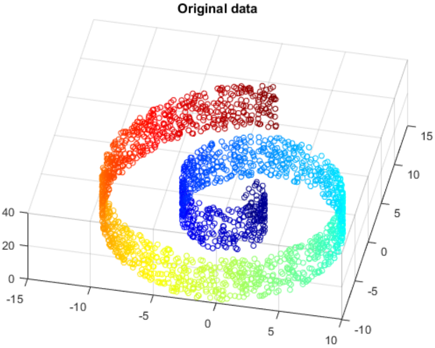
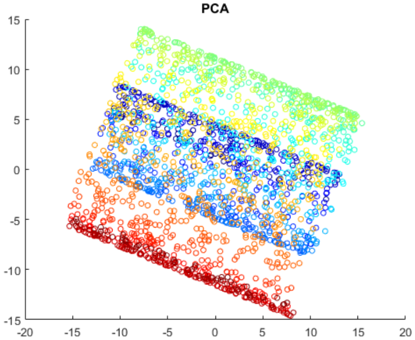


Figure: swiss roll data

PCA does not work well in nonlinear case

What if data structure is intrinsically nonlinear?



Capture nonlinearity

Kernel matrix

Each element of kernel matrix can be viewed as inner product in feature space. namely, $K(x, x') = \langle \Phi(x) \, , \Phi(x') \rangle$, where $\Phi(\cdot)$ is a mapping to feature space. $K(x, x')$ can be viewed as similarity between x and x'. Usually, we use kernel:

1. linear kernel $k(x, x') = \langle x \, , x' \rangle$
2. polynomial kernel $k(x, x') = (\langle x \, , x' \rangle + c)^d$
3. Gaussian kernel $k(x, x') = e^{-\frac{\|x - x'\|^2}{2\sigma^2}}$

Capture nonlinearity

Distance Matrix

For graph-based methods, usually we use pairwise distance information to obtain adjacency matrix. The linear kernel matrix (gram matrix $X^T X$) can also be obtained from Euclidean distance matrix by using double centering [4]:

$$K = -\frac{1}{2} H D^{(2)} H$$

where $D^{(2)}$ is the matrix of distance square, and H is centering matrix $I - \frac{1}{n}\mathbf{1}\mathbf{1}^T$.

$$k_{ij} = -\frac{1}{2}\left(d_{ij}^2 - \frac{1}{n}\sum_l d_{il}^2 - \frac{1}{n}\sum_l d_{jl}^2 + \frac{1}{n^2}\sum_{lm} d_{lm}^2\right).$$

Conversely, we can also get D from Gram matrix K.

Classical Scaling

$$\phi(\mathbf{Y}) = \sum_{ij} \left(d_{ij}^2 - \|\mathbf{y}_i - \mathbf{y}_j\|^2 \right)$$

$\mathbf{y}_i$ is restricted to be $\mathbf{x}_i \mathbf{M}$, and $\|\mathbf{m}_j\|^2 = 1$ for $\forall j$.

Isomap

- Define Adjacency Matrix from Distance.

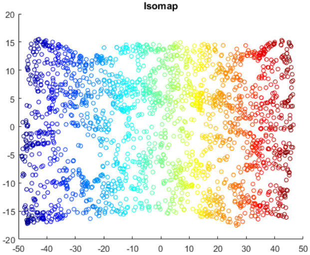
Isomap

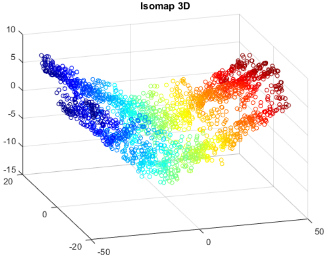
- Define Adjacency Matrix from Distance.
- Compute geodesic distance between two nodes.

Isomap

- Define Adjacency Matrix from Distance.
- Compute geodesic distance between two nodes.
- Implement MDS [4] to get embedded location.

Isomap


Isomap
20
15
10
5
0
-5
-10
-15
-20
-50
-40
-30
-20
-10
0
10
20
30
40
50


Isomap 3D
10
5
0
-5
-10
-15
20
0
-20
-50
0
50

Isomap

Weakness

- short-circuiting
- suffer from 'holes' in the manifold
- suffern from nonconvex manifold

Applications

- wood inspection
- visualization of biomedical data
- head pose estimation

Kernel PCA

$$\mathbf{y}_i = \left\{ \sum_{j=1}^{n} a_1^{(j)} \kappa(\mathbf{x}_j, \mathbf{x}_i), \ldots, \sum_{j=1}^{n} a_d^{(j)} \kappa(\mathbf{x}_j, \mathbf{x}_i) \right\}$$

Weakness: focus too much on global distances
Applications: face recognition, speech recognition, and novelty detection

MVU

Maximum Variance Unfolding (MVU) [4]

Assume that the data are centered at the origin. The Gram matrix $X^T X$. Then MVU learn the kernel matrix K in the following way:

Maximize $\sum_{ij} \|\vec{y}_i - \vec{y}_j\|^2$ **subject to:**
(1) $\|\vec{y}_i - \vec{y}_j\|^2 = \|\vec{x}_i - \vec{x}_j\|^2$ **for all** (i,j) **with** $\eta_{ij} = 1$.
(2) $\sum_i \vec{y}_i = 0$

Maximize trace(K) **subject to:**
(1) $K_{ii} - 2K_{ij} + K_{jj} = \|\vec{x}_i - \vec{x}_j\|^2$ **for all** (i,j) **with** $\eta_{ij} = 1$.
(2) $\Sigma_{ij} K_{ij} = 0$.
(3) $K \succeq 0$.

MVU

- SDP problem.

MVU

- SDP problem.
- Preserve local distance.

MVU

- SDP problem.
- Preserve local distance.
- Maximize global variance (unfolding).

Weakness: short-circuiting
Applications: sensor localization, DNA microarray data analysis

Diffusion maps

- contruct graph of data using Gaussian kernel
- use the weights as transition probability to form a Markov chain
- Choose eigenvectors of transition matrix as low dimensional representation of data

Diffusion maps

$\Pr(X_{t+1} = j | X_t = i) = M_{ij} = \frac{w_{ij}}{\sum_i w_{ij}}$, or $M = D^{-1}W$

From SVD,

$$M = \Phi \Lambda \Psi^T$$

(Diffusion Map) *Given a graph $G = (V, E, W)$ construct M and its decomposition described above. The Diffusion Map is a map $\phi_t : V \to \mathbb{R}^{n-1}$ given by*

$$\phi_t(v_i) = \begin{bmatrix} \lambda_2^t \varphi_2(i) \\ \lambda_3^t \varphi_3(i) \\ \vdots \\ \lambda_n^t \varphi_n(i) \end{bmatrix}.$$

Diffusion maps

Truncated diffusion maps

$$\phi_t^{(d)}(v_i) = \begin{bmatrix} \lambda_2^t \varphi_2(i) \\ \lambda_3^t \varphi_3(i) \\ \vdots \\ \lambda_{d+1}^t \varphi_{d+1}(i) \end{bmatrix}.$$

Diffusion maps

$$\left\|\phi_t\left(v_{i_1}\right)-\phi_t\left(v_{i_2}\right)\right\|^2=\sum_{j=1}^n \frac{1}{\deg(j)}\left[\operatorname{Prob}\left\{X(t)=j|X(0)=i_1\right\}-\operatorname{Prob}\left\{X(t)=j|X(0)=i_2\right\}\right]^2 .$$

Proof.
Note that $\sum_{j=1}^n \frac{1}{\deg(j)}\left[\operatorname{Prob}\left\{X(t)=j|X(0)=i_1\right\}-\operatorname{Prob}\left\{X(t)=j|X(0)=i_2\right\}\right]^2$ can be rewritten as

$$\sum_{j=1}^n \frac{1}{\deg(j)}\left[\sum_{k=1}^n \lambda_k^t \varphi_k(i_1)\psi_k(j)-\sum_{k=1}^n \lambda_k^t \varphi_k(i_2)\psi_k(j)\right]^2=\sum_{j=1}^n \frac{1}{\deg(j)}\left[\sum_{k=1}^n \lambda_k^t\left(\varphi_k(i_1)-\varphi_k(i_2)\right)\psi_k(j)\right]^2$$

and

$$\begin{aligned}\sum_{j=1}^n \frac{1}{\deg(j)}\left[\sum_{k=1}^n \lambda_k^t\left(\varphi_k(i_1)-\varphi_k(i_2)\right)\psi_k(j)\right]^2 &= \sum_{j=1}^n\left[\sum_{k=1}^n \lambda_k^t\left(\varphi_k(i_1)-\varphi_k(i_2)\right)\frac{\psi_k(j)}{\sqrt{\deg(j)}}\right]^2 \\ &= \left\|\sum_{k=1}^n \lambda_k^t\left(\varphi_k(i_1)-\varphi_k(i_2)\right)D^{-\frac{1}{2}}\psi_k\right\|^2 .\end{aligned}$$

Diffusion maps

Note that $D^{-\frac{1}{2}}\psi_k = v_k$ which forms an orthonormal basis, meaning that

$$\begin{aligned} \left\| \sum_{k=1}^{n} \lambda_k^t \left(\varphi_k(i_1) - \varphi_k(i_2)\right) D^{-\frac{1}{2}}\psi_k \right\|^2 &= \sum_{k=1}^{n} \left(\lambda_k^t \left(\varphi_k(i_1) - \varphi_k(i_2)\right)\right)^2 \\ &= \sum_{k=2}^{n} \left(\lambda_k^t \varphi_k(i_1) - \lambda_k^t \varphi_k(i_2)\right)^2, \end{aligned}$$

where the last inequality follows from the fact that $\varphi_1 = \mathbf{1}$ and concludes the proof of the theorem.

□

t-distributed stochastic neighbor embedding (t-SNE)

Define pairwise probabilities

$$p_{j|i} = \frac{\exp(-\|\mathbf{x}_i - \mathbf{x}_j\|^2/2\sigma_i^2)}{\sum_{k\neq i}\exp(-\|\mathbf{x}_i - \mathbf{x}_k\|^2/2\sigma_i^2)}$$

$$p_{ij} = \frac{p_{j|i} + p_{i|j}}{2N}$$

Pairwise probabilties in target space

$$q_{ij} = \frac{(1 + \|\mathbf{y}_i - \mathbf{y}_j\|^2)^{-1}}{\sum_{k\neq m}(1 + \|\mathbf{y}_k - \mathbf{y}_m\|^2)^{-1}}$$

minimize Kullback-Leibler divergence

$$KL(P||Q) = \sum_{i\neq j} p_{ij} \log \frac{p_{ij}}{q_{ij}}$$

LLE

Locally linear embedding (LLE)

- Construct graph from distance matrix using KNN

LLE

Locally linear embedding (LLE)

- Construct graph from distance matrix using KNN
- Assumes that each node is a convex combination of its neighbors (locally linear assumption)

LLE

Locally linear embedding (LLE)

- Construct graph from distance matrix using KNN
- Assumes that each node is a convex combination of its neighbors (locally linear assumption)
- Only need to solve two Least squares problem.

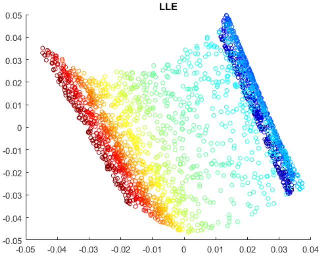
LLE

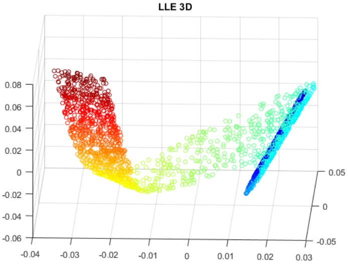
$$\min_{w_{ij}} \sum_i \|x_i - \sum_{j \in N(i)} w_{ij} x_j\|^2$$

$$\min_{y_i} \sum_i \|y_i - \sum_{j \in N(i)} w_{ij} y_j\|^2 \text{ s.t. } \|y^{(k)}\| = 1$$

solve for eigenvectors corresponding to d-smallest nonzero eigenvalues of $(I - W)^T (I - W)$.

LLE


LLE
0.05
0.04
0.03
0.02
0.01
0
-0.01
-0.02
-0.03
-0.04
-0.05
-0.05
-0.04
-0.03
-0.02
-0.01
0
0.01
0.02
0.03
0.04


LLE 3D
0.08
0.06
0.04
0.02
0
-0.02
-0.04
-0.06
-0.04
-0.03
-0.02
-0.01
0
0.01
0.02
0.03
0.05
0
-0.05

LLE

Weakness

- suffers from manifolds that contain holes
- tends to collapse large portions of the data very close together
- covariance constraint may give rise to undesired rescalings

Applications: sound source localization

Laplacian Eigenmaps

Laplacian Eigenmaps [1]

- Define similarity via Gaussian kernel $e^{-\frac{\|x_i - x_j\|^2}{t}}$.

Laplacian Eigenmaps

Laplacian Eigenmaps [1]

- Define similarity via Gaussian kernel $e^{-\frac{\|x_i - x_j\|^2}{t}}$.
- Compute geodesic distance between two nodes.

Laplacian Eigenmaps

Laplacian Eigenmaps [1]

- Define similarity via Gaussian kernel $e^{-\frac{\|x_i - x_j\|^2}{t}}$.
- Compute geodesic distance between two nodes.
- For each pair of similar nodes, it minimize the embedded pairwise distance

it solves the optimization problem

$$\min_{Y} \sum_{(i,j)} \|y_i - y_j\|^2 W_{ij}$$

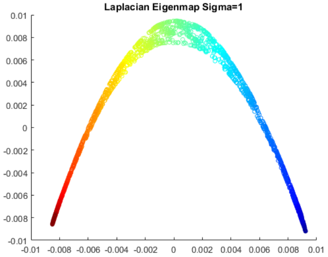
Laplacian Eigenmaps

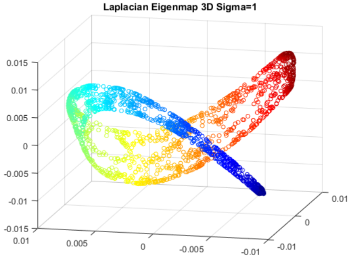
The problem above can be rewritten as:

$$\min_{Y} Tr(Y^T L Y)$$

where $Y^T D Y = I$ is used to fix the scale and avoid collapsed solution. Degree matrix D is diagonal matrix of row sum of W and graph Laplacian $L =: D - W$. The standard solution for the problem above is given by solving eigenvalue problem $Lf = \lambda D f$ and m-dimensional embedding $y_i = [f_1(i), f_2(i), ...f_m(i)]$ where f_i is the eigenvector corresponding to ith smallest eigenvalue (except 0).

Laplacian Eigenmaps





Laplacian Eigenmaps

Weakness: tends to collapse
Applications: face recognition, analysis of fMRI data, semi-supervised learning

Hessian LLE

- build a graph via KNN
- apply PCA for each $N(x_i)$ to find tengent spaces S_i
- estimate tangent Hessian H_i

$$\mathcal{H}_{lm} = \sum_i \sum_j \left((\mathbf{H}_i)_{jl} \times (\mathbf{H}_i)_{jm} \right).$$

Hessian LLE

Weakness: similar to Laplacian eigenmaps and LLE
Applications: sensor localization

Local Tangent Space Analysis (LTSA)

- build a graph via KNN
- apply PCA for each $N(x_i)$ to find tengent spaces Θ_i
- there exists a linear mapping L_i from the local tangent space coordinates Θ_{i_j} to the low-dimensional representations y_{i_j}

$$\min_{\mathbf{Y}_i,\mathbf{L}_i} \sum_i \|\mathbf{Y}_i \mathbf{J}_k - \mathbf{L}_i \Theta_i\|^2$$

Local Tangent Space Analysis (LTSA)

$$\mathbf{B}_{\mathcal{N}_i\mathcal{N}_i} = \mathbf{B}_{\mathcal{N}_{i-1}\mathcal{N}_{i-1}} + \mathbf{J}_k \left(\mathbf{I} - \mathbf{V}_i\mathbf{V}_i^T\right) \mathbf{J}_k$$

find eigenvectors corresponding to d smallest nonzero eigenvalues of the symmetric matrix $\frac{1}{2}(B + B^T)$

Local Tangent Space Analysis (LTSA)

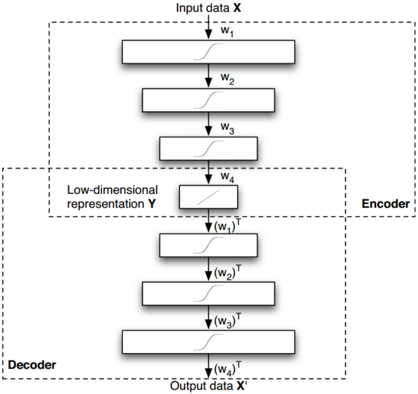
Weakness: trivial solutions
Applications: microarray data

Nonconvex Techniques: Sammon Mapping

$$\phi(\mathbf{Y}) = \frac{1}{\sum_{ij} d_{ij}} \sum_{i \neq j} \frac{(d_{ij} - \|\mathbf{y}_i - \mathbf{y}_j\|)^2}{d_{ij}}$$

Weakness: scales too much when d_{ij} small
Applications: gene and geospatial data

Nonconvex Techniques: Multilayer Autoencoders



Weakness: tedious training

Applications: data imputation, HIV data analysis

Nonconvex Techniques: Locally Linear Coordination (LLC)

- construct mixture of m factor analyzers using EM algorithms
- construct m data representations z_{ij} and their corresponding responsibilities r_{ij} for every datapoint x_i .
- build $n \times mD$ matrix U that contains $u_{ij} = r_{ij} z_{ij}$

Nonconvex Techniques: Locally Linear Coordination (LLC)

- construct mixture of m factor analyzers using EM algorithms
- construct m data representations z_{ij} and their corresponding responsibilities r_{ij} for every datapoint x_i .
- build $n \times mD$ matrix U that contains $u_{ij} = r_{ij}z_{ij}$

Model allignment:
solve $Av = \lambda Bv$, where A is inproduct of $M^T U$, $B = U^T U$,
$M = (I - W)^T (I - W)$ from LLE.
$L = [v_1, v_2, \cdots v_d]$ and $Y = UL$.

Nonconvex Techniques: Locally Linear Coordination (LLC)

Weakness: presence of local maxima in the log-likelihood function

Applications: images of a single person with variable pose and expression, handwritten digits

Nonconvex Techniques: Manifold Charting

1. find z_{ij} and r_{ij} as before.
2. find a linear mapping M from the data representations z_{ij} to the global coordinates y_i that minimizes the cost function

$$\phi(\mathbf{Y}) = \sum_{i=1}^{n} \sum_{j=1}^{m} r_{ij} \|\mathbf{y}_i - \mathbf{y}_{ij}\|^2$$

where $y_i = \sum_k y_{ik}$, $y_{ij} = z_{ij} M$.

$$\phi(\mathbf{Y}) = \sum_{i=1}^{n} \sum_{j=1}^{m} \sum_{k=1}^{m} r_{ij} r_{ik} \|\mathbf{y}_{ij} - \mathbf{y}_{ik}\|^2$$

Nonconvex Techniques: Manifold Charting

Can be rewritten as

$$\phi(\mathbf{Y}) = \mathbf{L}^T(\mathbf{D} - \mathbf{U}^T\mathbf{U})\mathbf{L}$$

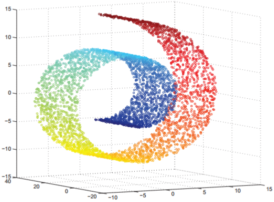
where $D = diag(D_j) = diag(\sum_i r_{ij} cov([\boldsymbol{Z_j}, \mathbf{1}])$, $u_{ij} = [r_{ij} z_{ij}, \mathbf{1}]$

$$\phi(\mathbf{Y}) = \sum_{i=1}^{n}\sum_{j=1}^{m}\sum_{k=1}^{m} r_{ij} r_{ik} \|\mathbf{y}_{ij} - \mathbf{y}_{ik}\|^2$$

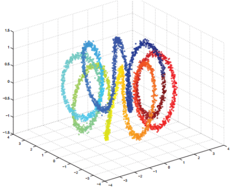
computation complexity

Technique	*Parametric*	*Parameters*	*Computational*	*Memory*
PCA	yes	none	$O(D^3)$	$O(D^2)$
Class. scaling	no	none	$O(n^3)$	$O(n^2)$
Isomap	no	k	$O(n^3)$	$O(n^2)$
Kernel PCA	no	$\kappa(\cdot,\cdot)$	$O(n^3)$	$O(n^2)$
MVU	no	k	$O((nk)^3)$	$O((nk)^3)$
Diffusion maps	no	σ, t	$O(n^3)$	$O(n^2)$
LLE	no	k	$O(pn^2)$	$O(pn^2)$
Laplacian Eigenmaps	no	k, σ	$O(pn^2)$	$O(pn^2)$
Hessian LLE	no	k	$O(pn^2)$	$O(pn^2)$
LTSA	no	k	$O(pn^2)$	$O(pn^2)$
Sammon mapping	no	none	$O(in^2)$	$O(n^2)$
Autoencoders	yes	net size	$O(inw)$	$O(w)$
LLC	yes	m, k	$O(imd^3)$	$O(nmd)$
Manifold charting	yes	m	$O(imd^3)$	$O(nmd)$

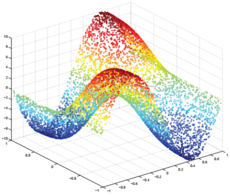
Artificial Data



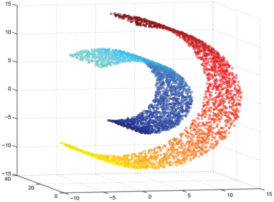
(a) Swiss roll dataset.



(b) Helix dataset.



(c) Twinpeaks dataset.



(d) Broken Swiss roll dataset.

Artificial Data

Evaluate to what extent the local structure of the data is retained:

1. the **generalization errors** of 1-nearest neighbor classifiers that are trained on the low-dimensional data representation.
2. **trustworthiness**: if low-dim points are close to each other, does high-dim ones have the same pattern?
3. **continuity**: if high dimensional points are close to each other, does the low-dim pts close to each other?

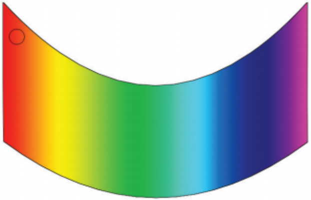
Artificial Data

$$T(k) = 1 - \frac{2}{nk(2n-3k-1)} \sum_{i=1}^{n} \sum_{j \in U_i^{(k)}} (r(i,j) - k)$$

$$C(k) = 1 - \frac{2}{nk(2n-3k-1)} \sum_{i=1}^{n} \sum_{j \in V_i^{(k)}} (\hat{r}(i,j) - k)$$

Artificial Data

(a) True underlying manifold.



(b) Reconstructed manifold up to a nonlinear warping.

Artificial Data

Technique	*Parameter settings*
PCA	None
Isomap	$5 \leq k \leq 15$
Kernel PCA	$\kappa = (\mathbf{XX}^T + 1)^5$
MVU	$5 \leq k \leq 15$
Diffusion maps	$10 \leq t \leq 100 \quad \sigma = 1$
LLE	$5 \leq k \leq 15$
Laplacian Eigenmaps	$5 \leq k \leq 15 \quad \sigma = 1$
Hessian LLE	$5 \leq k \leq 15$
LTSA	$5 \leq k \leq 15$
Sammon mapping	None
Autoencoders	Three hidden layers
LLC	$5 \leq k \leq 15 \quad 5 \leq m \leq 25$
Manifold charting	$5 \leq m \leq 25$

Artificial Data

Dataset (d)	None	PCA	Isomap	KPCA	MVU	DM	LLE	LEM	HLLE	LTSA	Sammon	Autoenc.	LLC	MC
Swiss roll (2D)	3.68%	29.76%	**3.40%**	30.24%	4.12%	33.50%	3.74%	22.06%	3.56%	3.90%	22.34%	49.00%	26.72%	22.66%
Helix (1D)	1.24%	35.50%	13.18%	38.04%	7.48%	35.44%	32.32%	15.24%	52.22%	**0.92%**	52.22%	52.22%	27.44%	25.94%
Twin peaks (2D)	0.40%	0.26%	0.22%	**0.12%**	0.56%	0.26%	0.94%	0.88%	0.14%	0.18%	0.32%	49.06%	11.04%	0.30%
Broken Swiss (2D)	2.14%	25.96%	14.48%	32.06%	32.06%	58.26%	36.94%	10.66%	**6.48%**	15.86%	27.40%	87.86%	37.06%	32.24%
HD (5D)	24.19%	22.18%	23.26%	27.46%	25.38%	23.14%	**20.74%**	24.70%	50.02%	42.62%	20.70%	49.18%	34.14%	21.34%

Table 3: Generalization errors of 1-NN classifiers trained on artificial datasets (smaller numbers are better).

Dataset (d)	None	PCA	Isomap	KPCA	MVU	DM	LLE	LEM	HLLE	LTSA	Sammon	Autoenc.	LLC	MC
Swiss roll (2D)	−−	0.88	0.99	0.88	**1.00**	0.81	**1.00**	0.92	**1.00**	**1.00**	0.89	0.46	0.81	0.88
Helix (1D)	−−	0.78	0.74	0.71	0.96	0.73	0.83	0.87	0.35	**1.00**	0.35	0.64	0.76	0.83
Twin peaks (2D)	−−	0.98	0.98	0.99	0.99	**1.00**	0.99	0.99	0.99	0.99	**1.00**	0.52	0.86	**1.00**
Broken Swiss (2D)	−−	0.96	**0.97**	0.96	**0.97**	0.78	0.94	**0.97**	0.92	0.89	**0.97**	0.70	0.86	0.96
HD (5D)	−−	**1.00**	0.98	**1.00**	0.98	**1.00**	**1.00**	0.98	0.56	0.94	**1.00**	0.68	0.89	**1.00**

Table 4: Trustworthinesses $T(12)$ on the artificial datasets (larger numbers are better).

Dataset (d)	None	PCA	Isomap	KPCA	MVU	DM	LLE	LEM	HLLE	LTSA	Sammon	Autoenc.	LLC	MC
Swiss roll (2D)	−−	**1.00**	0.99	0.99	**1.00**	0.91	**1.00**	0.99	**1.00**	**1.00**	**1.00**	0.50	0.99	**1.00**
Helix (1D)	−−	0.98	0.97	0.98	**1.00**	0.98	0.99	0.99	0.50	**1.00**	0.50	0.75	0.98	0.99
Twin peaks (2D)	−−	**1.00**	0.99	0.99	**1.00**	**1.00**	0.99	**1.00**	**1.00**	**1.00**	**1.00**	0.50	0.98	**1.00**
Broken Swiss (2D)	−−	**1.00**	0.98	0.99	**1.00**	0.90	0.98	0.99	0.99	0.99	**1.00**	0.73	0.99	**1.00**
HD (5D)	−−	**1.00**	0.99	0.99	0.99	**1.00**	0.99	0.99	0.56	0.98	**1.00**	0.89	0.91	**1.00**

Table 5: Continuity $C(12)$ on the artificial datasets (larger numbers are better).

Artificial Data

- graph-based methods in general performs well
- LLE/HLLE may perform less well on manifolds that are not isometric to Euclidean space.
- high generalization errors on the broken Swiss roll dataset
- nonlinear techniques may have problems when they are faced with a dataset with a high intrinsic dimensionality
- strong performance on the Swiss roll dataset does not always generalize to other dataset

Natural Data

Dataset: MNIST, COIL20, NiSIS, ORL, HIVA

Natural Data

Dataset (d)	*None*	*PCA*	*Isomap*	*KPCA*	*MVU*	*DM*	*LLE*	*LEM*	*HLLE*	*LTSA*	*Sammon*	*Autoenc.*	*LLC*	*MC*
MNIST (20D)	5.11%	**6.74%**	12.64%	13.86%	13.58%	25.00%	10.02%	11.30%	91.66%	90.32%	6.90%	7.18%	16.12%	14.84%
COIL20 (5D)	0.14%	3.82%	15.69%	7.78%	25.14%	11.18%	22.29%	95.00%	50.35%	4.17%	**0.83%**	51.11%	4.31%	27.36%
ORL (8D)	2.50%	4.75%	27.50%	6.25%	24.25%	90.00%	11.00%	97.50%	56.00%	12.75%	2.75%	**6.25%**	11.25%	22.50%
NiSIS (15D)	8.24%	**7.95%**	13.36%	9.55%	15.67%	48.98%	15.48%	47.59%	48.98%	24.68%	48.98%	9.22%	26.86%	18.91%
HIVA (15D)	4.63%	5.05%	4.92%	5.07%	4.94%	5.46%	4.97%	4.81%	**3.51%**	**3.51%**	**3.51%**	5.12%	**3.51%**	4.79%

Table 6: Generalization errors of 1-NN classifiers trained on natural datasets (smaller numbers are better).

Dataset (d)	*None*	*PCA*	*Isomap*	*KPCA*	*MVU*	*DM*	*LLE*	*LEM*	*HLLE*	*LTSA*	*Sammon*	*Autoenc.*	*LLC*	*MC*
MNIST (20D)	−−	**1.00**	0.96	0.99	0.92	0.95	0.96	0.89	0.54	0.54	**1.00**	**1.00**	0.93	0.97
COIL20 (5D)	−−	**0.99**	0.89	0.98	0.92	0.91	0.93	0.27	0.69	0.96	**0.99**	0.88	0.96	0.92
ORL (8D)	−−	**0.99**	0.78	0.98	0.95	0.49	0.95	0.29	0.76	0.94	**0.99**	**0.99**	0.79	0.82
NiSIS (15D)	−−	**0.99**	0.89	**0.99**	0.90	0.40	0.92	0.47	0.47	0.82	0.47	**0.99**	0.85	0.89
HIVA (15D)	−−	0.97	0.87	0.89	0.89	0.75	0.80	0.78	0.42	0.54	0.42	**0.98**	0.91	0.95

Table 7: Trustworthinesses $T(12)$ on the natural datasets (larger numbers are better).

Dataset (d)	*None*	*PCA*	*Isomap*	*KPCA*	*MVU*	*DM*	*LLE*	*LEM*	*HLLE*	*LTSA*	*Sammon*	*Autoenc.*	*LLC*	*MC*
MNIST (20D)	−−	**1.00**	0.94	0.89	0.93	0.95	0.96	0.70	0.50	0.50	**1.00**	**1.00**	0.91	0.96
COIL20 (5D)	−−	**1.00**	0.90	0.98	0.97	0.92	0.95	0.47	0.71	0.99	**1.00**	0.92	0.96	0.95
ORL (8D)	−−	**0.99**	0.76	0.95	0.97	0.57	0.95	0.49	0.76	0.94	**0.99**	0.98	0.80	0.79
NiSIS (15D)	−−	**1.00**	0.84	0.98	0.94	0.48	0.91	0.48	0.47	0.64	0.47	**1.00**	0.84	0.89
HIVA (15D)	−−	**0.99**	0.84	0.88	0.94	0.80	0.80	0.54	0.51	0.62	0.51	**0.99**	0.87	0.96

Table 8: Continuity $C(12)$ on the natural datasets (larger numbers are better).

Full Spectral methods

- **graph-based methods**: may suffer from short-curcuiting issue
- **kernel methods**: choose proper kernel is an issue (suffer from curse of dimensionality)

Sparse spectral methods:

- covariance constraint can be easily cheated
- curse of dimensionality
- difficulty of solving eigen problems
- overfitting (data distribution),
- outliers (use eps-ball instead of KNN)
- real-world data violates smoothness assumption

Nonconvex methods:

- cons: may stuck at local min/max
- pros: more flexibility in designing formulation, may allow higher model complexity and tackle more variations of data.

- M. Belkin, and P. Niyogi. "Laplacian eigenmaps for dimensionality reduction and data representation." *Neural computation* 15, no. 6 (2003): 1373-1396.
- L. Bin, R. Wilson, and E. Hancock. "Spectral embedding of graphs." *Pattern recognition* 36, no. 10 (2003): 2213-2230.
- S. Blake, and T. Jebara. "Structure preserving embedding." *In Proceedings of the 26th Annual International Conference on Machine Learning*, pp. 937-944. ACM, 2009.
- J. Kruskal, and M. Wish. Multidimensional scaling. Vol. 11. Sage, 1978.
- B. Mikhail, and P. Niyogi. "Laplacian eigenmaps and spectral techniques for embedding and clustering." *In NIPS*, vol. 14, pp. 585-591. 2001.

- B. Shaw. Graph embedding and nonlinear dimensionality reduction. Columbia University, 2011.
- B. Schlkopf, A. Smola, and K. Mller. "Nonlinear component analysis as a kernel eigenvalue problem." *Neural computation 10*, no. 5 (1998): 1299-1319.
- J. Tenenbaum "Mapping a manifold of perceptual observations." Advances in neural information processing systems (1998): 682-688.
- K. Weinberger, and L. Saul. "An introduction to nonlinear dimensionality reduction by maximum variance unfolding." *In AAAI*, vol. 6, pp. 1683-1686. 2006.
- S. Wold, E. Kim, and G. Paul. "Principal component analysis." *Chemometrics and intelligent laboratory systems* 2, no. 1-3 (1987): 37-52.

Thanks!