

Dimensionality Reduction

Evgeny Burnaev

Skoltech, Moscow, Russia



Skolkovo Institute of Science and Technology

- 1 Dimensionality Reduction: Problem Statement
- 2 Principal Component Analysis
- 3 Multi-Dimensional Scaling
- 4 Replicative Neural Networks
- 5 Graphs based on nearest neighbours
- 6 ISOMAP
- 7 Locally Linear Embedding
- 8 Stochastic Neighbour Embedding

1 Dimensionality Reduction: Problem Statement

- 2 Principal Component Analysis
- 3 Multi-Dimensional Scaling
- 4 Replicative Neural Networks
- 5 Graphs based on nearest neighbours
- 6 ISOMAP
- 7 Locally Linear Embedding
- 8 Stochastic Neighbour Embedding

- Object O is described by a p -dimensional vector $\mathbf{x}(O) \in \mathbb{R}^p$. Components of $\mathbf{x}(O)$ are features of O
- Example 1 (human face):** Image is represented by 1024×1024 pixels — dimension $p = 10^{20} \sim 10000000$

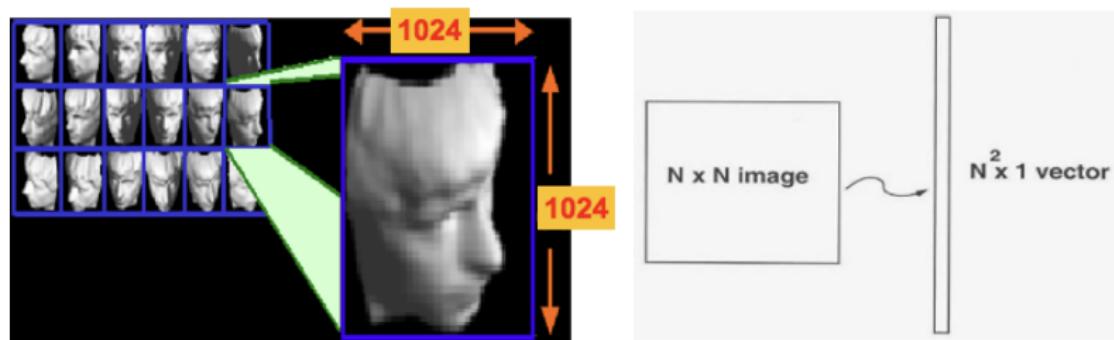
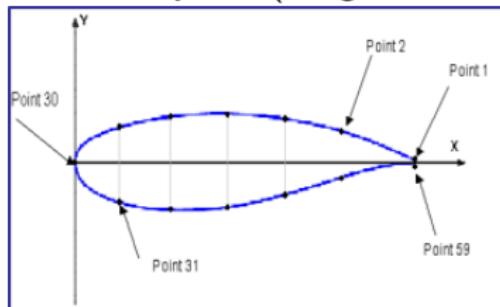


Figure – Grey-scale pixel-wise representation of a human face

- Example 2 (wing airfoil description)



Airfoil

$O \rightarrow \mathbf{x}(O) = (x_1, x_2, \dots, x_p)^\top$ – ordinates of upper and lower contours, $p \sim 50 \div 200$

- Example 3 (plots of dependencies, multidimensional time-series)

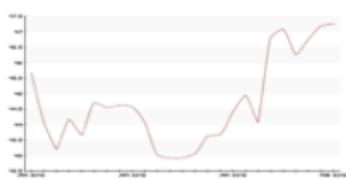


Figure – Electricity price curve

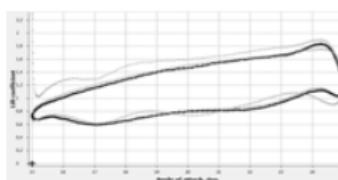


Figure – Lift force vs. AoA curve



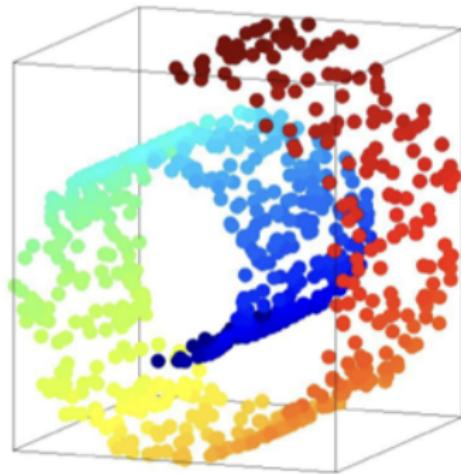
Figure – Speech recognition

Curve $f(x)$ is described by the vector $f = (f_1, f_2, \dots, f_p)^\top \in \mathbb{R}^p$, $f_j = f(t_j)$; time-series segment (x_1, x_2, \dots, x_p) is considered as a p -dimensional vector

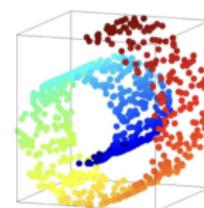
- Example 4 (MNIST)



- Example 5 (Famous Toy Problem “Swiss Roll”)



- High dimensionality p of $\mathbf{x}(O)$ is critical for efficient learning
 - We can visualize only in 2D/3D
- The main idea of Dimensionality Reduction is constructing reduced dimension representation $\mathbf{z}(O) \in \mathbb{R}^q, q \ll p$, of O without “significant loss of information”
- DR for
 - Visualization
 - Data compression
 - “Curse of dimensionality”
$$\mathbf{x} \rightarrow \mathbf{x}' \quad \text{s.t.} \quad \dim(\mathbf{x}') \ll \dim(\mathbf{x})$$
 - De-noising
 - Reasonable distance metrics



uncovers intrinsic
dimensionality
(invertible)

Compressed description $\mathbf{z}(O)$ is constructed for a detailed description $\mathbf{x}(O)$ using a learning sample

$$\mathbf{X}_m = \{\mathbf{x}_i = \mathbf{x}(O_i), i = 1, 2, \dots, m\} = \{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_m\}$$

of features for objects O_1, O_2, \dots, O_m

without using physical information about objects

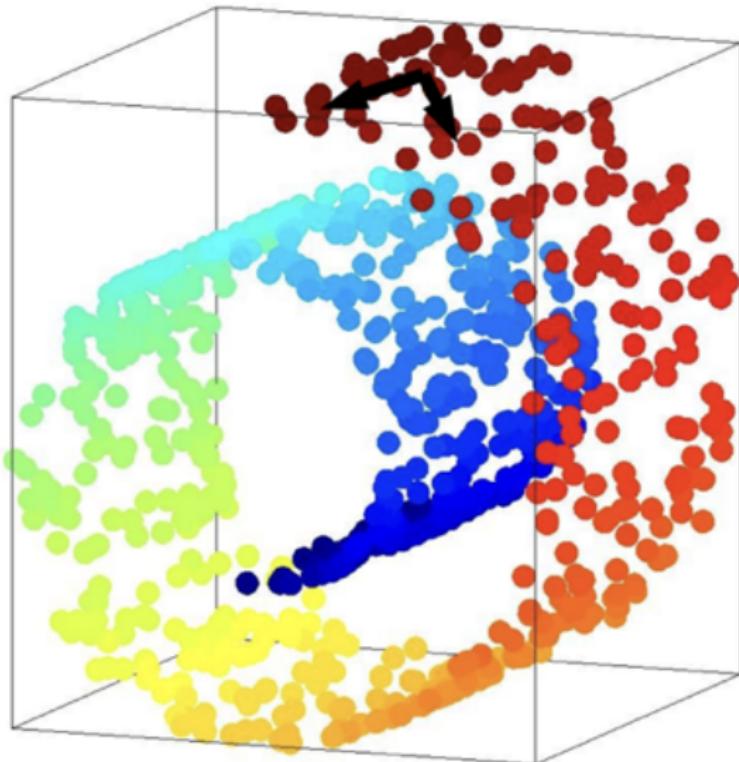


Figure – Appearance variation

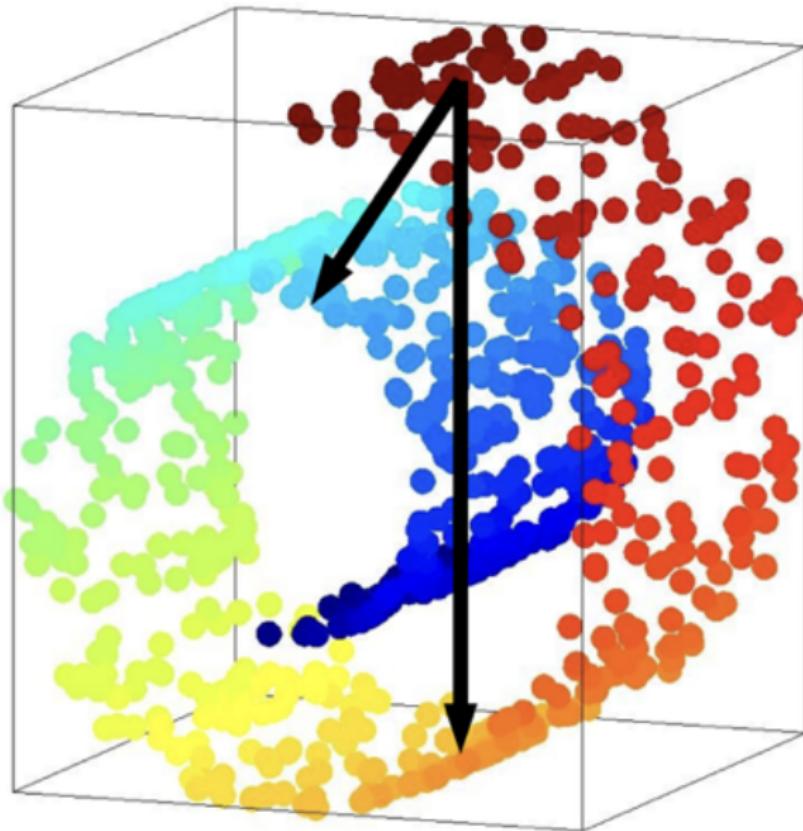
What is “Reasonable distance metrics”?



What is “Reasonable distance metrics”?



What is “Reasonable distance metrics”?



Reasonable distance metrics





linear interpolation



manifold interpolation

Types of manifold

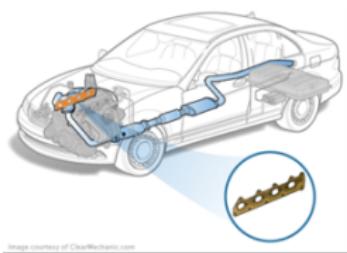


Figure – Exhaust Manifold

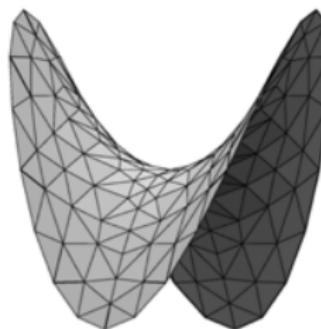


Figure – Low-dim surface embedded in high-dim space



Figure – Sir Walter Synnot Manifold (1849 - 1928)

Types of manifold

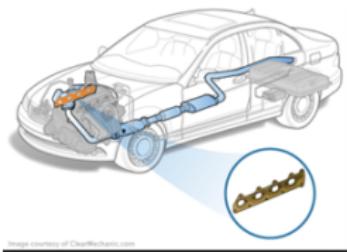


Figure – Exhaust Manifold

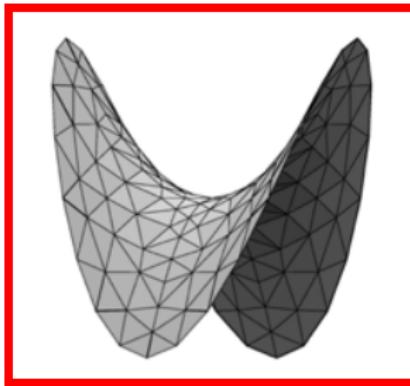


Figure – Low-dim surface embedded in high-dim space



Figure – Sir Walter Synnot Manifold
(1849 - 1928)

Embedding Problem. Using a sample

$$\mathbf{X}_m = \{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_m\} \subset \mathbb{R}^p$$

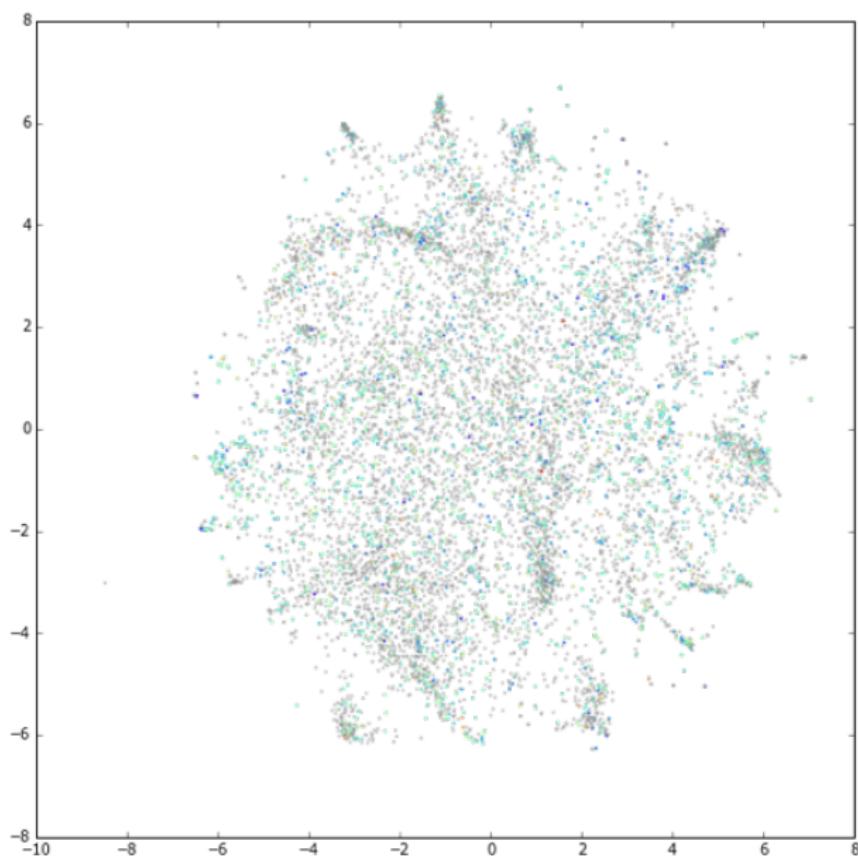
construct an embedding

$$h : \mathbf{X}_m \rightarrow \mathbf{Z}_m = h(\mathbf{X}_m) = \{\mathbf{z}_1, \mathbf{z}_2, \dots, \mathbf{z}_m\} \subset \mathbb{R}^q$$

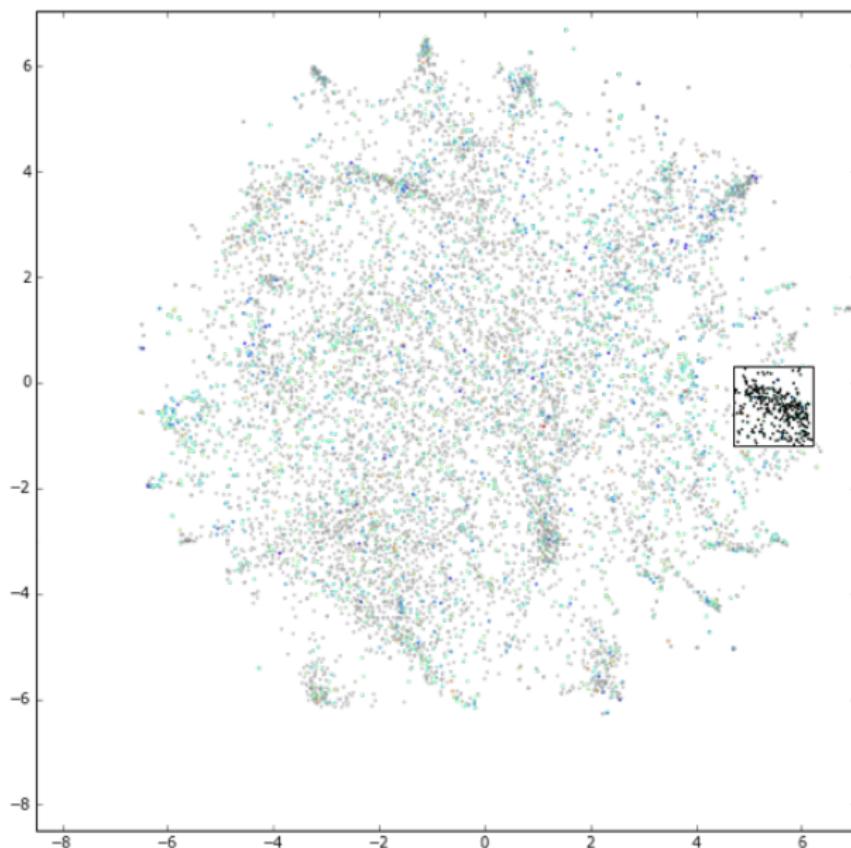
of sample points \mathbf{X}_m in \mathbb{R}^q , $q < p$, such that the set \mathbf{Z}_m “consistently represents” the set \mathbf{X}_m

(e.g., the set \mathbf{Z}_m should preserve some geometric structure of \mathbf{X}_m , etc.)

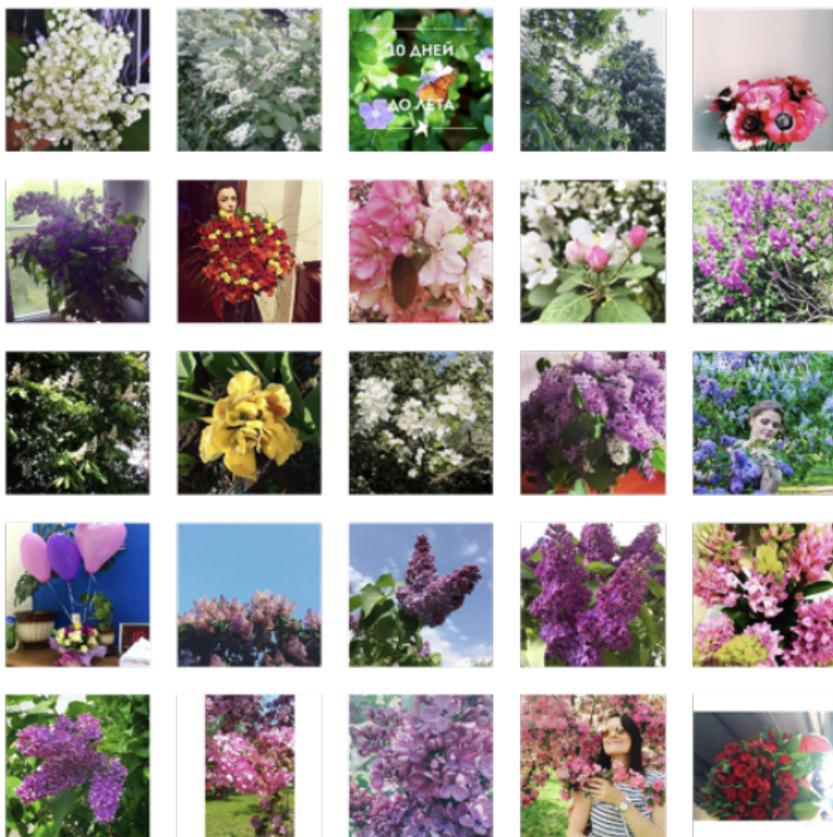
Embedding of Images



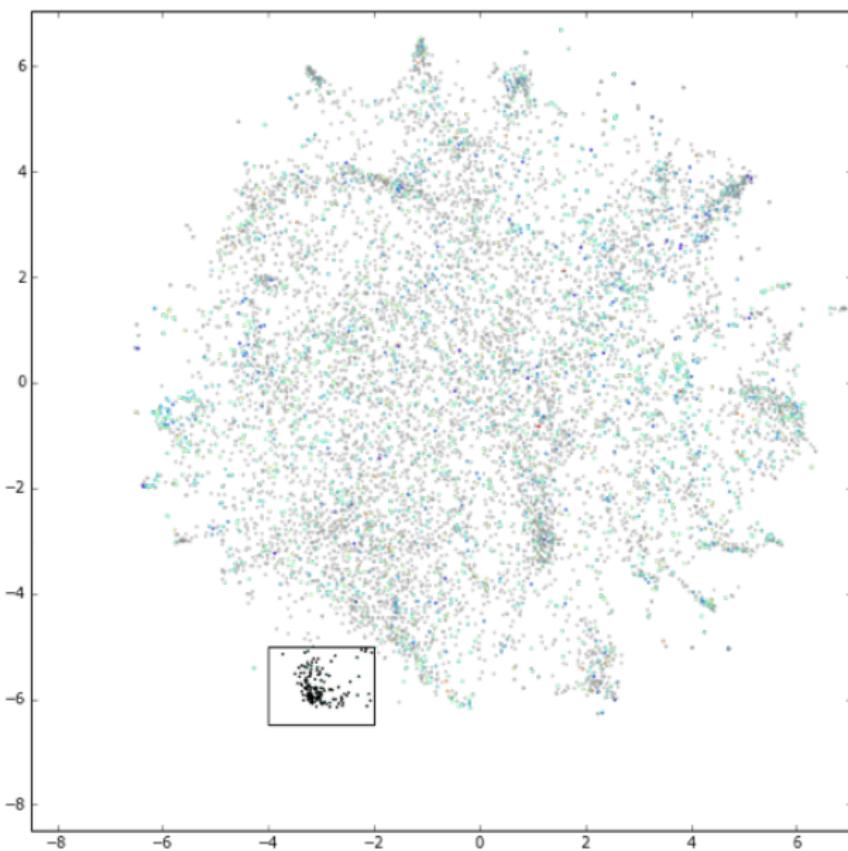
Embedding of Images



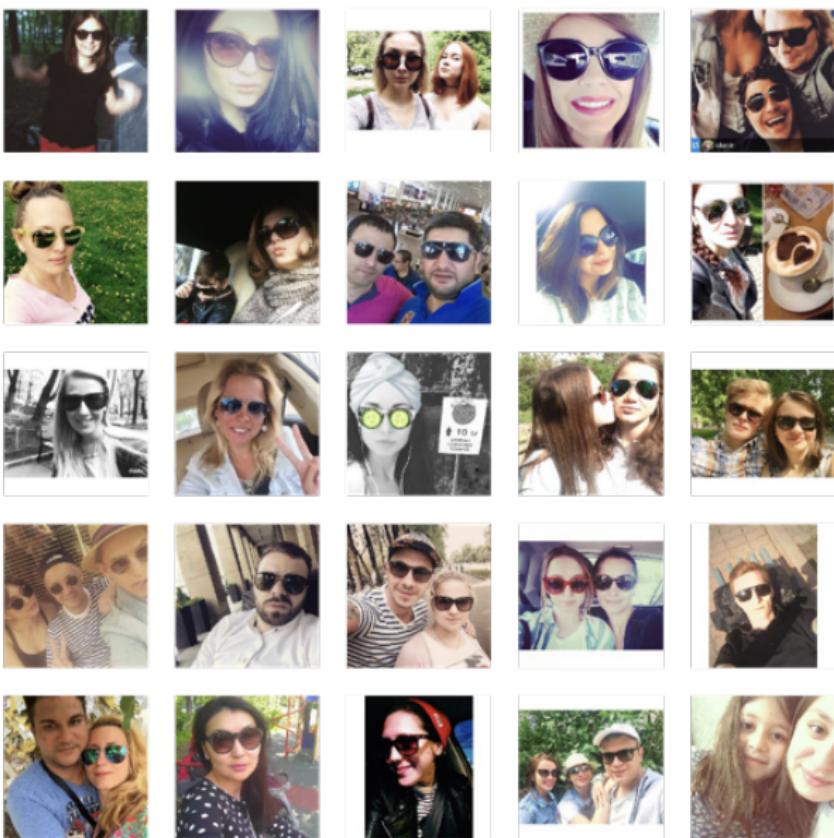
Embedding of Images



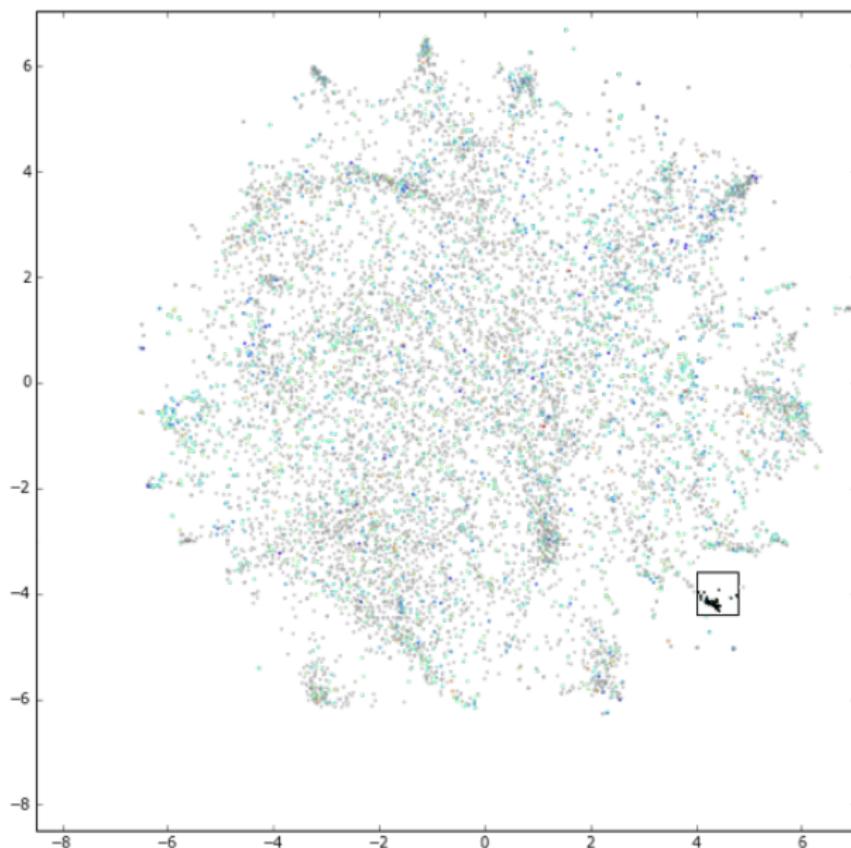
Embedding of Images



Embedding of Images



Embedding of Images



Embedding of Images



- **Typical Scheme:**

- construct some cost function $L(\mathbf{z}_1, \mathbf{z}_2, \dots, \mathbf{z}_m)$. E.g. for Multi Dimensional Scaling (**MDS**)

$$L(\mathbf{z}_1, \mathbf{z}_2, \dots, \mathbf{z}_m) = \sum_{i,j} (\rho(\mathbf{x}_i, \mathbf{x}_j) - \|\mathbf{z}_i - \mathbf{z}_j\|)^2$$

- optimize cost function
 - visually analyze results(Swiss Roll, Spiral, ...)

- **Popular methods to solve embedding problem:**

- Pursuit Projection; Principal Component Analysis
 - Locally Linear Embedding (LLE); Conformal Eigenmaps
 - Laplacian Eigenmaps (LE)
 - Hessian Eigenmaps (HE)
 - ISOmetric MAPping (ISOMAP); Landmark ISOMAP
 - Kernel PCA (KPCA)
 - Local Tangent Space Alignment (LTSA)

Using a sample

$$\mathbf{X}_m = \{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_m\} \subset \mathbb{R}^p$$

construct an embedding transformation

$$h : \mathbf{X} \subset \mathbb{R}^p \rightarrow \mathbf{Z} = h(\mathbf{X}) \subset \mathbb{R}^q$$

both for sample points \mathbf{X}_m and **for new** (out-of-sample) **points**

$$\mathbf{x}_{\text{new}} \in \mathbf{X}/\mathbf{X}_m$$

without solving embedding problem for $\mathbf{X}_m \cup \{\mathbf{x}\}$ anew

Example — face recognition

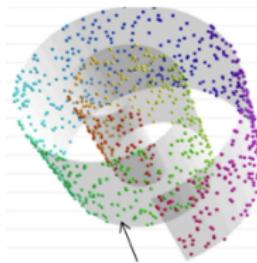
- For the extended embedding problem we need a Data Model \mathbf{X} : description of \mathbf{X} and of a mechanism, which generates points \mathbf{X}_m and new points \mathbf{x}_{new} from \mathbf{X}
- Usually we assume that real data is well approximated by some manifold, i.e.

Manifold Data Model:

$$\mathbf{X} = \{\mathbf{x} = f(\mathbf{b}) \in \mathbb{R}^p : \mathbf{b} \in \mathbf{B} \subset \mathbb{R}^q\} \subset \mathbb{R}^p$$

- open set \mathbf{B} (inner coordinate space)
- smooth bijective transformation $f : \mathbf{B} \rightarrow \mathbb{R}^p$

- Manifold Learning Problem!!!



Using a sample

$$\mathbf{X}_m = \{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_m\} \subset \mathbb{R}^p$$

construct embedding transformation

$$h : \mathbf{X} \subset \mathbb{R}^p \rightarrow \mathbf{Z} = h(\mathbf{X}) \subset \mathbb{R}^q$$

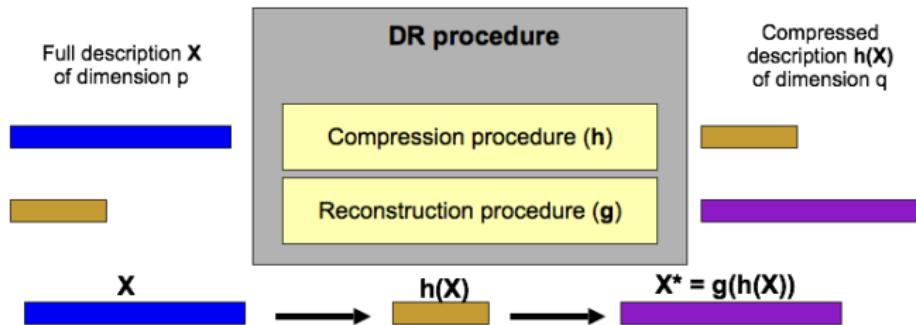
and **Reconstruction transformation**

$$g : \mathbf{Z} \subset \mathbb{R}^q \rightarrow \mathbf{X} \subset \mathbb{R}^p$$

such that

$$g(h(\mathbf{x})) \approx \mathbf{x} \text{ for all } \mathbf{x} \in \mathbf{X}$$

Example - dimension reduction of airfoils



Requirements:

- minimal dimension $q = \dim h(\mathbf{x})$ of compressed description $h(\mathbf{x})$, providing required proximity between $\mathbf{x}^* = g(h(\mathbf{x}))$ and \mathbf{x}
or
- maximal proximity between \mathbf{x} and $\mathbf{x}^* = g(h(\mathbf{x}))$ for a fixed dimension $q = \dim h(\mathbf{x})$ of compressed description

1 Dimensionality Reduction: Problem Statement

2 Principal Component Analysis

3 Multi-Dimensional Scaling

4 Replicative Neural Networks

5 Graphs based on nearest neighbours

6 ISOMAP

7 Locally Linear Embedding

8 Stochastic Neighbour Embedding

Problem: find \mathbb{R}^p an affine subspace

$$L(q) = \left\{ \mathbf{x} \in \mathbb{R}^p : \mathbf{x} = \mathbf{x}_0 + \sum_{j=1}^q z_j \times \mathbf{e}_j, (z_1, z_2, \dots, z_q) \in \mathbb{R}^q \right\}$$

of dimension $q < p$, **which best approximates the set of points**

$$\mathbf{X}_m = \left\{ \mathbf{x}_i, i = 1, 2, \dots, m \right\} \subset \mathbb{R}^p$$

in PCA: “the best” = minimize w.r.t. $\mathbf{x}_0, \{\mathbf{e}_1, \mathbf{e}_2, \dots, \mathbf{e}_q\} \subset \mathbb{R}^p$

$$\frac{1}{m} \sum_{j=1}^m \| \mathbf{x}_j - P_{L(q)} \mathbf{x}_j \|^2$$

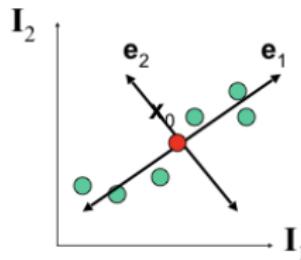
$$\text{Pr}_{L(q)}(\mathbf{x}) = \mathbf{x}_0 + \sum_{j=1}^q z_j(\mathbf{x}) \times \mathbf{e}_j, \quad z_j(\mathbf{x}) = (\mathbf{x} - \mathbf{x}_0, \mathbf{e}_j)$$

\mathbf{x}_{mean} — empirical mean of $\{\mathbf{x}_i, i = 1, 2, \dots, m\}$

$\{\mathbf{e}_1, \mathbf{e}_2, \dots, \mathbf{e}_p\}$ — eigenvalues of an empirical covariance ($p \times p$)-matrix

$$\Sigma = \frac{1}{m} \sum_{j=1}^m (\mathbf{x}_j - \mathbf{x}_{\text{mean}}) \times (\mathbf{x}_j - \mathbf{x}_{\text{mean}})^{\top}$$

providing orthonormal basis in \mathbb{R}^p



$$L(1) = \{\mathbf{x} \in R^2 : \mathbf{x} = \mathbf{x}_0 + z_1 \times \mathbf{e}_1, z_1 \in R^1\}, p = 2, q = 1$$

Solution: $\mathbf{x}_0 = \mathbf{x}_{\text{mean}}$, $L(q) = \mathbf{x}_0 \oplus \text{Span}(\mathbf{e}_1, \mathbf{e}_2, \dots, \mathbf{e}_q)$,

where orthonormal eigenvectors $\{\mathbf{e}_1, \mathbf{e}_2, \dots, \mathbf{e}_q\}$ of Σ correspond to q highest eigenvalues of this matrix

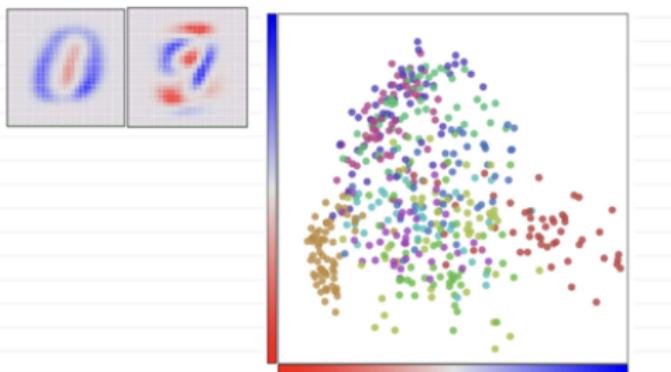
embedding procedure:

$$\mathbf{x} \in \mathbb{R}^p \rightarrow h(\mathbf{x}) = (z_1(\mathbf{x}), z_2(\mathbf{x}), \dots, z_q(\mathbf{x}))^\top \in \mathbb{R}^q$$

reconstruction procedure:

$$\mathbf{z} = (z_1, z_2, \dots, z_q)^\top \in \mathbb{R}^q \rightarrow g(\mathbf{z}) = \mathbf{x}_0 + z_1 \times \mathbf{e}_1 + z_2 \times \mathbf{e}_2 + \dots + z_q \times \mathbf{e}_q$$

PCA on MNIST



<http://colah.github.io/posts/2014-10-Visualizing-MNIST/>

- Eigenvectors $\mathbf{e}_1, \mathbf{e}_2, \dots, \mathbf{e}_p \in \mathbb{R}^p$ in space of faces: $p \sim 10^6$

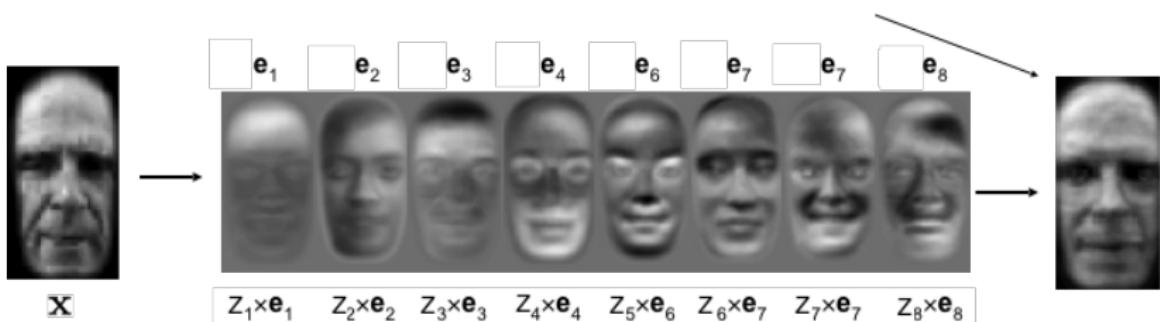
$$\mathbf{x} \rightarrow ((\mathbf{x} - \mathbf{x}_0, \mathbf{e}_1), (\mathbf{x} - \mathbf{x}_0, \mathbf{e}_2), \dots, (\mathbf{x} - \mathbf{x}_0, \mathbf{e}_p))^\top \in \mathbb{R}^p$$

where

$$z_1 = (\mathbf{x} - \mathbf{x}_0, \mathbf{e}_1), z_2 = (\mathbf{x} - \mathbf{x}_0, \mathbf{e}_2), \dots, z_p = (\mathbf{x} - \mathbf{x}_0, \mathbf{e}_p)$$

- We select first q , $q \sim 10^2$ eigenvectors $\mathbf{e}_1, \mathbf{e}_2, \dots, \mathbf{e}_q \in \mathbb{R}^q$

$$\mathbf{x}^* \approx \mathbf{x}_0 + z_1 \times \mathbf{e}_1 + z_2 \times \mathbf{e}_2 + \dots + z_q \times \mathbf{e}_q$$



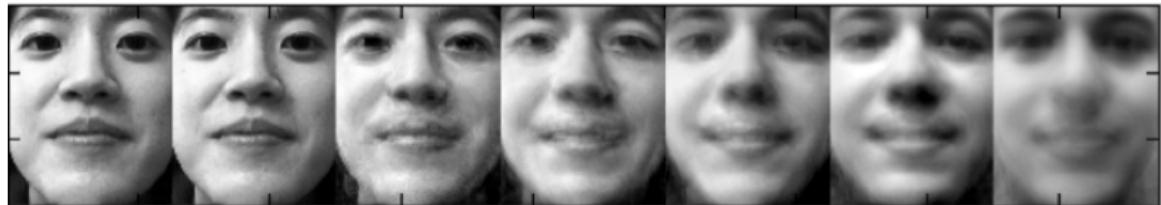
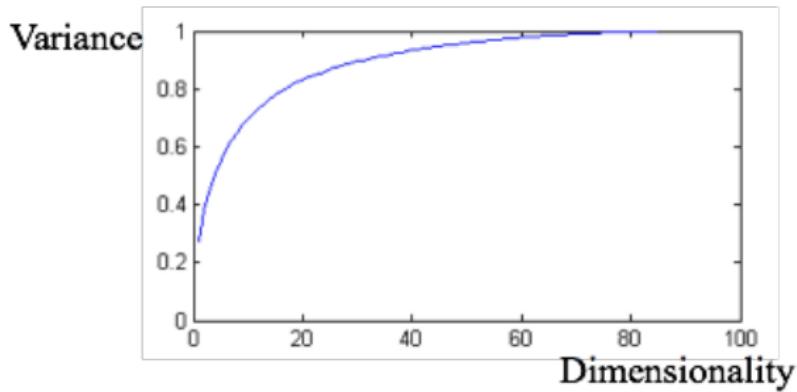


Figure – Left to right: original, reconstruction from 84, 40, 20, 3, 2 and 1 dimensions.



1 Dimensionality Reduction: Problem Statement

2 Principal Component Analysis

3 Multi-Dimensional Scaling

4 Replicative Neural Networks

5 Graphs based on nearest neighbours

6 ISOMAP

7 Locally Linear Embedding

8 Stochastic Neighbour Embedding

Minimize quadratic form

$$\sum_{i,j} (\rho(O_i, O_j) - \|\mathbf{z}_i - \mathbf{z}_j\|)^2$$

w.r.t. $\mathbf{z}_1, \mathbf{z}_2, \dots, \mathbf{z}_m \in \mathbb{R}^q$,

where ρ is some metric in the object feature space

$\mathbf{z}_1, \mathbf{z}_2, \dots, \mathbf{z}_m \in (R)^q$ are defined **except for shift and rotation**, thus we use normalization, e.g.

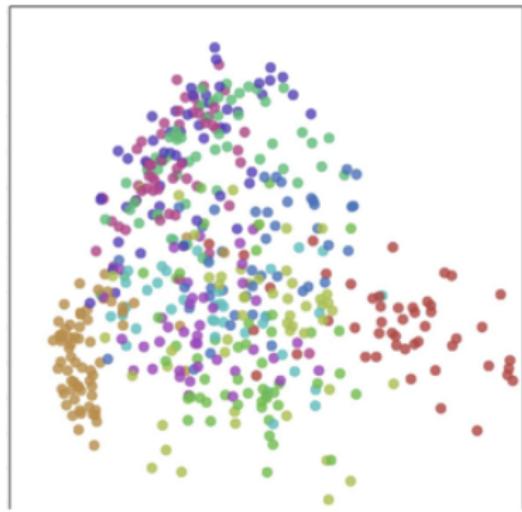
$$\mathbf{Z}^\top \times \mathbf{Z} = I_q \text{ and } \mathbf{Z}^\top \times \mathbf{1} = \mathbf{0},$$

where $\mathbf{Z}^\top = (\mathbf{z}_1 : \mathbf{z}_2 : \dots : \mathbf{z}_m)$ — $(q \times m)$ -matrix, $\mathbf{1} \in \mathbb{R}^m$ — vector of ones

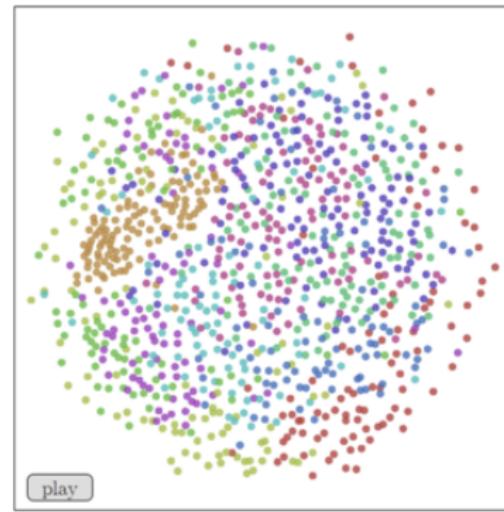
If $\rho(O_i, O_j) = \|\mathbf{x}(O_i) - \mathbf{x}(O_j)\|$, then MDS = PCA

Common useful idea: local distances are often more important to preserve

$$\sum_{i,j} \frac{(\rho(O_i, O_j) - \|\mathbf{z}_i - \mathbf{z}_j\|)^2}{(\rho(O_i, O_j))^2}$$



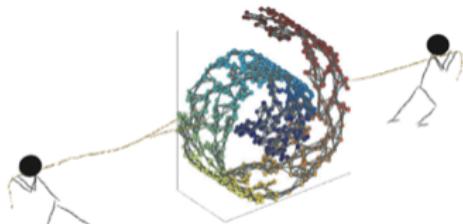
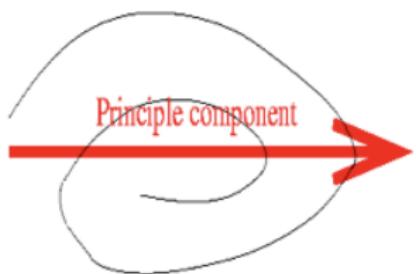
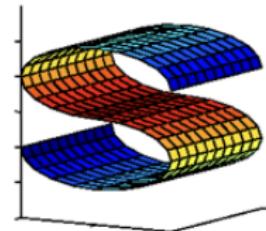
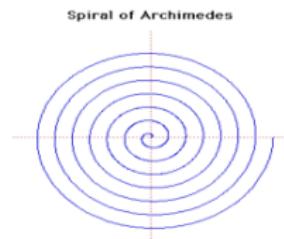
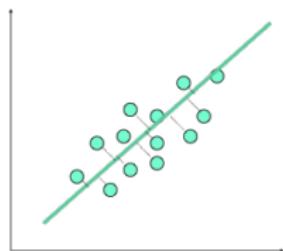
PCA



Sammon

play

MDS, PCA, PP, MDS (for “Euclidean proximity”) — linear methods, do not work for “nonlinear data”



1 Dimensionality Reduction: Problem Statement

2 Principal Component Analysis

3 Multi-Dimensional Scaling

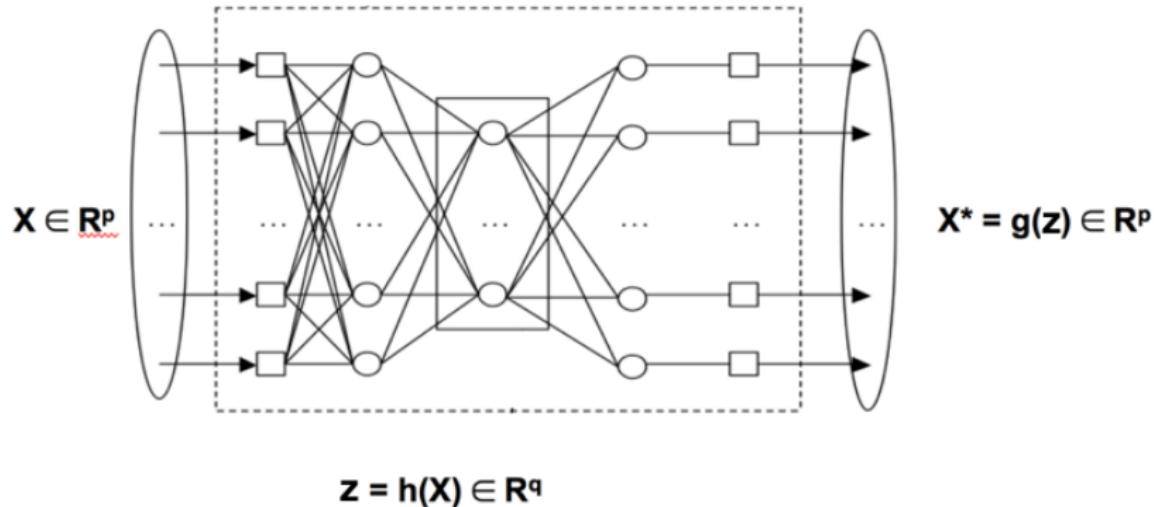
4 Replicative Neural Networks

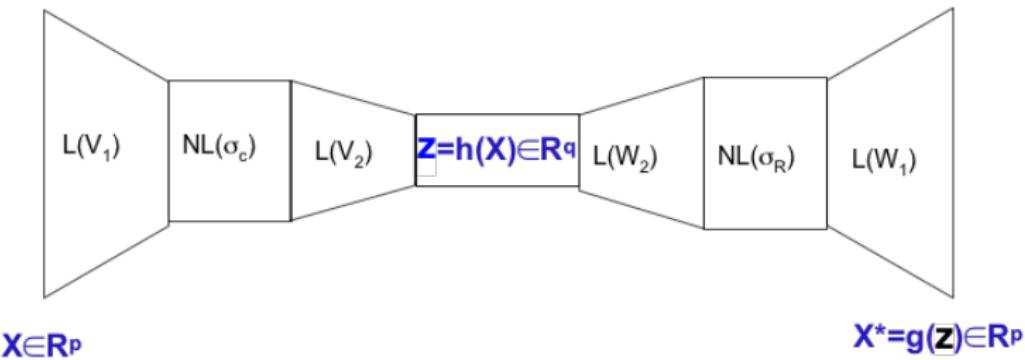
5 Graphs based on nearest neighbours

6 ISOMAP

7 Locally Linear Embedding

8 Stochastic Neighbour Embedding



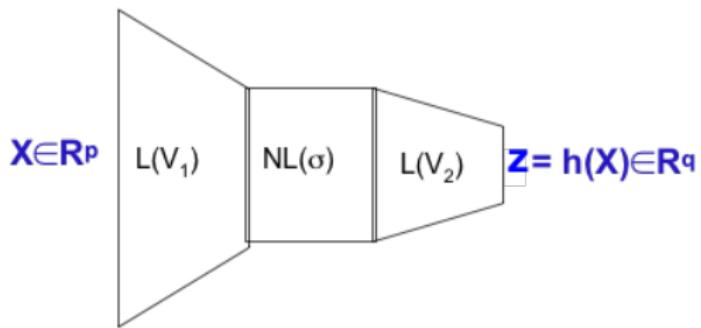
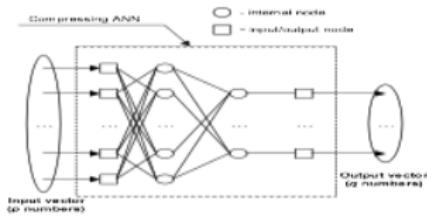


$$R = \frac{1}{m} \sum_{j=1}^m \|\mathbf{x}_j - \mathbf{x}_j^*\|^2$$

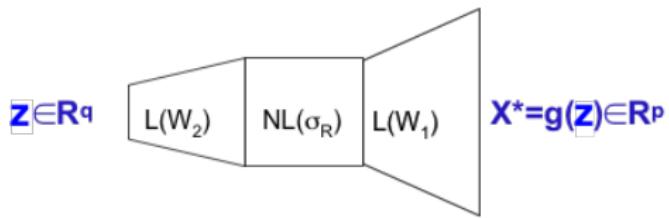
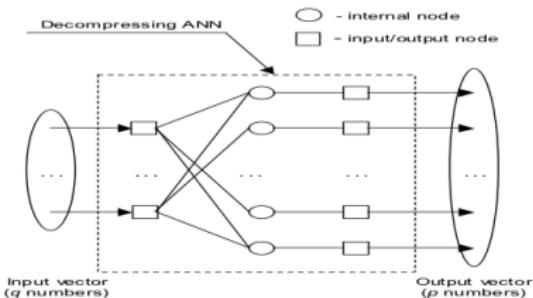
- Four linear transformations with matrices V_1, V_2, W_1 and W_2
- Two fixed **nonlinear** transformations

$$\sigma(x) - \text{sigmoid function, e.g., } \sigma(x) = \frac{1}{(1 + e^{-x})}$$

- Back-propagation to optimize w.r.t. to parameters V_1, V_2, W_1 and W_2 the reconstruction error R



Reconstruction Transformation



Curse of dimensionality

$$V(r) \sim C(d) * r^d$$

$$\frac{V(r) - V(r(1 - \epsilon))}{V(r)} = \frac{1 - (1 - \epsilon)^d}{1} \rightarrow 1$$

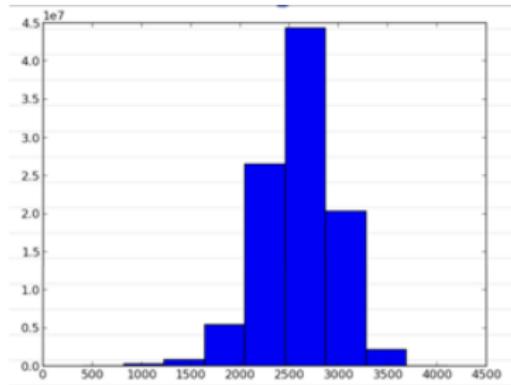


Figure – Histogram of pairwise Euclidian distances for MNIST

Consider more clever distance!

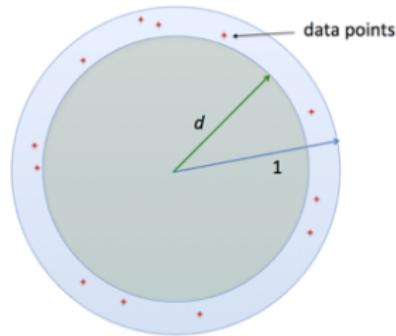


Figure – All volume of a ball is close to its cover

- Locally Linear Embedding, LLE
- Laplacian Eigenmaps, LE
- Hessian Eigenmaps, HE
- ISOmetric MAPing, ISOMAP
- Kernel PCA, KPCA - Spectral Embedding Algorithm, SEA
- Riemannian Manifold Learning, RML
- Local Tangent Space Alignment, LTSA ...

1 Dimensionality Reduction: Problem Statement

2 Principal Component Analysis

3 Multi-Dimensional Scaling

4 Replicative Neural Networks

5 Graphs based on nearest neighbours

6 ISOMAP

7 Locally Linear Embedding

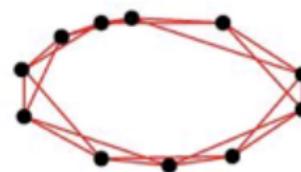
8 Stochastic Neighbour Embedding

- **Step 1. Construct neighbourhoods** (ϵ -Neighbourhoods, or using k -Nearest Neighbours)

$$U(\mathbf{x}) = U(\mathbf{x}|\rho, \epsilon) = \{\mathbf{x}_i \in \mathbf{X}_m : \rho(\mathbf{x}, \mathbf{x}_i) \leq \epsilon\}$$

\Rightarrow **Graph** $\Gamma(\mathbf{X}_m) = (\mathbf{X}_m, V) : (\mathbf{x}_i, \mathbf{x}_j) \in \mathbf{V} \Leftrightarrow$

$\mathbf{x}_j \in U(\mathbf{x}_i)$ and $\mathbf{x}_i \in U(\mathbf{x}_j)$



- **Step 2. Construct Weights**

$$w_{ij} = w(\mathbf{x}_i, \mathbf{x}_j) = w(v), v = (\mathbf{x}_i, \mathbf{x}_j) \in \mathbf{V}$$

$w_{0,ij} = 1$ for all $(\mathbf{x}_i, \mathbf{x}_j) \in V$, $w_{0,ij} = 0$ in other cases

1 Dimensionality Reduction: Problem Statement

2 Principal Component Analysis

3 Multi-Dimensional Scaling

4 Replicative Neural Networks

5 Graphs based on nearest neighbours

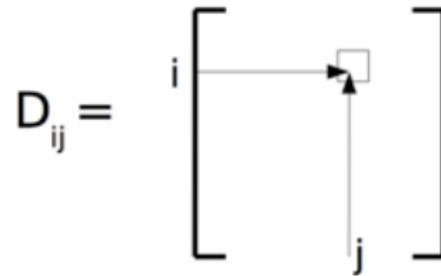
6 ISOMAP

7 Locally Linear Embedding

8 Stochastic Neighbour Embedding

Usual MDS

For m data points, and a distance matrix D

$$D_{ij} = \begin{bmatrix} & i \\ & \downarrow \\ j & \end{bmatrix}$$


... we can construct an q -dimensional space to preserve inter-points distances by using the top eigenvectors of D scaled by their eigenvalues

$$\mathbf{z}_i = [\sqrt{\lambda_1}v_1^i, \sqrt{\lambda_2}v_2^i, \dots, \sqrt{\lambda_q}v_q^i]$$

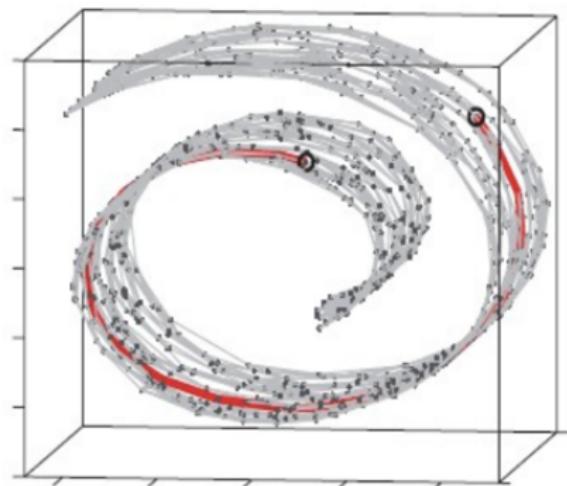
Infer a distance matrix using distances along the manifold.



- Build a sparse graph with k -nearest neighbours

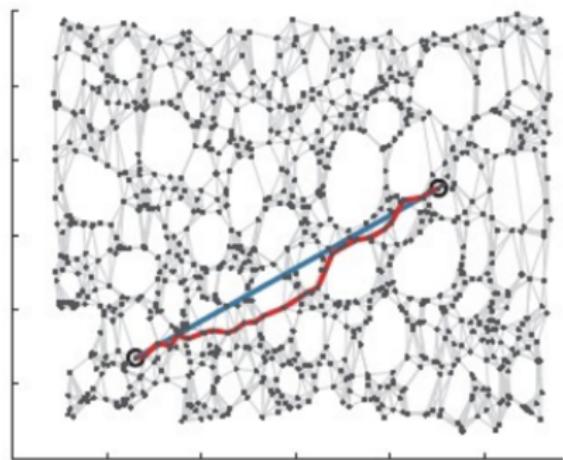
$$D_g = \begin{bmatrix} & & \\ & \text{blue oval} & \\ & & \end{bmatrix}$$

Figure – Distance matrix is sparse



- Infer other interpoint distances by finding shortest paths on the graph (Dijkstra's algorithm).

$$D_g = \begin{bmatrix} & \\ & \end{bmatrix}$$



Usual MDS

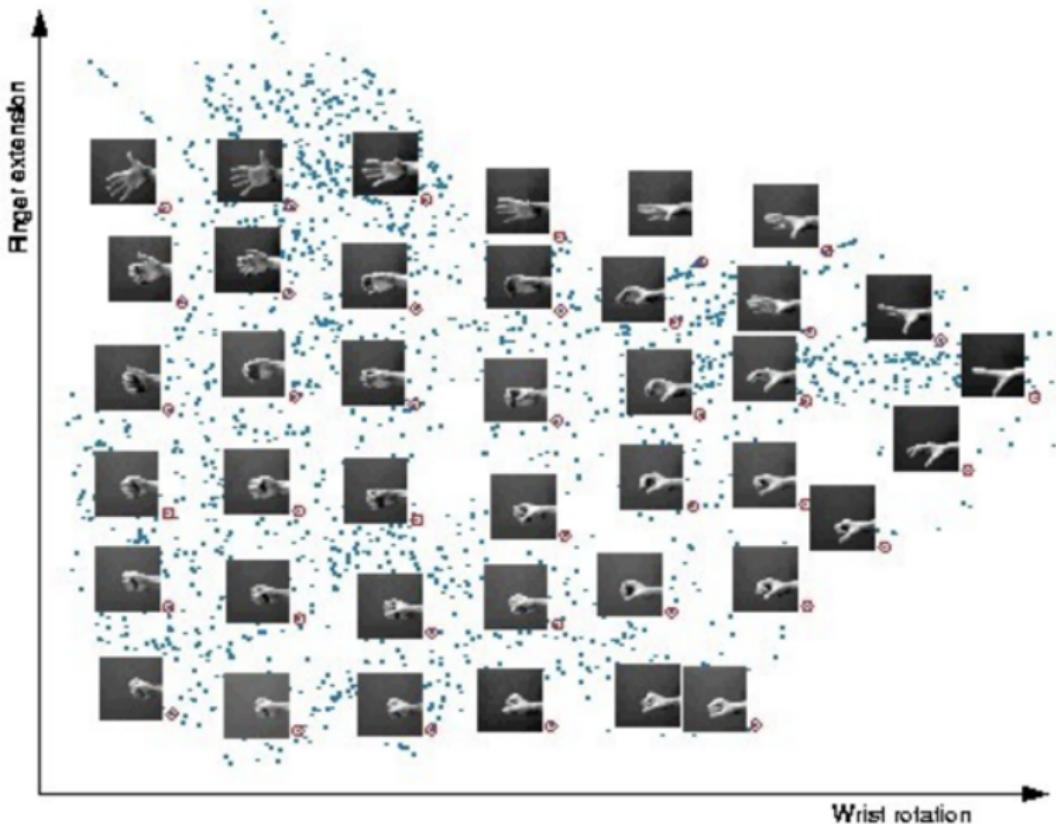
- Build a low- D embedded space to best preserve the complete distance matrix. Error function:

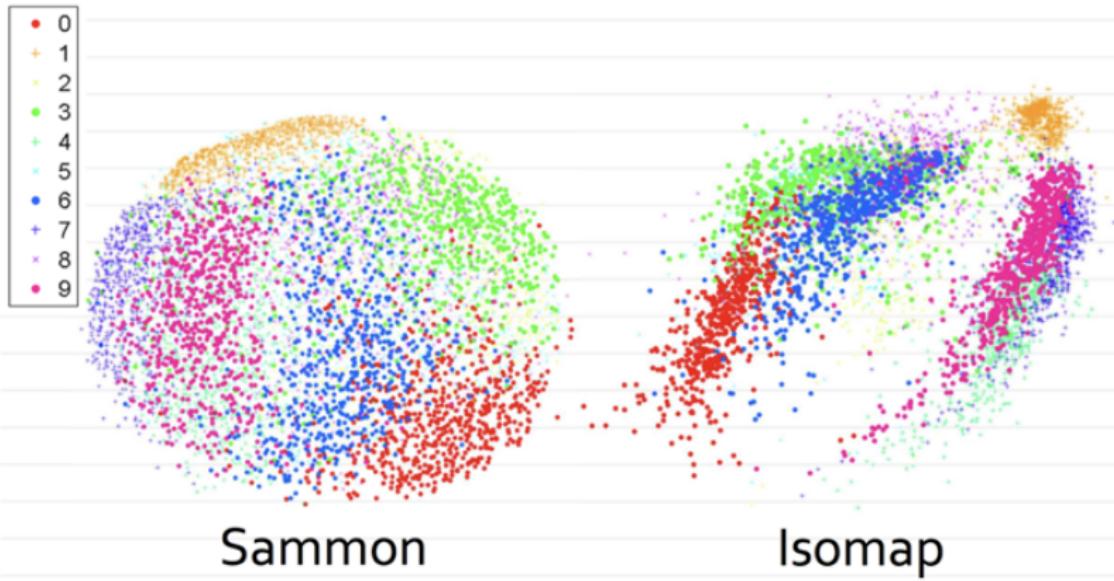
$$E = \|\tau(D_G) - \tau(D_Z)\|_{L^2}$$

where D_G — inner product distances in graph, D_Z — inner product distances in a new coordinate system

- Solution — set points Z to top eigenvectors D_g

ISOMAP results: hands





- preserves global structure
- few free parameters
- sensitive to noise, noise edges
- computationally expensive (dense matrix eigen-reduction)

1 Dimensionality Reduction: Problem Statement

2 Principal Component Analysis

3 Multi-Dimensional Scaling

4 Replicative Neural Networks

5 Graphs based on nearest neighbours

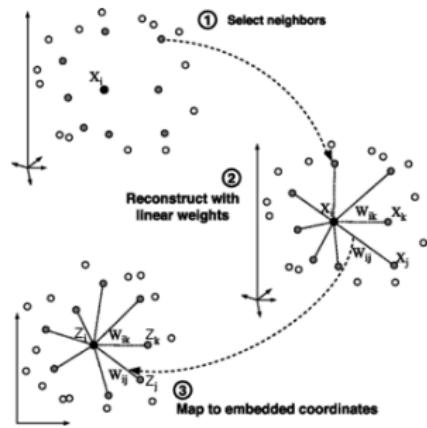
6 ISOMAP

7 Locally Linear Embedding

8 Stochastic Neighbour Embedding

Find a mapping to preserve
local linear relationships
between neighbors





- **Step 1.** k -nearest neighbours
- **Step 2.** Get “Baricentric” coordinates W_{ij} by minimizing

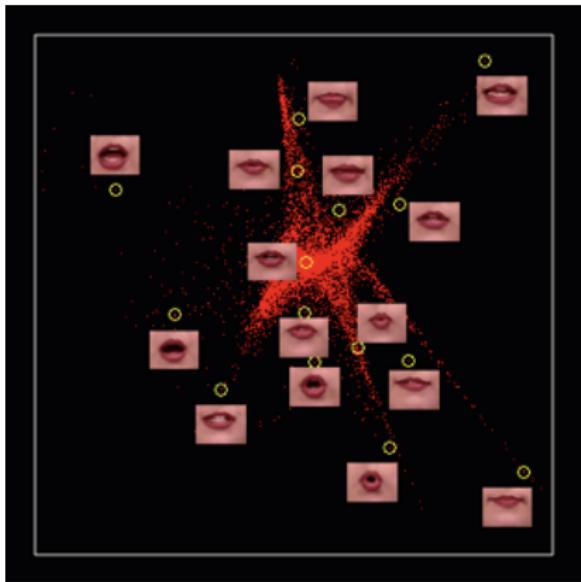
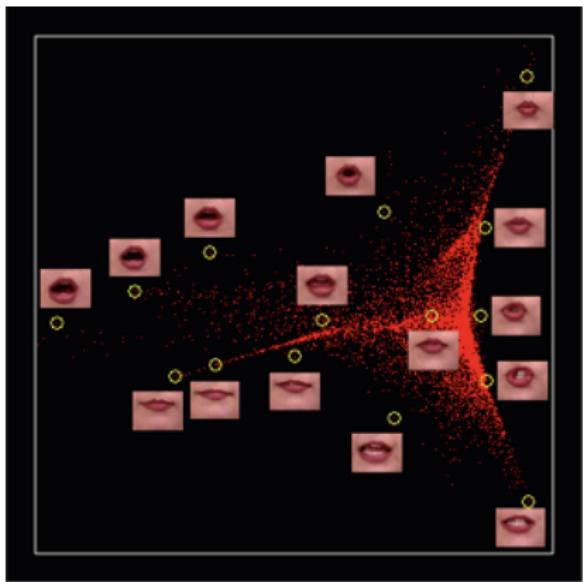
$$J_1(\mathbf{W}) = \sum_{i=1}^m \left\| \mathbf{x}_i - \sum_{j=1}^K W_{ji} \mathbf{x}_{j(i)} \right\|^2$$

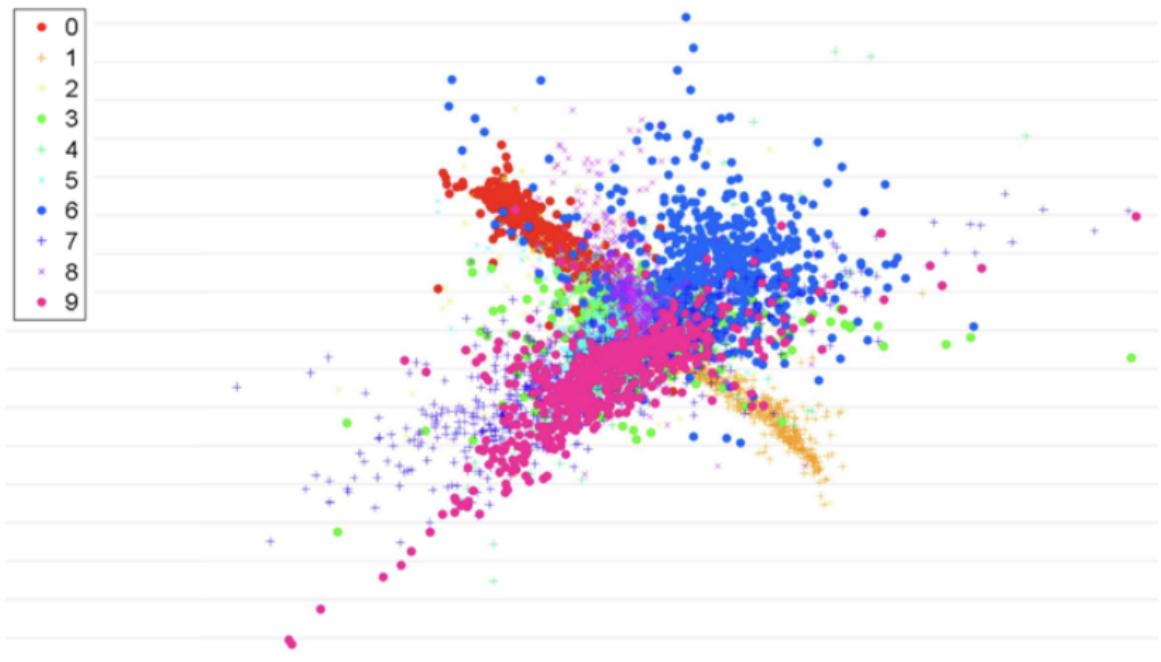
- Get $\mathbf{z}_1, \mathbf{z}_2, \dots, \mathbf{z}_m \in \mathbb{R}^q$ by minimizing

$$J_2(\mathbf{Z}_m) = \sum_{i=1}^m \left\| \mathbf{z}_i - \sum_{j=1}^m W_{ji} \mathbf{z}_{j(i)} \right\|^2$$

with standard normalizing conditions

LLE: Lips pictures





- no local minima, one free parameter
- incremental fast
- simple linear algebra operations
- can distort global structure

Not covered here:

- **Riemannian Manifold Learning, RML**
- **Local Tangent Space Alignment, LTSA**

- 1 Dimensionality Reduction: Problem Statement
- 2 Principal Component Analysis
- 3 Multi-Dimensional Scaling
- 4 Replicative Neural Networks
- 5 Graphs based on nearest neighbours
- 6 ISOMAP
- 7 Locally Linear Embedding
- 8 Stochastic Neighbour Embedding

- "Stochastic" similarity in the initial space

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

- Similarity in the compressed space

$$q_{j|i} = \frac{\exp(-\|\mathbf{z}_i - \mathbf{z}_j\|^2)}{\sum_{k \neq i} \exp(-\|\mathbf{z}_i - \mathbf{z}_k\|^2)}$$

- Cost function

$$\text{Cost} = \sum_i KL(P_i \parallel Q_i) = \sum_i \sum_j p_{j|i} \log \frac{p_{j|i}}{q_{j|i}}$$

$$\frac{\partial \text{Cost}}{\partial \mathbf{z}_i} = 2 \sum_j (p_{j|i} - q_{j|i} + p_{i|j} - q_{i|j})(\mathbf{z}_i - \mathbf{z}_j)$$

To escape poor minima:

- Initialize with Gaussian noise
- Add diminishing Gaussian noise on each stage

t-sne: Similarity in the compressed space in case of $t(1)$

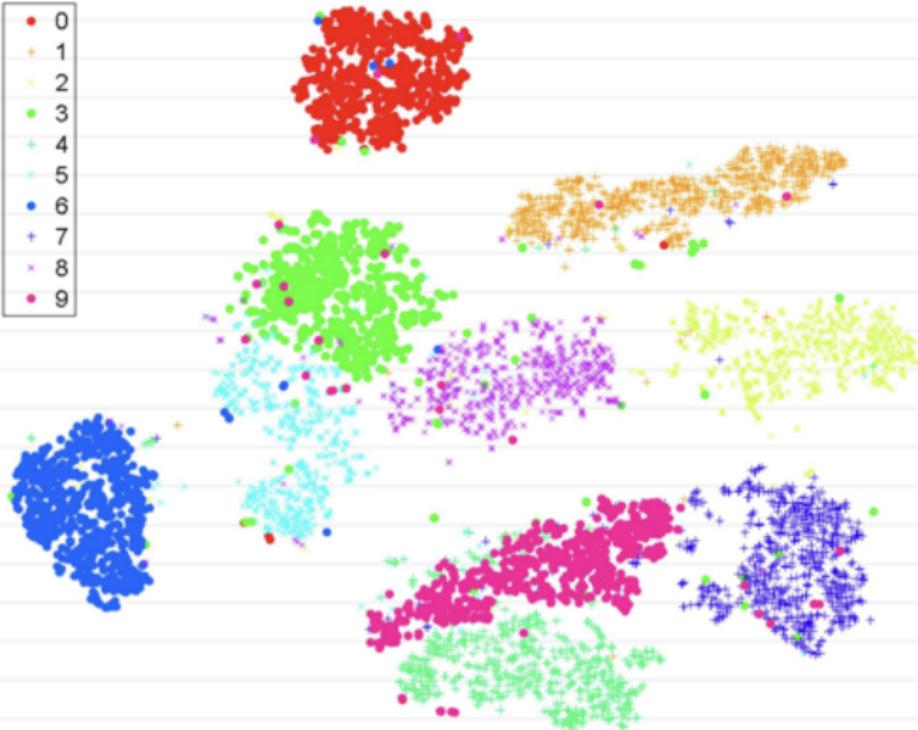
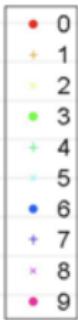
$$q_{j|i} = \frac{(1 + \|\mathbf{z}_i - \mathbf{z}_j\|^2)^{-1}}{\sum_{k \neq i} (1 + \|\mathbf{z}_k - \mathbf{z}_l\|^2)^{-1}}$$

t-sne: minimize KL between the whole densities P and Q

$$\text{Cost} = KL(P_i \parallel Q_i) = \sum_i \sum_j p_{ij} \log \frac{p_{ij}}{q_{ij}}$$

$$p_{ij} = \frac{p_{j\parallel i} + p_{i\parallel j}}{2n}$$

t-stochastic neighbor embedding: MNIST



The “curvier” your manifold, the denser your data must be

