

Computer Vision

CS308

Feng Zheng

SUSTech CS Vision Intelligence and Perception

Week 7



南方科技大学
SOUTHERN UNIVERSITY OF SCIENCE AND TECHNOLOGY



Content

- Brief Review
- Dimensionality Reduction
 - PCA
 - Manifold Learning
- Clustering
 - K-Means
 - Mean-Shift

Brief Review



Matching with Features

- Steps

- Detect feature points in both images
 - Find corresponding pairs
 - Use these pairs to align images
- } Previous Lecture





Fitting: Issues

- If **we know which points belong to the line**, how do we find the "optimal" line parameters?
 - Least squares
- What if there are **outliers**?
 - Robust fitting, RANSAC
- What if there are **many lines**?
 - Voting methods: RANSAC, Hough transform
- What if we're **not even sure** it's a line?
 - Model selection

Machine Learning



Machine Learning Problems

- Taxonomy

	<i>Supervised Learning</i>	<i>Unsupervised Learning</i>
<i>Discrete</i>	classification or categorization	clustering
<i>Continuous</i>	regression	<div>dimensionality reduction</div>

Dimensionality Reduction (Visualization)



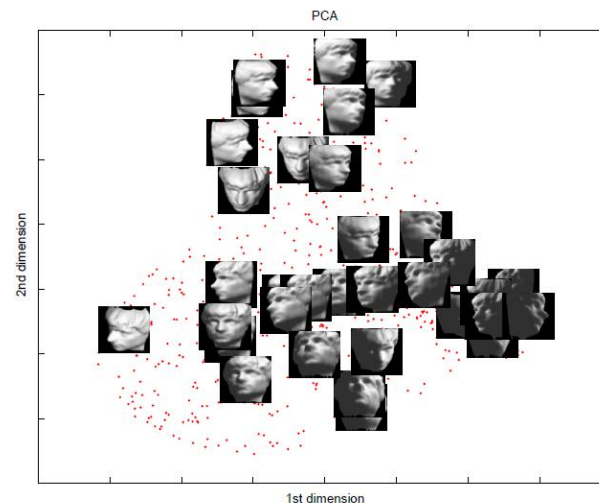
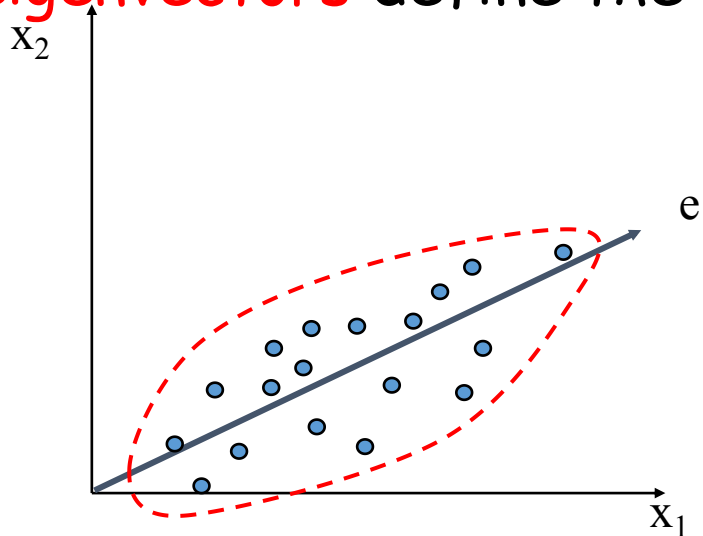
Dimensionality Reduction vs. Manifold Learning

- Primary methods
 - Linear methods
 - ✓ Principal component analysis (PCA)
 - ✓ Multidimensional scaling (MDS)
 - Nonlinear methods
 - ✓ Kernel PCA
 - ✓ Locally linear embedding (LLE)
 - ✓ Isomap
 - ✓ Laplacian eigenmaps (LE)
 - ✓ T-distributed stochastic neighbor embedding



Principal **Component** Analysis (PCA)

- History: Karl Pearson, 1901
- Goal:
 - Find projections that capture the largest amounts of **variation** in data
 - Find the eigenvectors of the **covariance matrix**, and these **eigenvectors** define the new space



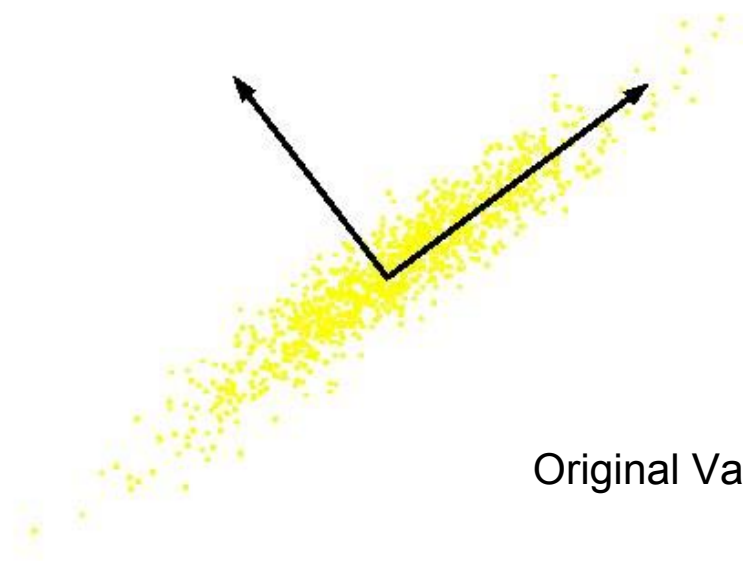
What is the original dimension of images?



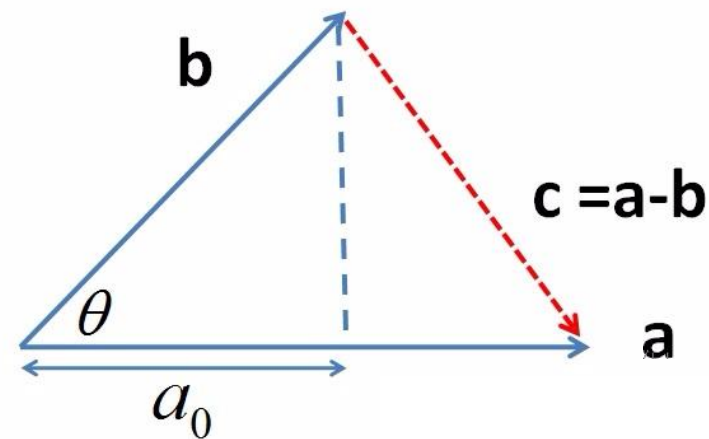
Principal Component Analysis (PCA)

- Definition:

- Given a set of data $X \in \mathbb{R}^{d \times N}$, find the principal axes are those **orthonormal** axes onto which the **variance** retained under projection is **maximal**



Original Variable A



$$a \bullet b = |a||b| \cos \theta$$



PCA: One Attribute First

- Question: how much spread is in the data along the **axis**? (distance to the mean)
- Variance = Standard deviation²

$$s^2 = \frac{\sum_{i=1}^n (X_i - \bar{X})^2}{(n - 1)}$$

Temperature
42
40
24
30
15
18
15
30
15
30
35
30
40
30



PCA: Now Consider Two Dimensions

- Covariance: measures the **correlation** between X and Y
- $cov(X, Y) = 0$: **independent**
- $cov(X, Y) > 0$: move same direction
- $cov(X, Y) < 0$: move opposition direction

90.81632653	57.14286
57.14285714	100

$$cov(X, Y) = \frac{1}{n} \sum_{i=1}^n (X_i - \bar{X})(Y_i - \bar{Y})$$

X=Temperature	Y=Humidity
40	90
40	90
40	90
30	90
15	70
15	70
15	70
30	90
15	70
30	70
30	70
30	90
40	70
30	90



Covariance Matrix: Similarity Between Variables

- Contains covariance values between all possible dimensions (=attributes):

$$C^{n \times n} = (c_{ij} \mid c_{ij} = \text{cov}(Dim_i, Dim_j))$$

- Example for three attributes (x, y, z):

$$S = \begin{pmatrix} \text{cov}(x, x) & \text{cov}(x, y) & \text{cov}(x, z) \\ \text{cov}(y, x) & \text{cov}(y, y) & \text{cov}(y, z) \\ \text{cov}(z, x) & \text{cov}(z, y) & \text{cov}(z, z) \end{pmatrix}$$



Formulation

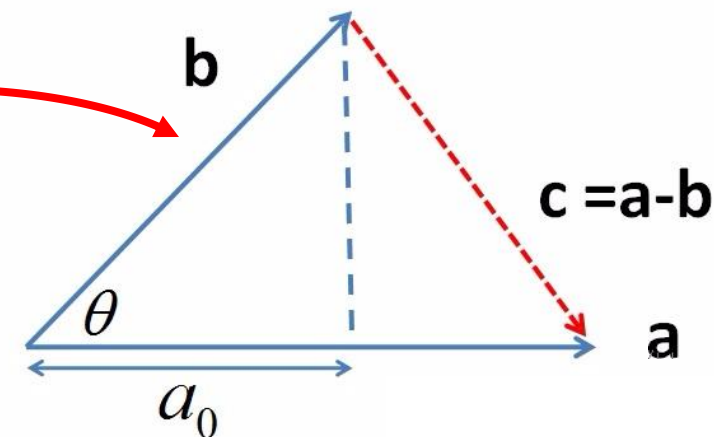
- Variance on the first (one) dimension

- $\text{var}(U_1) = \text{var}(\mathbf{w}^T X) = \mathbf{w}^T S \mathbf{w}$
- $S = X X^T$: covariance matrix of X

- Objective: the variance retains the maximal

- Formulation

$$\begin{aligned} \max_{\mathbf{w}} \quad & \mathbf{w}^T S \mathbf{w} \\ \text{s.t.} \quad & \mathbf{w}^T \mathbf{w} = 1 \end{aligned}$$



- Solving procedure

- Construct Lagrangian
- Set the partial derivative on to zero
- As $\mathbf{w} \neq \mathbf{0}$ then \mathbf{w} must be an eigenvector of S with eigenvalue λ_1

$$L(\mathbf{w}, \lambda_1) = \mathbf{w}^T S \mathbf{w} - \lambda_1 (\mathbf{w}^T \mathbf{w} - 1)$$

$$\frac{\partial L}{\partial \mathbf{w}} = 0 \Rightarrow S \mathbf{w} = \lambda_1 \mathbf{w}$$

$$\mathbf{w}^T S \mathbf{w} = \lambda_1 \mathbf{w}^T \mathbf{w} = \lambda_1$$



PCA: Another Interpretation

- A rank- k linear approximation model

$$X = f(\mathbf{y}) = \bar{\mathbf{x}} + U_k \mathbf{y}$$

- Fit the model with minimal **reconstruction error**

$$\min_{U_k, \mathbf{y}} \sum_{i=1}^N \|\mathbf{x}_i - U_k \mathbf{y}_i\|^2 \quad \text{suppose } \bar{\mathbf{x}} = \mathbf{0}$$

- Optimal condition

$$\frac{d}{d\mathbf{y}_i} = 0 \Rightarrow \mathbf{y}_i = U_k^T \mathbf{x}_i$$

- Objective

➤ Can be expressed as SVD of X

$$\min_{U_k} \sum_{i=1}^N \|\mathbf{x}_i - U_k U_k^T \mathbf{x}_i\|^2 \quad X = U \Sigma V^T$$

Σ

Diagonal matrix
of eigenvalues

<https://courses.cs.washington.edu/courses/cse446/17wi/slides/pca-annotated>. $\text{error}_K = N \sum_{j=K+1}^D \mathbf{u}_j^T \Sigma \mathbf{u}_j$



PCA: Algorithm

- Step 1: Covariance matrix
- Step 2: Eigenvector decomposition

Algorithm 1 Direct PCA Algorithm

Input: Given data $\mathcal{D} = \{\mathbf{x}_i\}_{i=1}^N$, $\mathbf{x}_i \in \mathbb{R}^d$;

Recover basis: Calculate $XX^\top = \sum_{i=1}^N \mathbf{x}_i \mathbf{x}_i^\top$ and U as eigenvectors of XX^\top for the top k eigenvalues.

Encode training data: $Y = U^\top X$, where Y is a $k \times N$ matrix of encodings of the original data.

Reconstruct training data: $\hat{X} = UY = UU^\top X$.

Encode test data: $y = U^\top x$, where y is a k -dimensional encoding of x .

Reconstruct test data: $\hat{x} = Uy = UU^\top x$.



Kernel Function: Similarity Between Samples

- Map the data into higher dimensional spaces: the data could become more easily separated or better structured
 - Support vector machine (SVM) -> Nonlinear SVM
 - Principal component analysis -> Kernel PCA

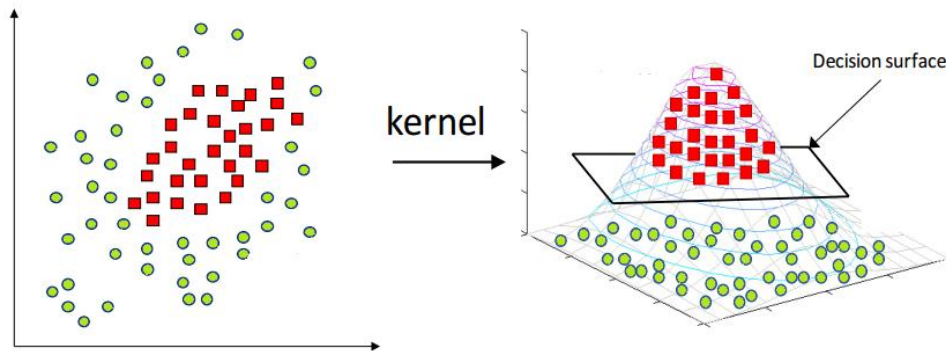
$$k(x,y) = \langle \Phi(x), \Phi(y) \rangle \quad \Phi : x \rightarrow \mathcal{H} \quad x \mapsto \Phi(x)$$

- Must be continuous, symmetric, and most preferably should have a positive (semi-) definite **Gram** matrix

- Kernel Functions

- Linear Kernel
- Polynomial Kernel
- Gaussian Kernel

$$k(x,y) = x^T y + c$$
$$k(x,y) = (\alpha x^T y + c)^d$$
$$k(x,y) = \exp\left(-\frac{\|x - y\|^2}{2\sigma^2}\right)$$





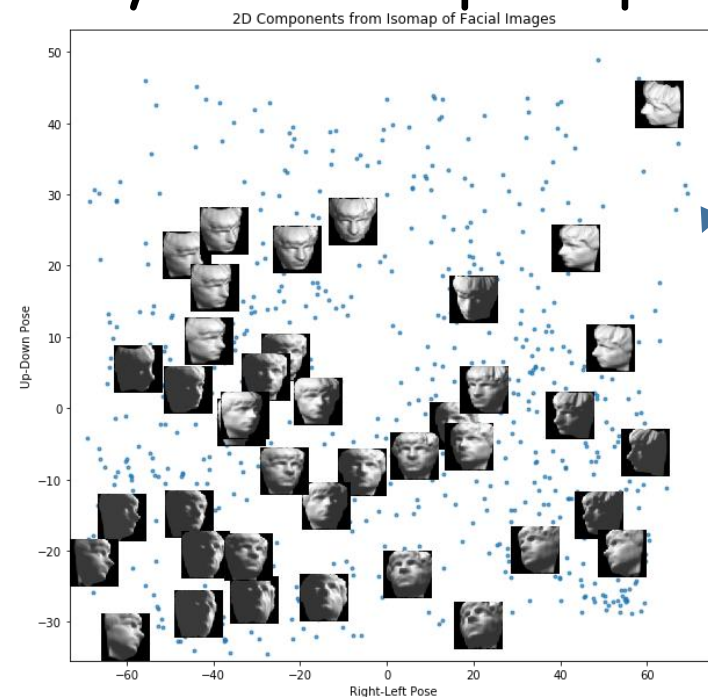
Kernel PCA

- History: S. Mika et al, NIPS, 1999
- Data may lie on or near a nonlinear manifold, not a linear subspace
- Find principal components that are nonlinearly to the input space via nonlinear mapping $\Phi : x \rightarrow \mathcal{H} \quad x \mapsto \Phi(x)$

- Objective

$$\min_{U_k} \sum_{i=1}^N \left\| \Phi(\mathbf{x}_i) - U_k U_k^T \Phi(\mathbf{x}_i) \right\|^2$$

- Solution found by SVD: $\Phi(X) = U \Sigma V^T$
 U contains the eigenvectors of $\Phi(X) \Phi(X)^T$





Kernel PCA

- Centering

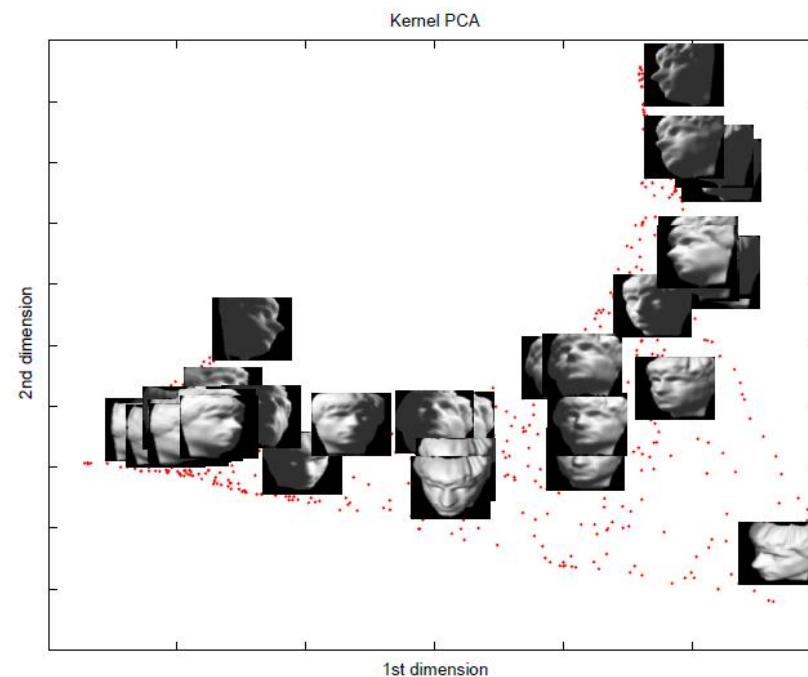
$$\tilde{\Phi}(X) = \Phi(X) - E_x[\Phi(X)]$$

x, y both are the samples not the variables

$$\tilde{K}(x, y) = \tilde{\Phi}(x)\tilde{\Phi}(y)$$

$$\begin{aligned}\tilde{K}(x, y) &= (\Phi(x) - E_x[\Phi(x)])(\Phi(y) - E_y[\Phi(y)]) \\ &= K(x, y) - E_x[K(x, y)] - E_y[K(x, y)] + E_x[E_y[K(x, y)]]\end{aligned}$$

- Issue: Difficult to calculate $\Phi(X)\Phi(X)^T$
 - Using $\tilde{K}(x, y)$ to calculate the eigenvectors





Two Matrices

$$X = \begin{pmatrix} \text{---} \\ \text{---} \\ \text{---} \\ \text{---} \\ \text{---} \\ \text{---} \\ \text{---} \\ \text{---} \\ \text{---} \\ \text{---} \end{pmatrix}_{n \times D}$$

1. Gram Matrix (Sample correlation matrix)

$$K = (XX^T)_{n \times n} \quad K\mu_i^T = \tau_i\mu_i^T \quad \text{where } i = \{1, 2, \dots, n\}$$

$$K = (XX^T)_{n \times n} = \left(I - \frac{1}{n} \mathbf{1}_n^T \mathbf{1}_n \right) E_X \left(I - \frac{1}{n} \mathbf{1}_n^T \mathbf{1}_n \right)$$

where $E_X(i, j) = d_{ij}$

Similarity
Between
Samples

2. Covariance Matrix

$$C = \frac{1}{n} (X^T X)_{D \times D} = \frac{1}{n} \sum_i x_i^T x_i$$

$$Cv_i^T = \lambda_i v_i^T \quad \text{where } i = \{1, 2, \dots, D\} \text{-----(0)}$$

(a): $\lambda_i v_i^T = \frac{1}{n} \sum_j x_j^T \langle x_j, v_i \rangle, \quad \text{where } \lambda_i \neq 0$

Similarity
Between
Variables



Two Matrices

1. Relationship

- Existing coefficients: $v = \sum_{j=1}^n \alpha(j) x_j$
- For all samples x_k : $\lambda x_k v^T = x_k C v^T$ -----(1)

$$\lambda x_k \sum_{j=1}^n \alpha(j) x_j^T = x_k \left(\frac{1}{n} \sum_i x_i^T x_i \right) \sum_{j=1}^n \alpha(j) x_j^T$$
 -----(2)

- If set $K_{ij} = \langle x_i, x_j \rangle$,

$$n \lambda K \alpha = K^2 \alpha$$
 -----(3)

$$n \lambda \alpha = K \alpha$$
 -----(4)

- Conclusion:

(b): $\alpha_i = X v_i^T = \sqrt{\lambda_i} \mu_i$; (c): $n \lambda_i = \tau_i$;

(d): $v_i x^T = \sum_{j=1}^n \alpha_i(j) x_j x^T$ (x is a new sample)

The required projection

Inner product

$$X = \begin{pmatrix} \text{---} \\ \text{---} \\ \text{---} \\ \text{---} \\ \text{---} \end{pmatrix}_{n \times D}$$

(a): $\lambda_i v_i^T = \frac{1}{n} \sum_j x_j^T \langle x_j, v_i \rangle$

$\alpha_i(j)$

For Kernel PCA:

What do we know? Kernel

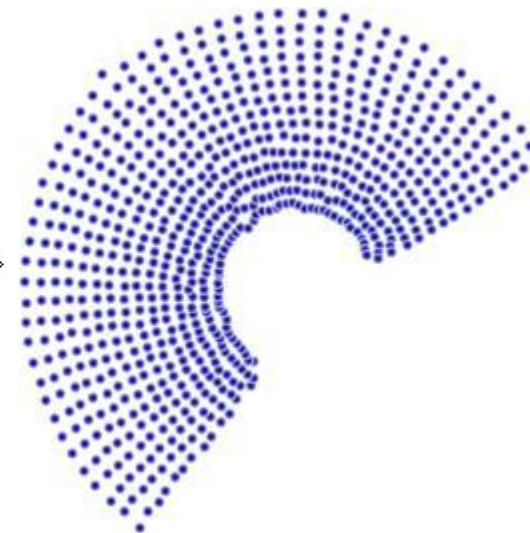
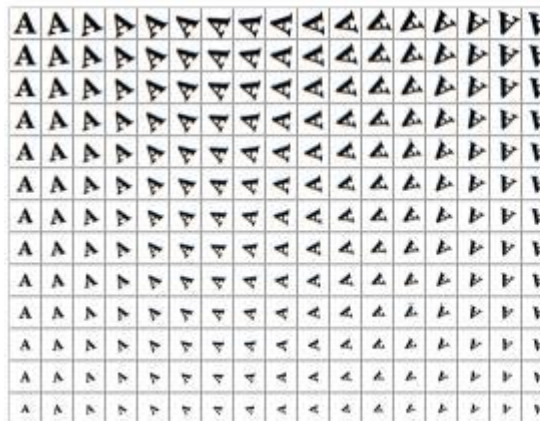
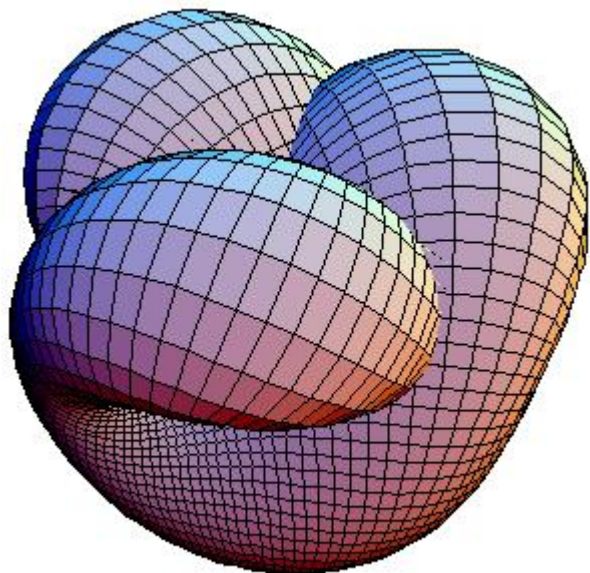
What do we not know? Covariance

Manifold Learning



Manifold \rightarrow Graph

- In mathematics, a **manifold** is a topological space that locally resembles Euclidean space near each point

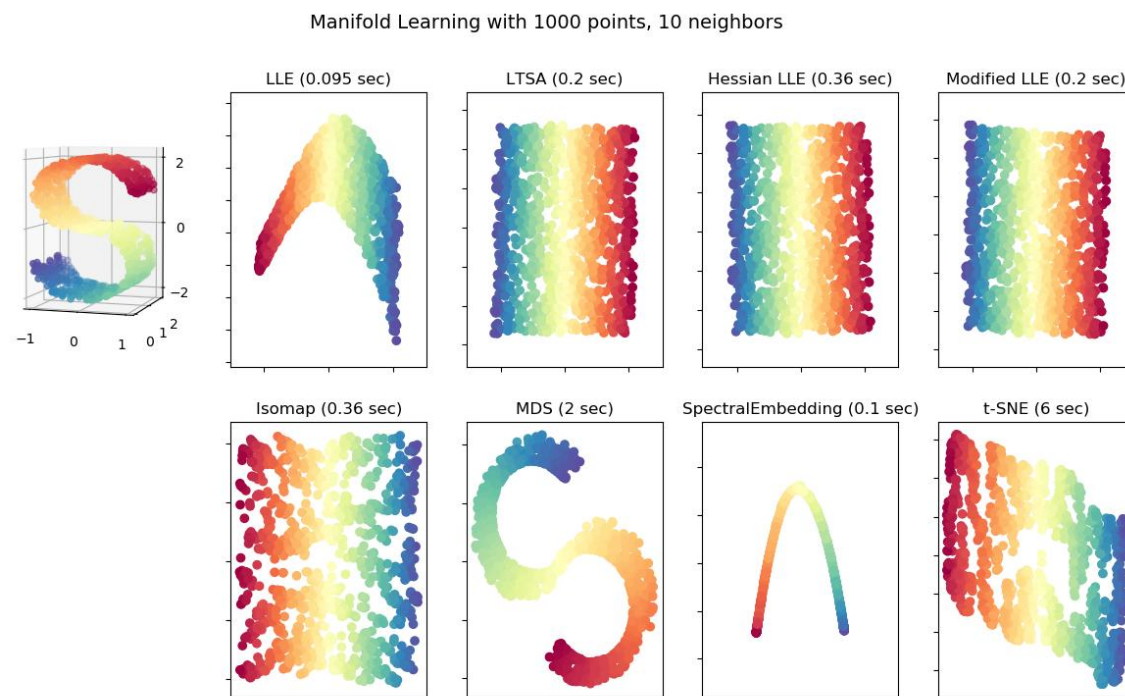


Plot of the two-dimensional points that results from using a NLDR algorithm. In this case, Manifold Sculpting used to reduce the data into just two dimensions (**rotation and scale**).



Nonlinear Dimensionality Reduction

- **High-dimensional** data, meaning data that requires more than two or three dimensions to represent, can be **difficult** to interpret.
- One approach to **simplification** is to assume that the data of interest lie on an **embedded non-linear manifold** within the higher-dimensional space.
- If the **manifold** is of **low** enough dimension, the data can be **visualised** in the **low-dimensional** space.





Locally Linear Embedding (LLE)

- History: S. Roweis and L. Saul, *Science*, 2000
- Procedure
 - Identify the **neighbors** of each data point
 - Compute weights that best **linearly reconstruct the point** from its **neighbors**

$$\min_{\mathbf{w}} \sum_{i=1}^N \left\| \mathbf{x}_i - \sum_{j=1}^k w_{ij} \mathbf{x}_{N_i(j)} \right\|^2$$

Locally

- Find the **low-dimensional embedding vector** which is best reconstructed by the weights determined in Step 2

$$\min_Y \sum_{i=1}^N \left\| \mathbf{y}_i - \sum_{j=1}^k w_{ij} \mathbf{y}_{N_i(j)} \right\|^2 \iff \min_Y \text{tr}(\mathbf{Y}^\top \mathbf{Y} \mathbf{L}) \quad \text{Centering Y with unit variance}$$

where $\mathbf{L} = \mathbf{R} - \mathbf{W}$, \mathbf{R} is diagonal and $R_{ii} = \sum_{j=1}^N W_{ij}$.

<https://cs.nyu.edu/~roweis/lle/papers/lleintro.pdf>



Laplacian Eigenmaps (LE)

- History: M. Belkin and P. Niyogi, 2003
- Similar to locally linear embedding
- **Different in weights** setting and objective function

➤ Weights

$$W_{ij} = \begin{cases} 1 & i, j \text{ are connected} \\ \exp\left(\frac{-\|x_i - x_j\|^2}{s}\right) & \text{otherwise} \end{cases}$$

Locally

➤ Objective

Has a different meaning to the weights in LLE

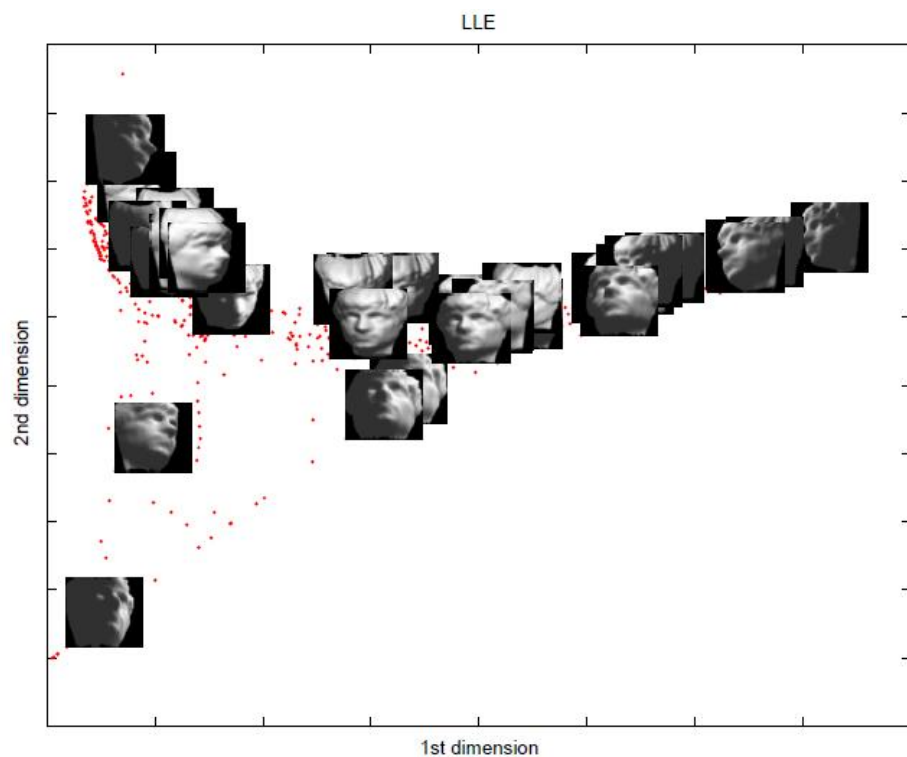
$$\min_Y \sum_{i=1}^N \sum_{j=1}^N (y_i - y_j)^2 W_{ij} \iff \min_Y \text{tr}(YLY^\top)$$

where $L = R - W$, R is diagonal and $R_{ii} = \sum_{j=1}^N W_{ij}$.

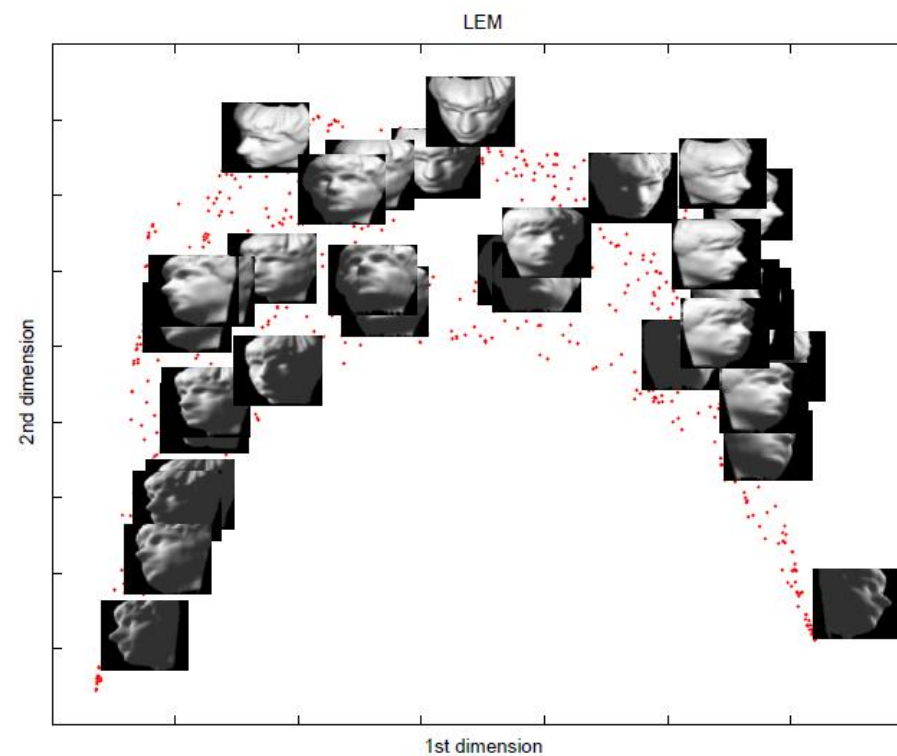


LLE and LE Examples

- Two-dimensional visualization



LLE



LE

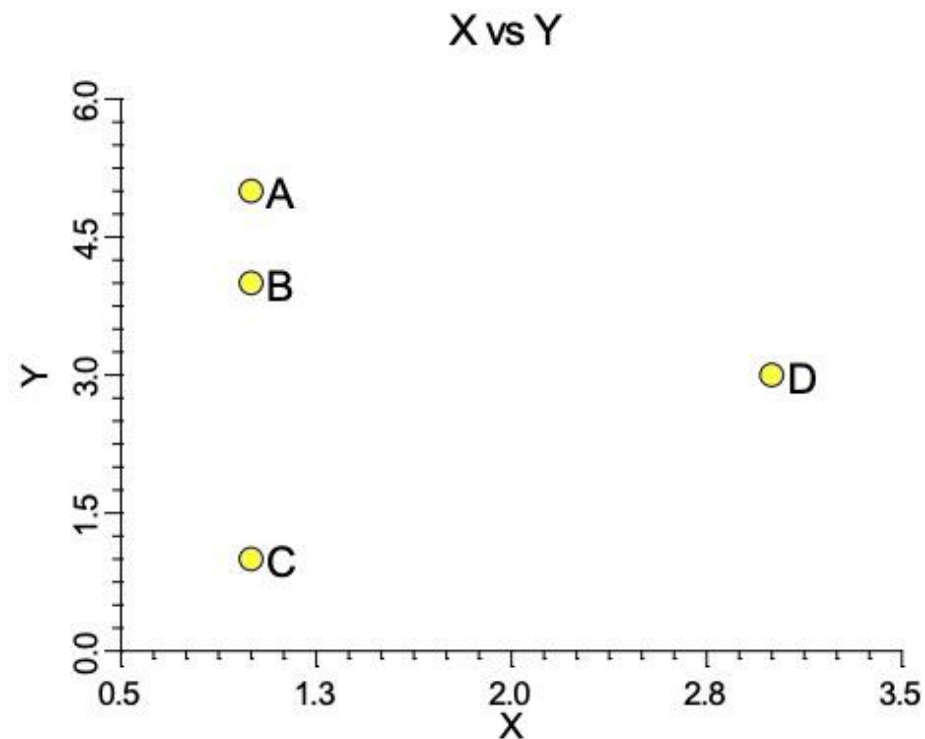


Multidimensional Scaling (MDS)

- The following example will help explain what MDS does. Consider the following set of data

Original Data Matrix

Label	X	Y
A	1	5
B	1	4
C	1	1
D	3	3

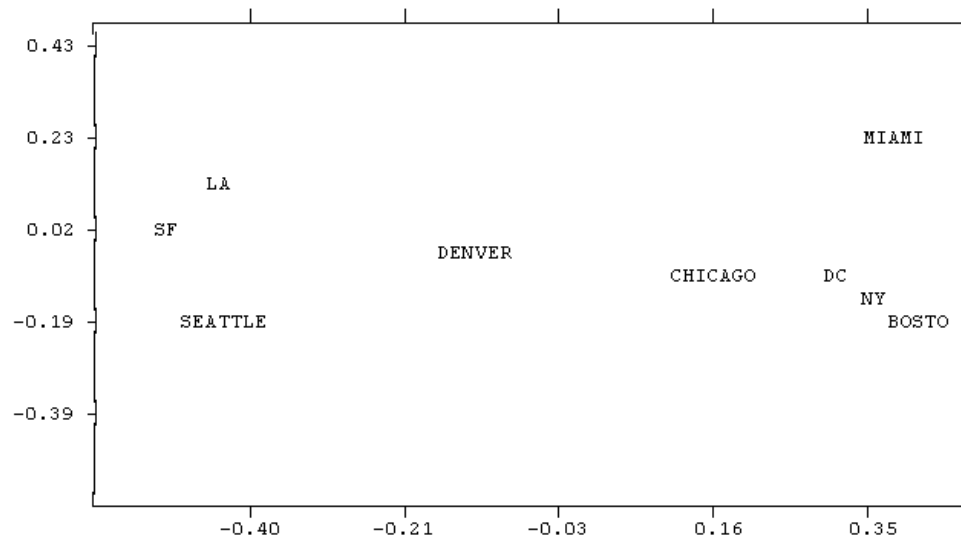




Multidimensional Scaling (MDS)

- Given the matrix of distances among cities, MDS produces this map

		1	2	3	4	5	6	7	8	9
		BOST	NY	DC	MIAM	CHIC	SEAT	SF	LA	DENV
1	BOSTON	0	206	429	1504	963	2976	3095	2979	1949
2	NY	206	0	233	1308	802	2815	2934	2786	1771
3	DC	429	233	0	1075	671	2684	2799	2631	1616
4	MIAMI	1504	1308	1075	0	1329	3273	3053	2687	2037
5	CHICAGO	963	802	671	1329	0	2013	2142	2054	996
6	SEATTLE	2976	2815	2684	3273	2013	0	808	1131	1307
7	SF	3095	2934	2799	3053	2142	808	0	379	1235
8	LA	2979	2786	2631	2687	2054	1131	379	0	1059
9	DENVER	1949	1771	1616	2037	996	1307	1235	1059	0



- We may find the $N \times N$ Gram matrix $B = X^T X$, rather than X .

The solutions are not unique



Multidimensional Scaling (MDS)

- History: T. Cox and M. Cox, 2001
- Goal: attempts to preserve **pairwise distances**

$$\min_Y \sum_{i=1}^N \sum_{j=1}^N (d_{ij}^{(X)} - d_{ij}^{(Y)})^2$$

Distance

where $d_{ij}^{(X)} = \|x_i - x_j\|^2$ and $d_{ij}^{(Y)} = \|y_i - y_j\|^2$.

- Different formulation of PCA, but **yields similar result** form
- Transformation

Proximity matrix

Gram matrix B

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

where $H = I - \frac{1}{N} \mathbf{1}\mathbf{1}^T$.

➤ Is equivalent to:

$$\min_Y \sum_{i=1}^N \sum_{j=1}^N (x_i^T x_j - y_i^T y_j)^2$$

Inner product

<http://fourier.eng.hmc.edu/e176/lectures/MultidimensionScaling.pdf>

<https://www.sjsu.edu/faculty/guangliang.chen/Math253S20/lec9md>

pdf



Multidimensional Scaling (MDS)

- Steps of a Classical MDS algorithm:

- Set up the squared proximity matrix

- Apply double centering

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

- Determine the **largest** k eigenvalues and corresponding eigenvectors

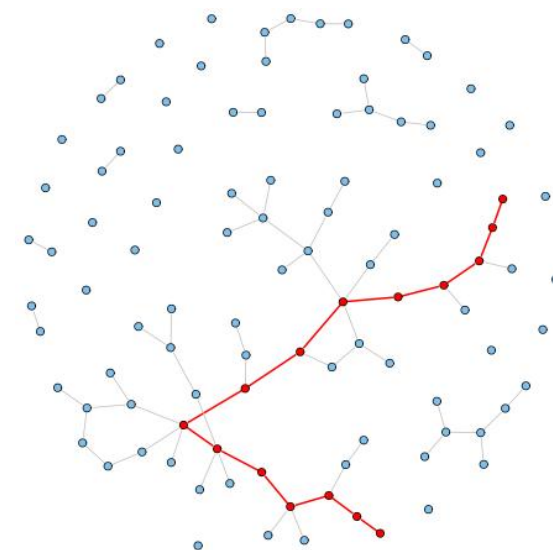
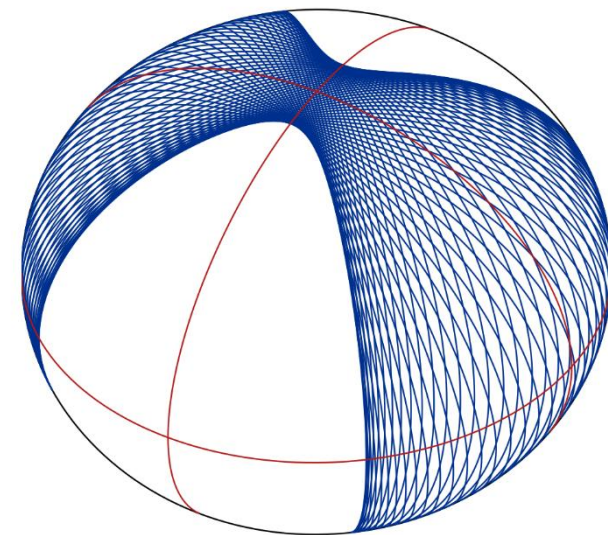
- The original coordinate is $X = \Lambda^{1/2} V'$, if we have had

- The NEW coordinate is $X_k = \Lambda_k^{1/2} V'_k$



Isomap

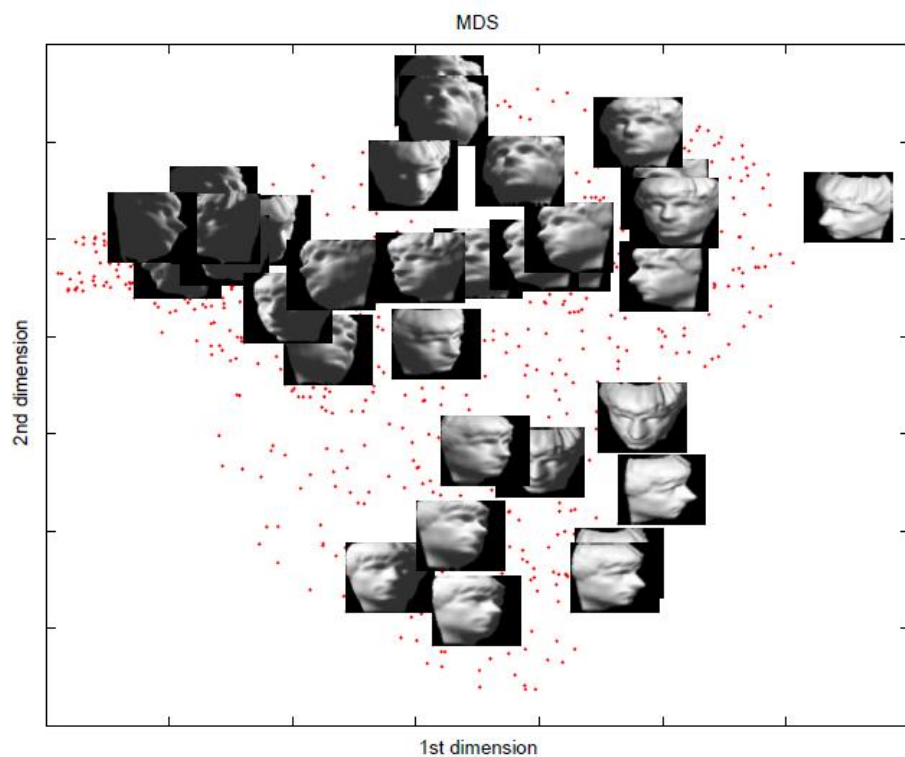
- History: J. Tenenbaum et al, Science 2000
 - A nonlinear generalization of classical MDS
 - Perform MDS, not in the original space, but in the **geodesic space**
- Procedure-similar to LLE
 - Find **neighbors** of each data point - graph
 - Compute geodesic pairwise distances (e.g., **shortest path distance**) between all points
 - **Embed the data** via MDS



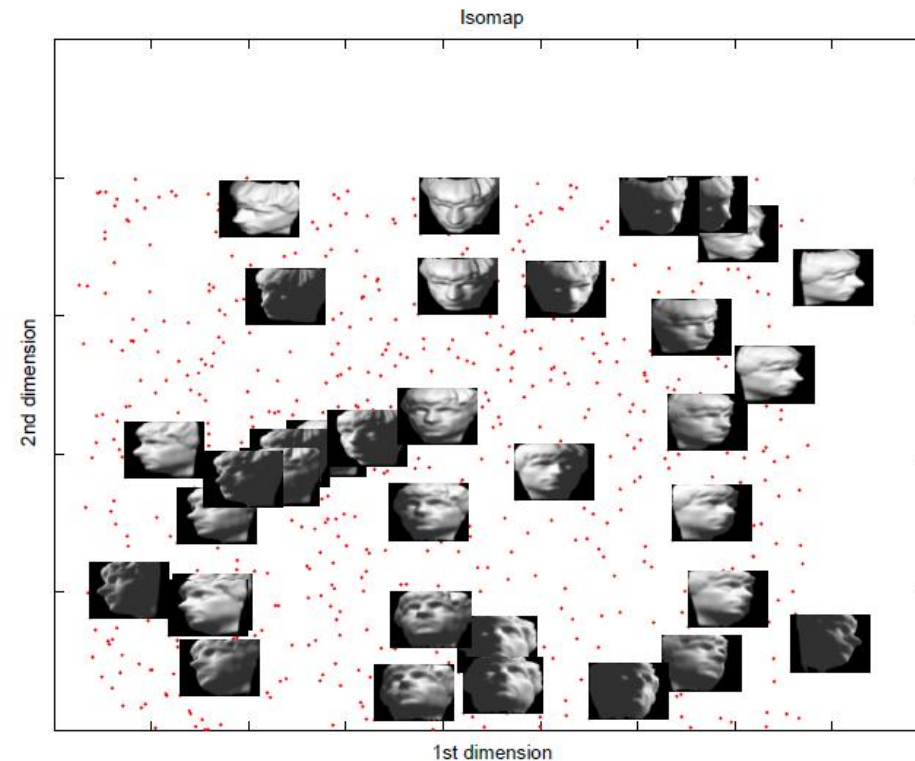


MDS and Isomap Example

- Two-dimensional visualization



MDS



Isomap



Intrinsic of Manifold Learning

- Preserve the local similarities (smoothness)

Manifold \rightarrow graph



Revisiting PCA

- Maximizing the variance
=
- Minimizing the reconstruction error
=
- Preserving the similarities or distances (classical MDS)
- OTHERS
 - Local reconstruction error (LLE)
 - Local similarities (LE)



Stochastic Neighbor Embedding

- The **similarity** of data point x_j to data point x_i is the conditional probability: $p_{j|i}$

$$p_{j|i} = \frac{\exp(-\|x_i - x_j\|^2 / 2\sigma_i^2)}{\sum_{k \neq i} \exp(-\|x_i - x_k\|^2 / 2\sigma_i^2)}$$

The relationships
are only related to
point i

- For the **low-dimensional** counterparts, a similar conditional **probability** is defined as: $q_{j|i}$

$$q_{j|i} = \frac{\exp(-\|y_i - y_j\|^2)}{\sum_{k \neq i} \exp(-\|y_i - y_k\|^2)}$$

What is preserved? **Similarity distribution**



Stochastic Neighbor Embedding

- SNE **minimizes** the sum of **Kullback-Leibler divergences** over all data points using a **gradient descent method**. The cost function C is given by

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

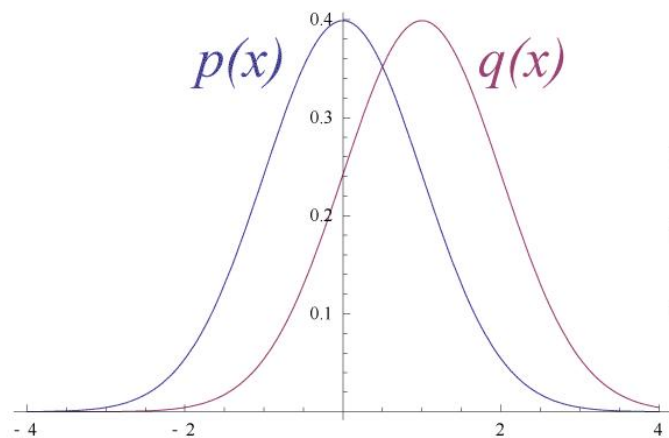
- P_i : conditional probability distribution over all others given x_i
- Q_i : conditional probability distribution over all other map points given map point y_i
- The gradient has a surprisingly simple form

$$\frac{\delta C}{\delta y_i} = 2 \sum_j (p_{j|i} - q_{j|i} + p_{i|j} - q_{i|j})(y_i - y_j)$$

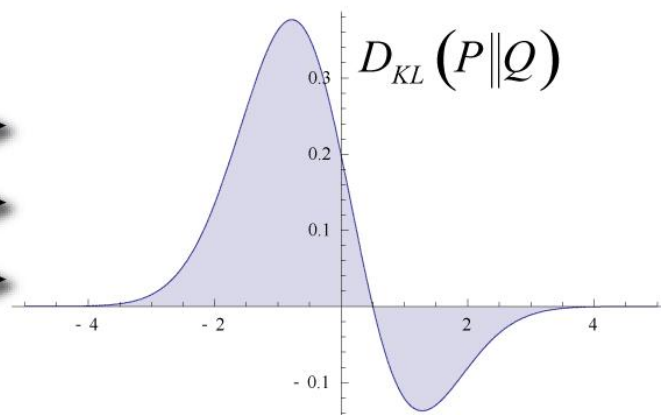
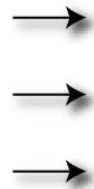


Kullback-Leibler Divergences

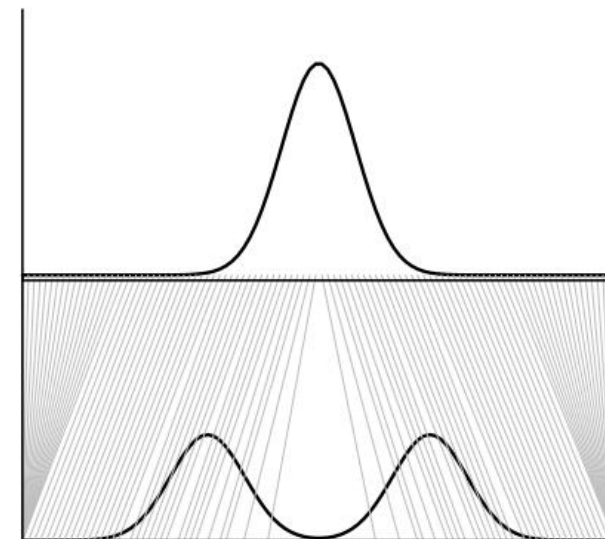
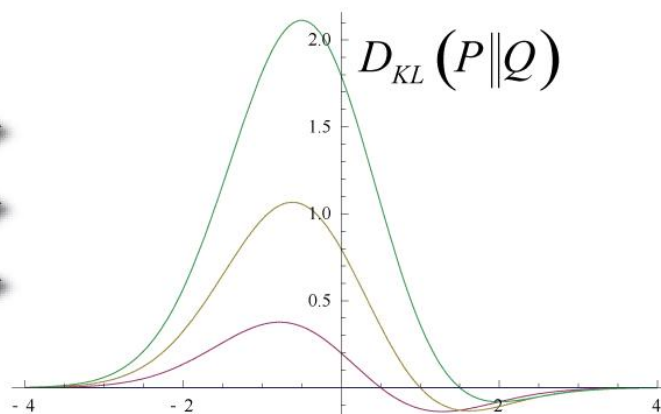
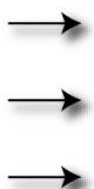
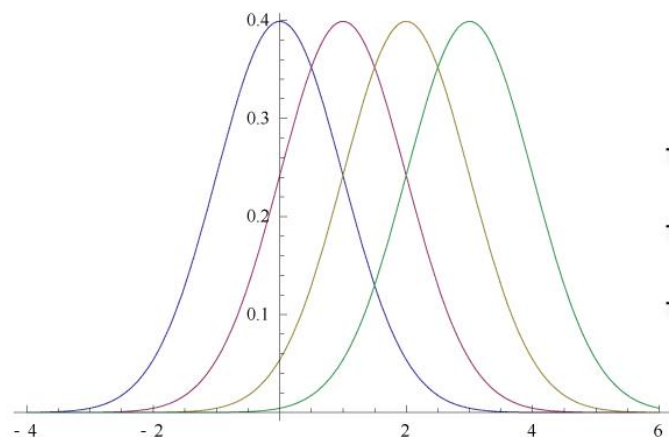
$$D_{\text{KL}}(P \parallel Q) = \int_{-\infty}^{\infty} p(x) \log \left(\frac{p(x)}{q(x)} \right) dx$$



Original Gaussian PDF's



KL Area to be Integrated



Wasserstein distance
Kantorovich–Rubinstein metric
Earth Mover's Distance



Symmetric SNE

- In **symmetric** SNE, the pairwise similarities in the low-dimensional map is

$$q_{ij} = \frac{\exp(-\|y_i - y_j\|^2)}{\sum_{k \neq l} \exp(-\|y_k - y_l\|^2)} \quad \text{All points}$$

- The pairwise similarities in the high-dimensional space is:

$$p_{ij} = \frac{\exp(-\|x_i - x_j\|^2 / 2\sigma^2)}{\sum_{k \neq l} \exp(-\|x_k - x_l\|^2 / 2\sigma^2)}$$

- The **gradient** of symmetric SNE is fairly similar to that of asymmetric SNE

$$\frac{\delta C}{\delta y_i} = 4 \sum_j (p_{ij} - q_{ij})(y_i - y_j)$$



T-distributed Stochastic Neighbor Embedding (T-SNE)

- The crowding problem
 - The **area** of the two-dimensional map that is available to accommodate moderately distant data points will not be nearly **large enough** compared with the area available to accommodate **nearby** data points
 - For example, it is possible to have 11 data points that are mutually equidistant in a ten-dimensional manifold but it is not possible to model this faithfully in a two-dimensional map. Therefore, if the **small distances** can be modeled accurately in a map, most of the **moderately distant data points** will be **too far away** in the two-dimensional map



T-distributed Stochastic Neighbor Embedding (T-SNE)

- Employ a **Student t-distribution** with one degree of freedom

$$\nu=1$$

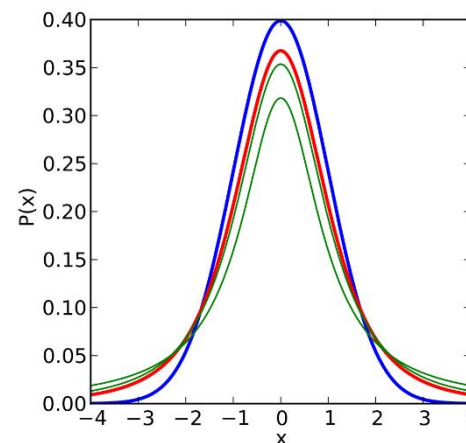
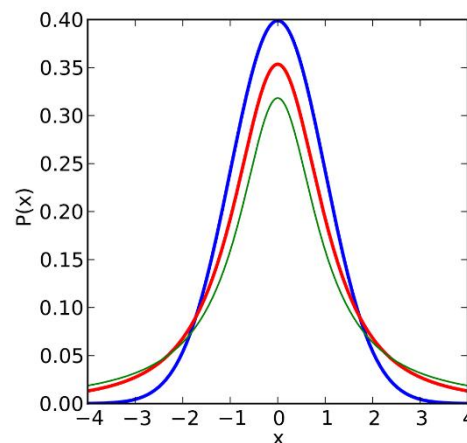
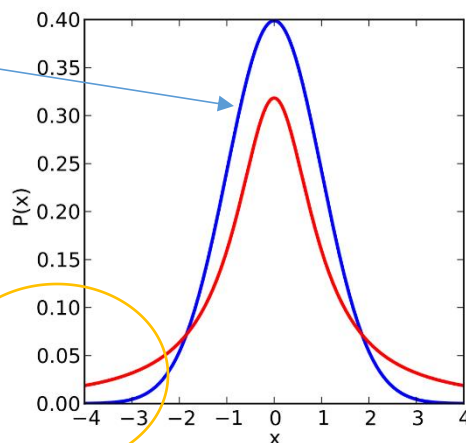
$$q_{ij} = \frac{(1 + \|y_i - y_j\|^2)^{-1}}{\sum_{k \neq l} (1 + \|y_k - y_l\|^2)^{-1}}$$

$$f(t) = \frac{\Gamma(\frac{\nu+1}{2})}{\sqrt{\nu\pi}\Gamma(\frac{\nu}{2})} \left(1 + \frac{t^2}{\nu}\right)^{-\frac{\nu+1}{2}}$$

- The gradient of the Kullback-Leibler divergence

$$\frac{\delta C}{\delta y_i} = 4 \sum_j (p_{ij} - q_{ij})(y_i - y_j) (1 + \|y_i - y_j\|^2)^{-1}$$

Gaussian



Density of the t-distribution (red) for 1, 2, 3 degrees of freedom compared to the standard normal distribution (blue)

When distances lose the ability to discriminate



Gradient Descent Method

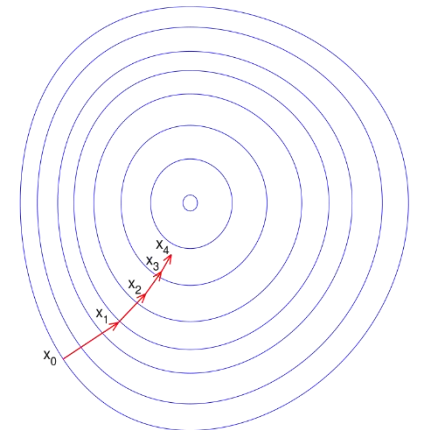
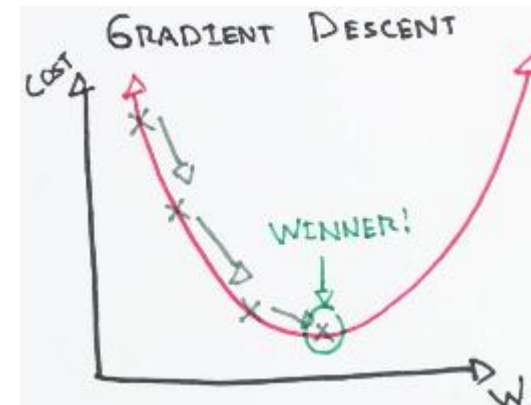
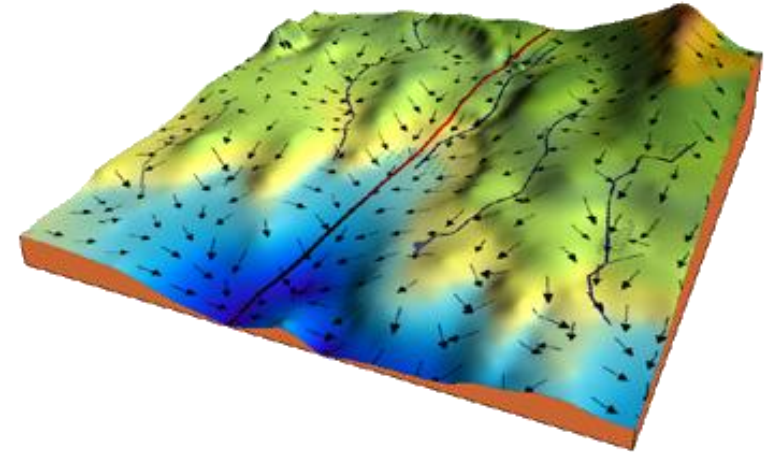
- Hypothesis space: linear function (m, b)
- Given the cost function

$$f(m, b) = \frac{1}{N} \sum_{i=1}^n (y_i - (mx_i + b))^2$$

- Gradient descent

$$f'(m, b) = \begin{bmatrix} \frac{df}{dm} \\ \frac{df}{db} \end{bmatrix} = \begin{bmatrix} \frac{1}{N} \sum -2x_i(y_i - (mx_i + b)) \\ \frac{1}{N} \sum -2(y_i - (mx_i + b)) \end{bmatrix}$$

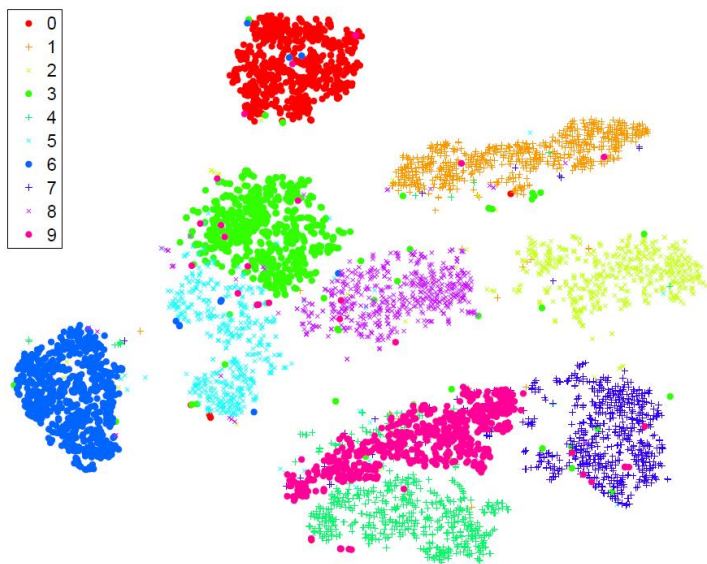
- Types of Gradient Descent:
 - **Batch** Gradient Descent
 - **Stochastic** Gradient Descent



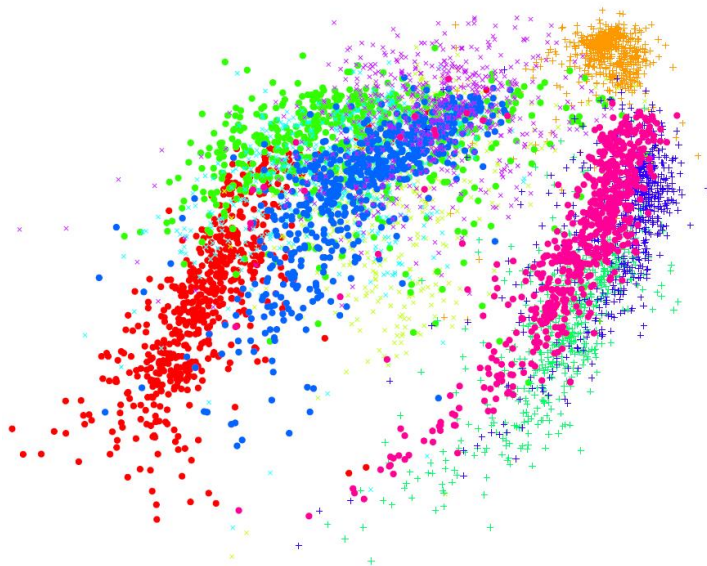


2D Visualization

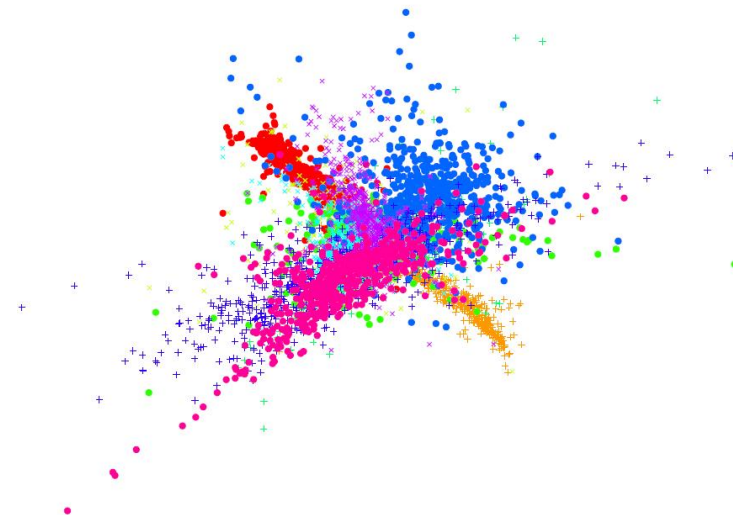
- Comparison



T-SNE



Isomap



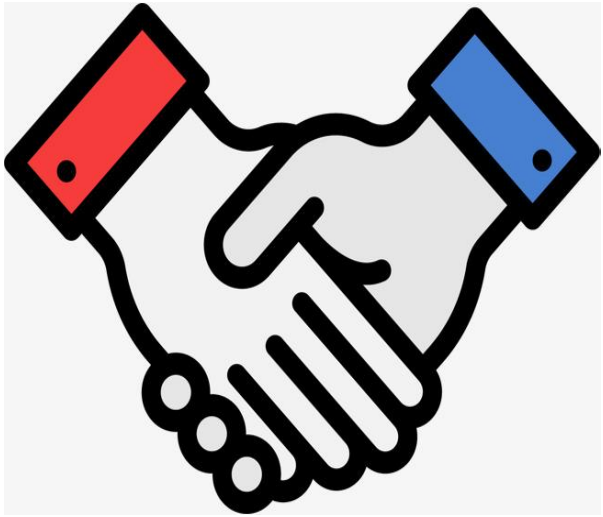
LLE

Conclusions



Conclusion

- Dimensionality Reduction
 - Linear
 - ✓ PCA
 - ✓ MDS
 - Manifold Learning (Nonlinear)
 - ✓ LLE
 - ✓ LE
 - ✓ Isomap
 - ✓ T-SNE



Thanks



zhengf@sustc.edu.cn