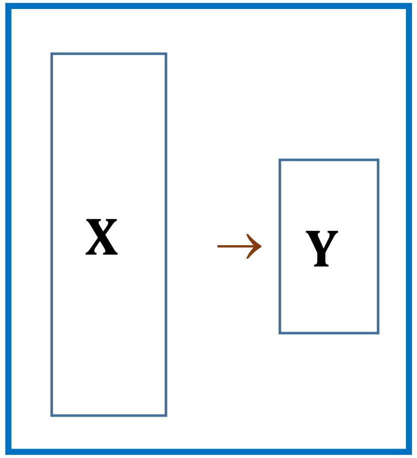
Lecture 8:

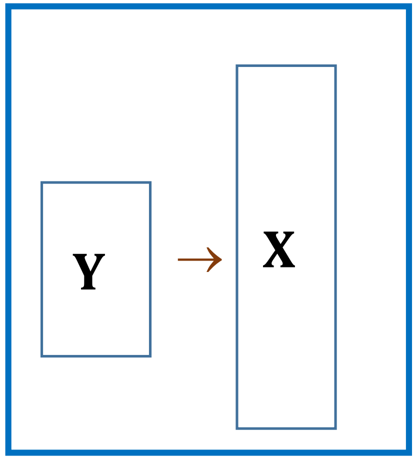
Manifold learning (1)

1. **Manifold model of high-dimensional data**
2. **Locally Linear Embedding (LLE) algorithm**
3. **ISOmetric MAPping (ISOMAP) algorithm**
4. **Out-of-Sample Extension through Kernel PCA**

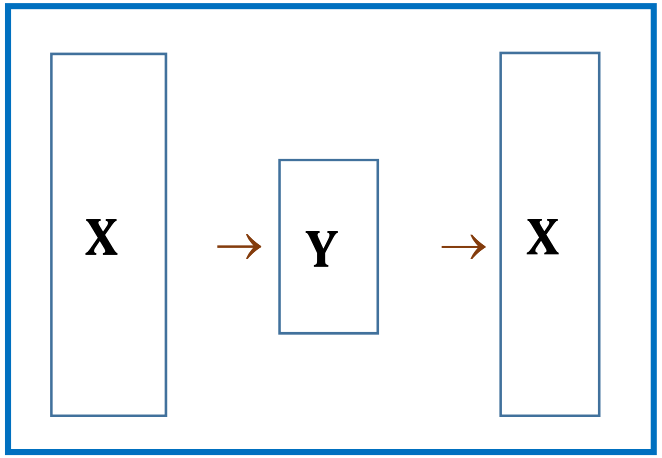
Neural Networks approach to dimensionality reduction



Encoder: embedding mapping $h: X \in R^p \rightarrow y = h(X) \in R^q$



Decoder: recovering mapping $g: y = h(X) \in R^q \rightarrow g(y) \in R^p$



Autoencoder: input vector $X \rightarrow$ recovered vector $X^* = g(h(X))$

When we can achieve the desired property: recovered vector $X^* = g(h(X))$ close to input vector X

$$X^* \approx X$$

$\mathbf{X}$ - a set consisting of 'all possible' input vectors X

$$\mathbf{X^*} \approx \mathbf{X}$$

$\mathbf{X^*} = \{X^* = g(h(X)) : X \in \mathbf{X}\}$ - a resulted set consisting of all recovered vectors

$$\mathbf{X^*} = \{X^* = g(h(X)): X \in \mathbf{X}\} = \{X^* = g(y) \in R^p: y \in \mathbf{Y} = h(\mathbf{X}) \in R^q\}$$

- q-dimensional surface in p-dimensional space

$\mathbf{X} \approx \mathbf{X}^*$: accurate dimensionality reduction is possible only when Data space X is approximately q-dimensional surface in p-dimensional space

Nonlinear Data model: Seung, Lee - The **Manifold** Ways of **Perception**. Science (**2000**)

Manifold model: the data lie on or near an unknown **Data manifold** $\mathbf{M}$ of lower dimension $q < p$ embedded in an ambient high-dimensional input space R^p

Dimensionality Reduction as Sample Embedding problem:

Given an input dataset $\mathbf{X}_n = \{X_1, X_2, \dots, X_n\} \subset R^p$, find an '$n$-point' Embedding mapping

$$h_{(n)}: \mathbf{X}_n \to \mathbf{Y}_n = \mathbf{Y}_{(n)}(X_n) = \{y_1, y_2, \dots, y_n\} \subset R^q,$$

such that the resulting q-dimensional dataset $\mathbf{Y}_n$, $q < p$, ***faithfully represents*** the sample $\mathbf{X}_n$

The term ***faithfully represents*** is not formalized in general:

in different methods it can be different due to choosing an optimized cost function $L_{(n)}(\mathbf{Y}_n|\mathbf{X}_n)$ which reflects the desired properties of the mapping $h_{(n)}$ to preserve certain subject-driven data properties

Clustering

a discovering groups (structures) in dataset $\mathbf{X}_n$

that contain *'similar'* (in one sense or another) sample points

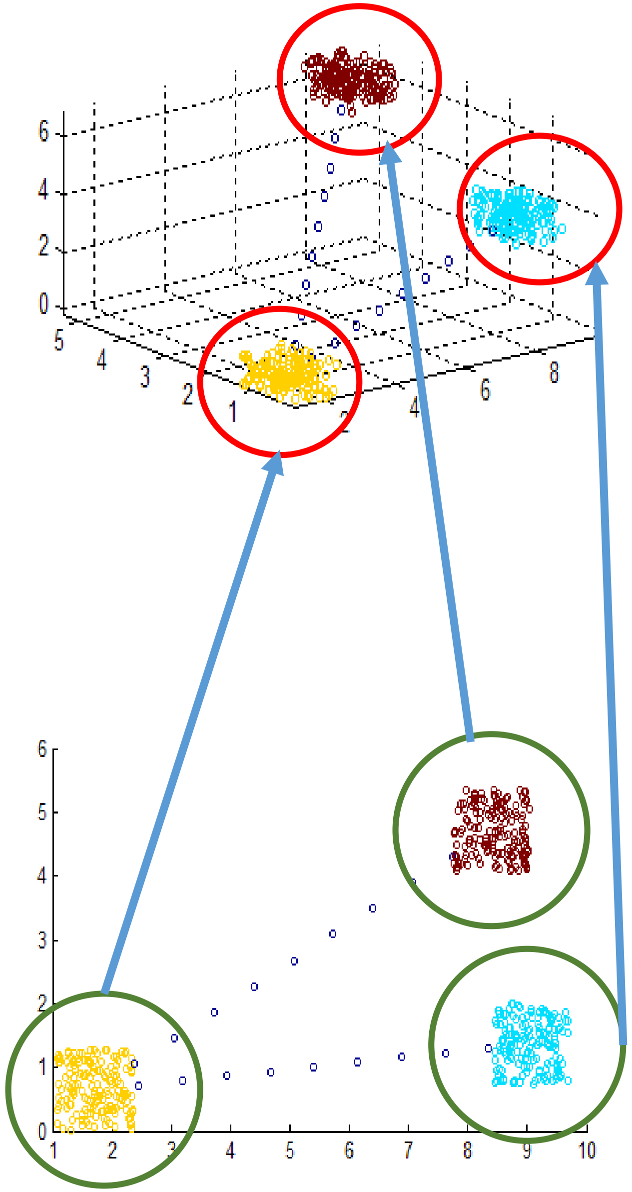
Embedding results in constructed low-dimensional dataset $\mathbf{Y}_n$

if *'faithfully represents'* in Embedding means a preserving *'similar'*

relations in Clustering, we solve Clustering problem for the dataset $\mathbf{Y}_n$

a solution of Clustering for problem for $\mathbf{X}_n$: the

preimages of clusters discovered in reduced dataset $\mathbf{Y}_n$



Graph-based algorithms

The graph-based algorithms have 3 basic steps.

1. Find sample points from small neighborhoods of the selected points

   - selected points = all sample points

   - ε -ball, K nearest neighbors

2. Estimate local properties of manifold by looking at neighborhoods found in Step 1

   - depends on a method

3. Find a global embedding that preserves the properties found in Step 2.

Locally linear embedding (LLE)

(Roweis, Saul: Nonlinear dimensionality reduction by locally linear embedding, 2000)

The method uses a linear mapping to capture local neighbourhood relations that are considered as representative of the local geometry of the Data manifold

- Sample points from small neighborhood of the selected point X lie approximately on q-dimensional linear subspace

- Linear relations between these sample points are approximately preserved for their linear projections into this linear subspace

- **LLE:** If the points $V_1, V_2, \ldots, V_k$, $k = q+1$, lie in q-dimensional linear subspace in 'general positions', any point V from the subspace is their linear combination $V = \sum_{j=1}^{k} w_j \times V_j$

$\{w_1, w_2, \ldots, w_k\}$ - barycentric coordinates

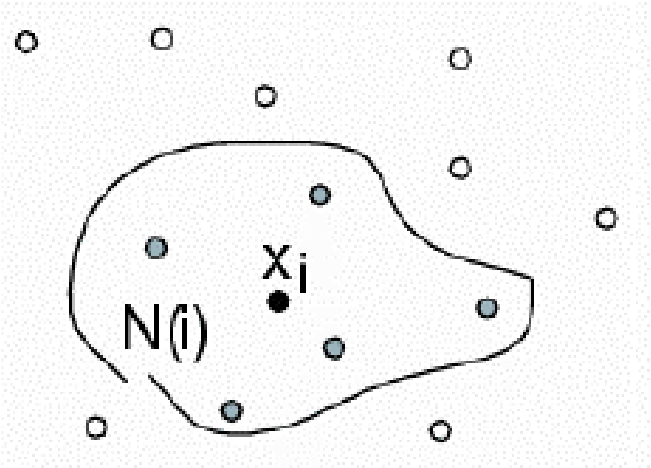
Manifold assumption: Data manifold is approximately “linear” when viewed locally

Step 1

Let $k > q + 1$, $X_i \in \mathbf{X}_n$ - selected sample point, a set

$N(i) = \{X_{1(i)}, X_{2(i)}, \dots , X_{k(i)}\} \in \mathbf{X}_n$ consists of its

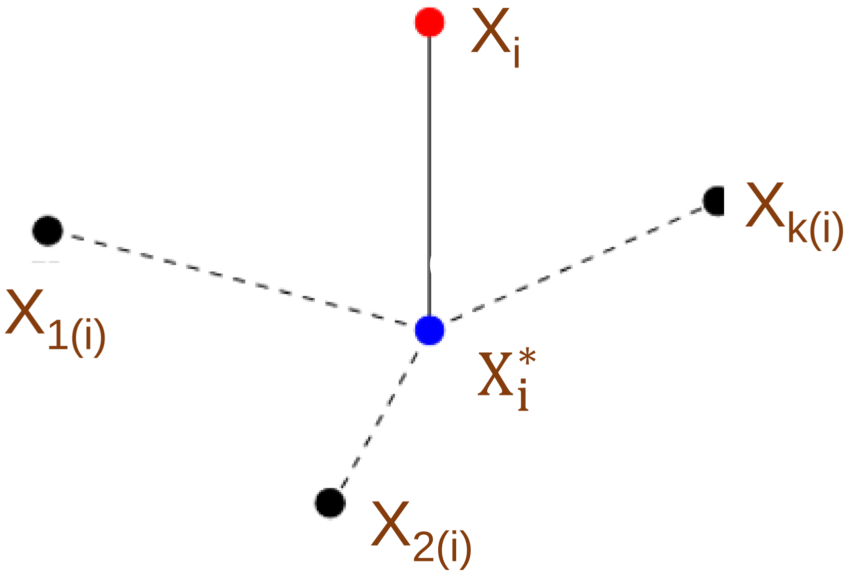
k Nearest Neighbors, excluding the point X_i itself



Step 2

- Look for 'the best' linear approximation of the point $X_i \in \mathbf{X}_n$ through its Nearest Neighbors: look for the weights $\{w_{1(i)}, w_{2(i)}, \dots, w_{k(i)}\}$ to minimize cost function $E_i = \left\| X_i - \sum_{j=1}^{k} w_{j(i)} \times X_{j(i)} \right\|^2$

- project X_i to the $\mathrm{Span}\{w_{1(i)}, w_{2(i)}, \dots, w_{k(i)}\} \rightarrow X_i^*$



- find barycentric coordinates of X_i^*

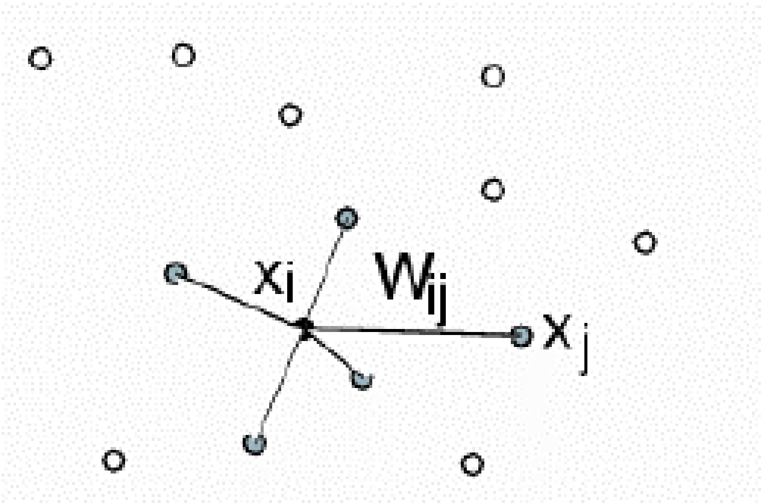
$$X_i^* = \sum_{j=1}^{k} w_{j(i)} \times X_{j(i)}$$

- the weights $\{w_{1(i)}, w_{2(i)}, \dots, w_{k(i)}\}$ are chosen so that X_i^* is the 'center od mass':

$$\sum_{j=1}^{k} w_{j(i)} = 1$$

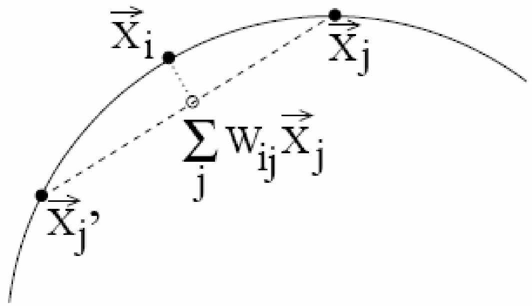
This provides an unaffectedness of the cost function from the data shift

- The weights $\{W_{1(i)}, W_{2(i)}, \dots, W_{k(i)}\}$ are the solution to this Least Squares task. Introduce the weights $W_i = \{W_{i1}, W_{i2}, \dots, W_{in}\}$ as



$$W_{it} = 0, \text{ if } X_t \notin N(i) \qquad W_{it} = W_{j(i)}, \text{ if } X_t = X_{j(i)}$$

- Thus: $\min E_i = E_i(W_i) = \left\|X_i - \sum_{j=1}^{n} W_{ij} \times X_j\right\|^2$



Step 3

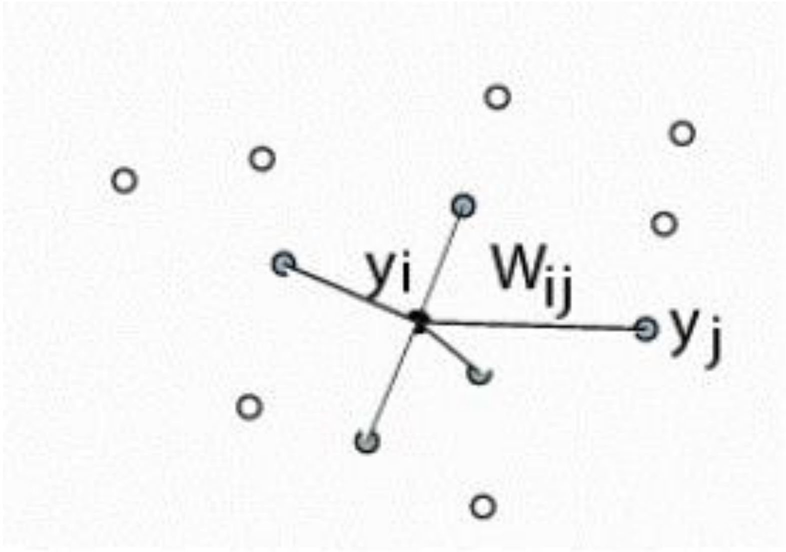
- Local geometry of the Data manifold in a vicinity of selected point X_i is characterized by the found weights $\{W_{i1}, W_{i2}, \ldots, W_{in}\}$ – what we wish to preserve (LLE!)

- Let $\mathbf{Y}_n = \{y_1, y_2, \ldots, y_n\}$ be desired q-dimensional features and $\{y_{1(i)}, y_{2(i)}, \ldots, y_{k(i)}\}$ is the subset corresponding to the Nearest Neighbors $\{X_{1(i)}, X_{2(i)}, \ldots, X_{k(i)}\}$. To preserve local geometry of the Data manifold, these features should provide small value of the quantity

$$\left\| y_i - \sum_{j=1}^{k} W_{j(i)} \times y_{j(i)} \right\|^2$$

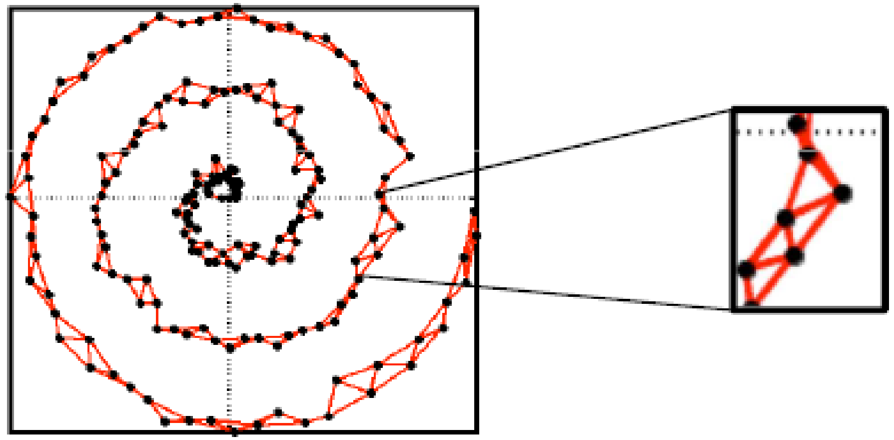
or, the same:

$$\left\| y_i - \sum_{j=1}^{n} W_{ij} \times y_j \right\|^2$$



- Local geometry should be preserved in the vicinities of **all** sample points, so the following 'total' cost function characterizes how much the features preserve the geometry:

$$L(\mathbf{Y}_n) = \sum_{i=1}^{n} \left\| y_i - \sum_{j=1}^{k} W_{j(i)} \times y_{j(i)} \right\|^2$$



- Under constraints $\{\sum_{j=1}^{n} W_{ij} = 1\}$, the features may be determined up to a shift, so require for definiteness: $\sum_{i=1}^{n} y_i = 0$

- to avoid trivial degenerate solution, introduce the additional constraint $\frac{1}{n}\sum_{i=1}^{n} y_i \times y_i^T = I_q$

Denote:

$\mathbf{Y} = (y_1\ y_2\ \ldots\ y_n)$ - $q \times n$ matrix whose i-th column is y_i

W - $n \times n$ matrix whose i-th row is vector-row $(W_{i1}, W_{i2}, \ldots, W_{in})$

The total cost function and the constraint can be written as:

$$L(\mathbf{Y}) = \mathrm{Tr}(\mathbf{Y} \times (I_n - W)^T \times (I_n - W) \times \mathbf{Y}^T)$$

$$\mathbf{Y} \times \mathbf{Y}^T = n \times I_q$$

$$\mathbf{Y} \times \mathbf{1}_n = 0$$

The Eigenvalues problem: to minimize quadratic form $L(\mathbf{Y}) = \mathrm{Tr}(\mathbf{Y} \times (I_n - W)^T \times (I_n - W) \times \mathbf{Y}^T)$ over $\mathbf{Y}$ under the constraints $\mathbf{Y} \times \mathbf{Y}^T = n \times I_q$ and $\mathbf{Y} \times \mathbf{1}_n = 0$

The LLE-solution corresponds to the eigenvectors of the $n \times n$ matrix $(I_n - W)^T \times (I_n - W)$ with the smallest eigenvalues, namely:

- the smallest eigenvalue is 0 that corresponds to the vector $\mathbf{1}_n$ that should be discarded

- q eigenvectors with next smallest eigenvalues, after normalizing, are chosen are rows of the $q \times n$ matrix $\mathbf{Y}$

Step 2: How to construct the weighted matrix W

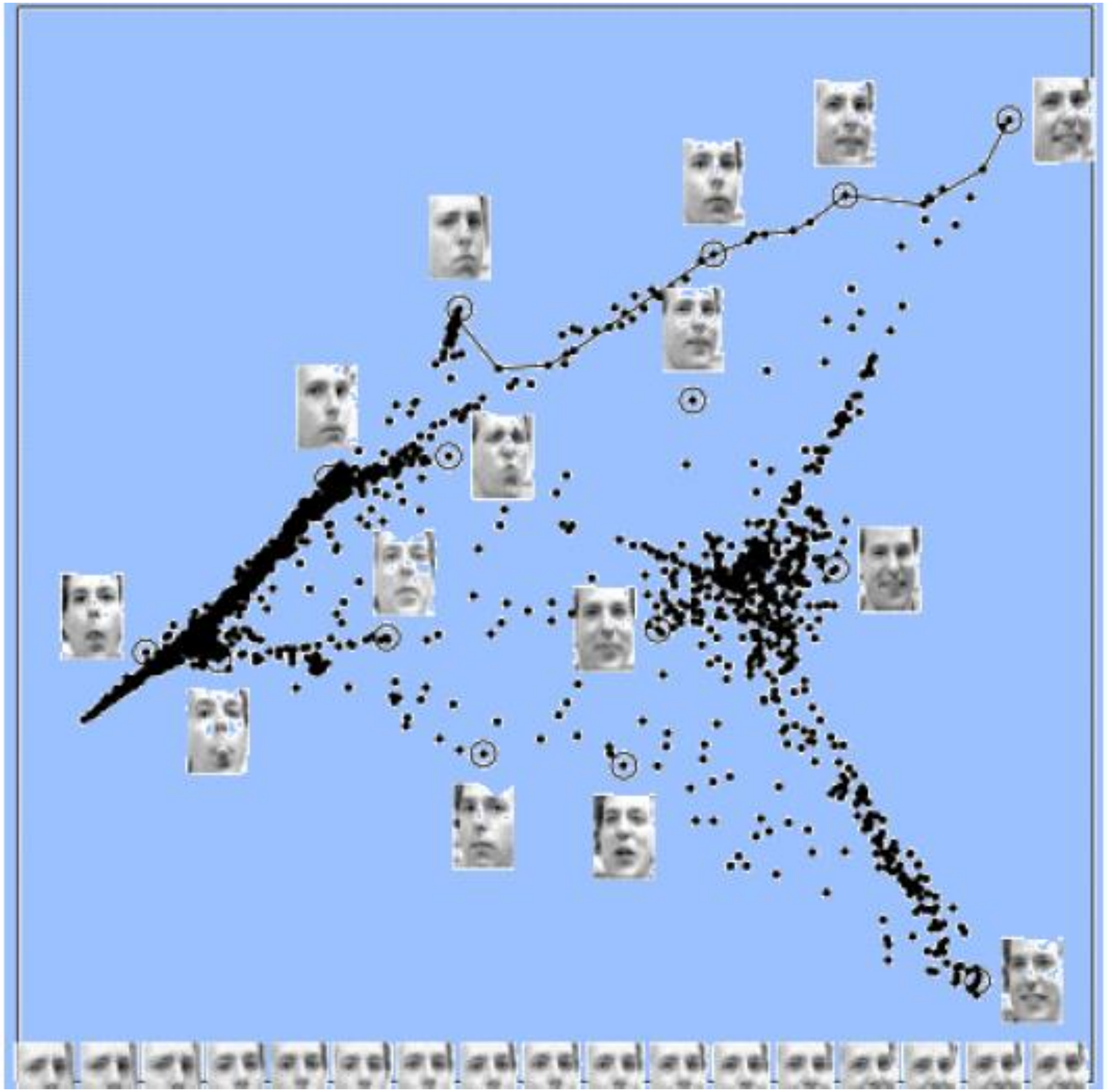
The weights $\{W_{1(i)}, W_{2(i)}, \ldots, W_{k(i)}\}$ are following Least Squares solution to the minimization problem for the cost function $E_i = \left\| X_i - \sum_{j=1}^{k} w_{j(i)} \times X_{j(i)} \right\|^2$:

- Local $k \times k$ Gram matrix $G_i = \left\| G_{ts(i)} \right\|$ is constructed from the subsample $\{X_i, X_{1(i)}, X_{2(i)}, \ldots, X_{k(i)}\}$ as: $G_{ts(i)} = (X_{t(i)} - X_i, X_{s(i)} - X_i)$

- $W_{t(i)} = \dfrac{\sum_{s=1}^{k} G_{tsi}^{-1}}{\sum_{t,s=1}^{k} G_{tsi}^{-1}}$ $\quad t = 1, 2, \ldots, k$

Pose expression:

- p = 560 (pixels)
- n = 1965
- q = 2
- k = 12 Nearest Neighbors

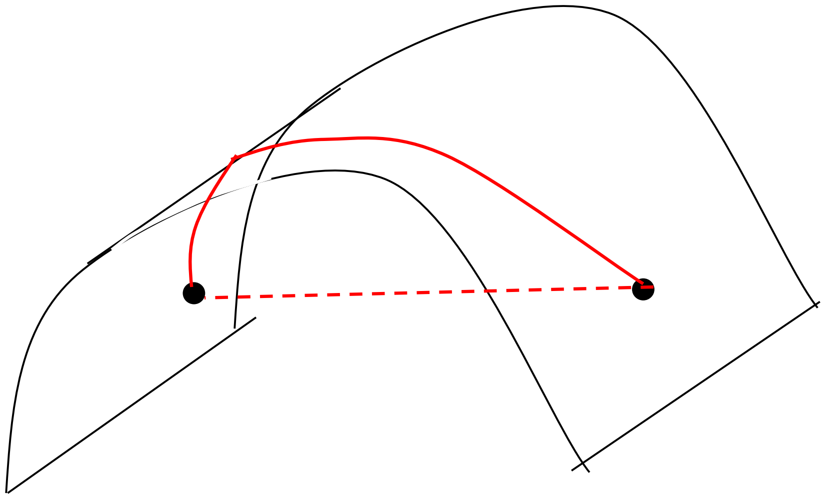


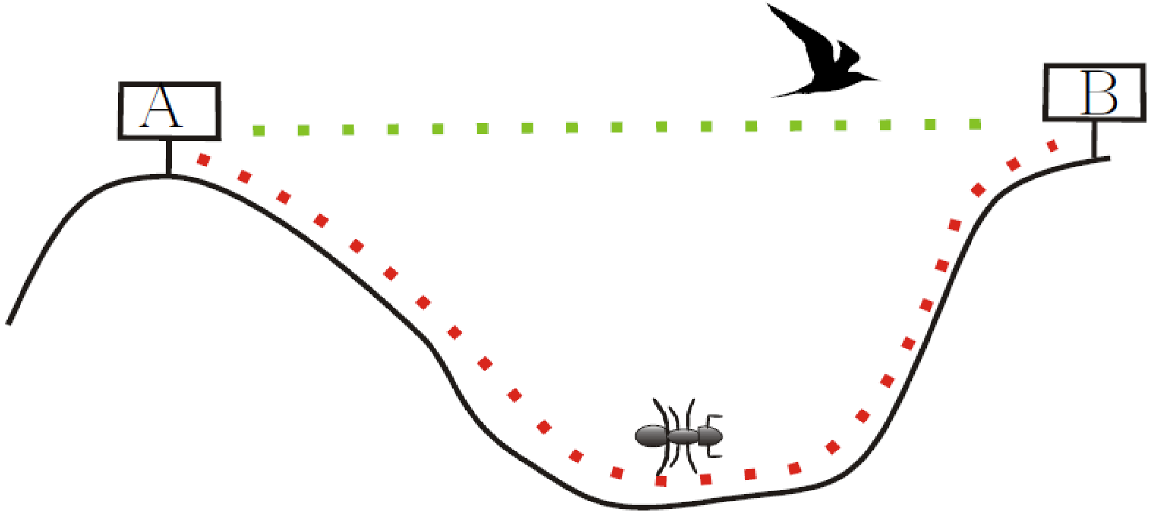
ISOmetric MAPing (ISOMAP)

(Tehenbaum, de Silva, Langford: A global geometric framework for nonlinear dimensionality reduction, 2000)

- Metric MDS: preserves the Euclidean proximities
- Metric MDS is equivalent to the PCA and is ‘the best’ in linear space

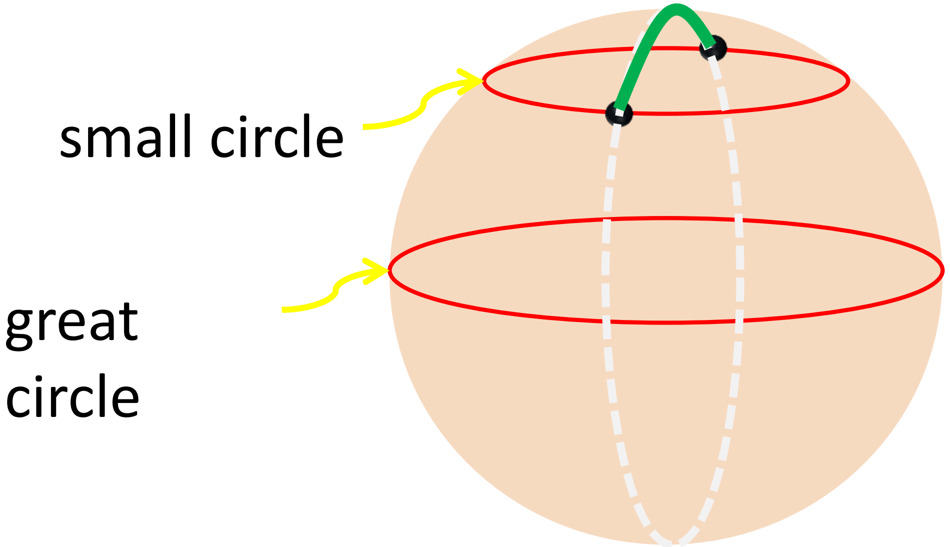
Data manifold is embedded in Euclidean space, but Euclidean distance between the manifold points is ***not*** the correct way to measure distance – is not a ‘shortest path’

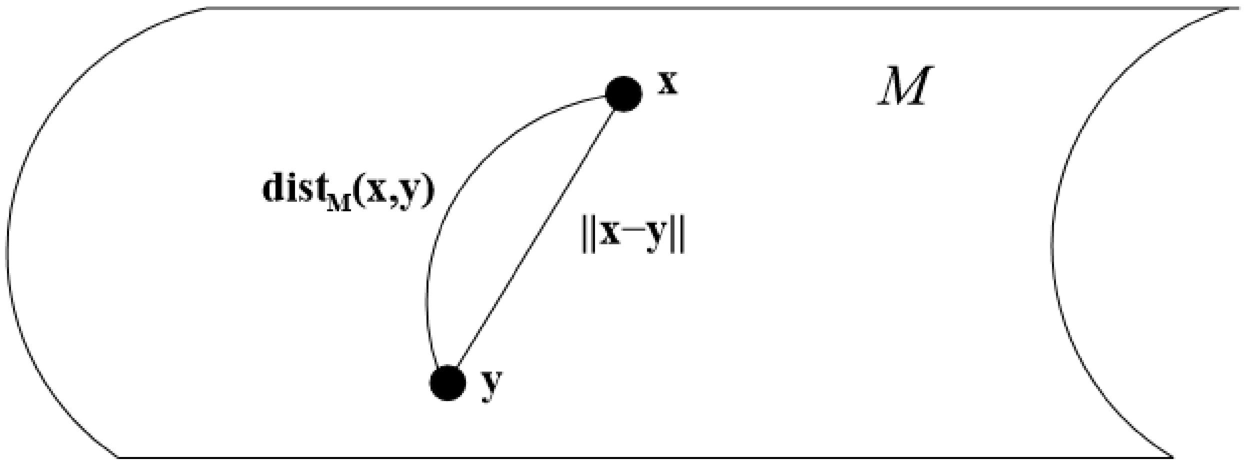




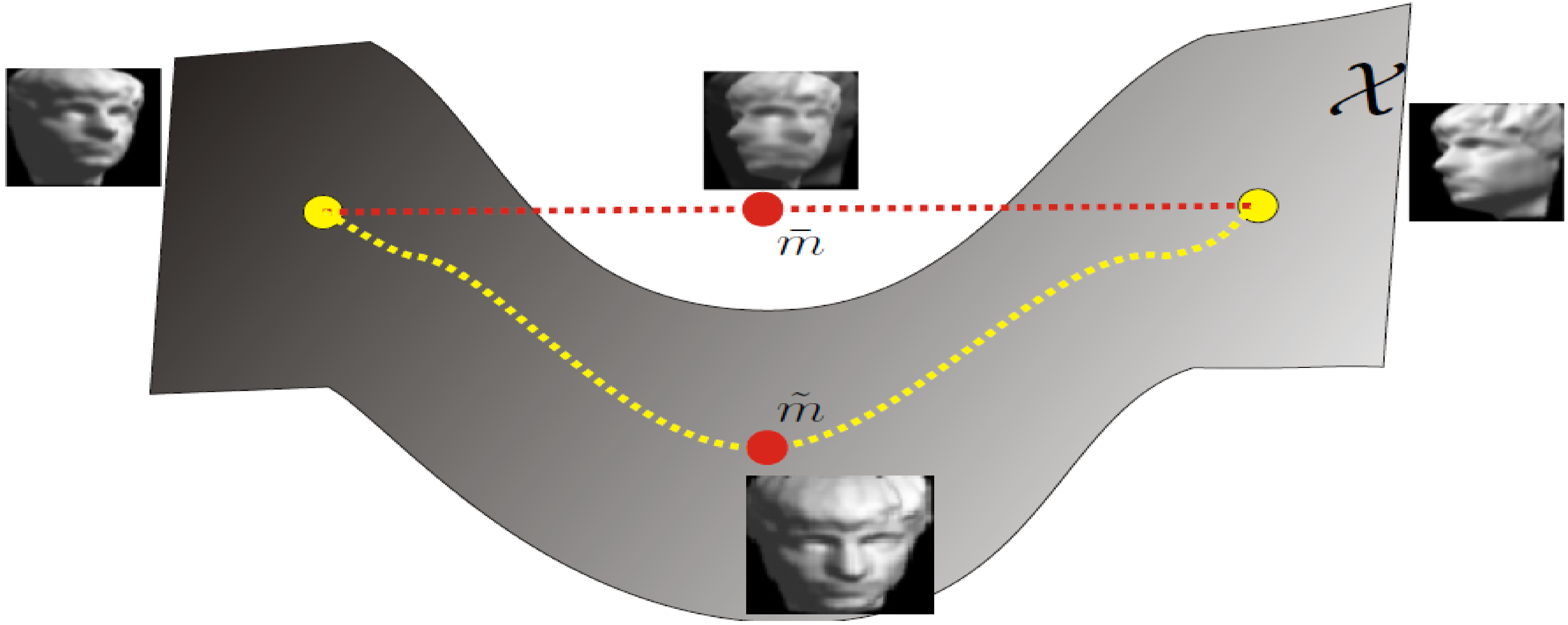
- Euclidean distances do not reflect the proximities in nonlinear case

The shortest “geodesic path” between two manifold points is geodesic way

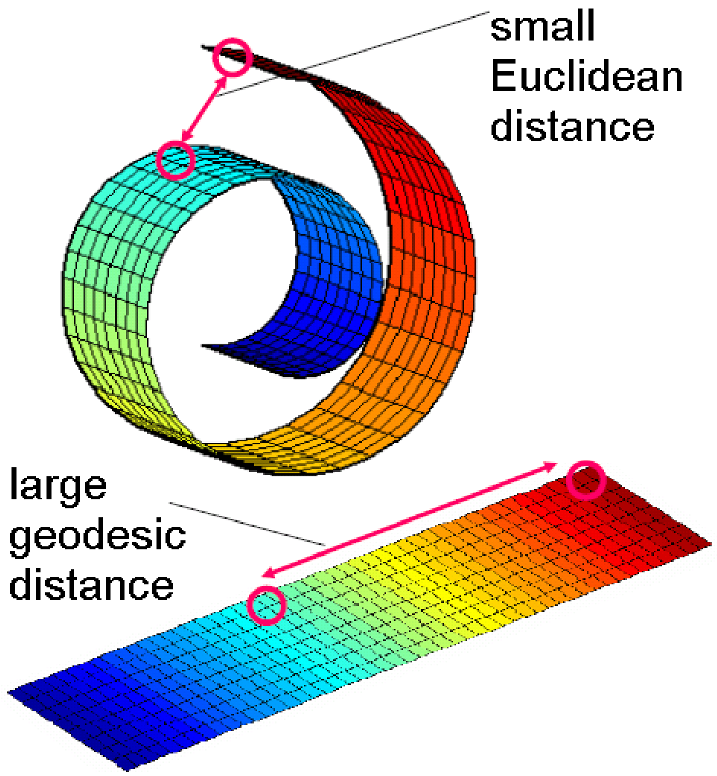




The shortest "geodesic path" between two images passes only through the Image manifold points, in contrast to the Euclidean shortest path between these points in ambient Euclidean space



- The Euclidean distance ‘shortcuts’ the manifold
- The geodesic distance calculates the shortest path along the manifold

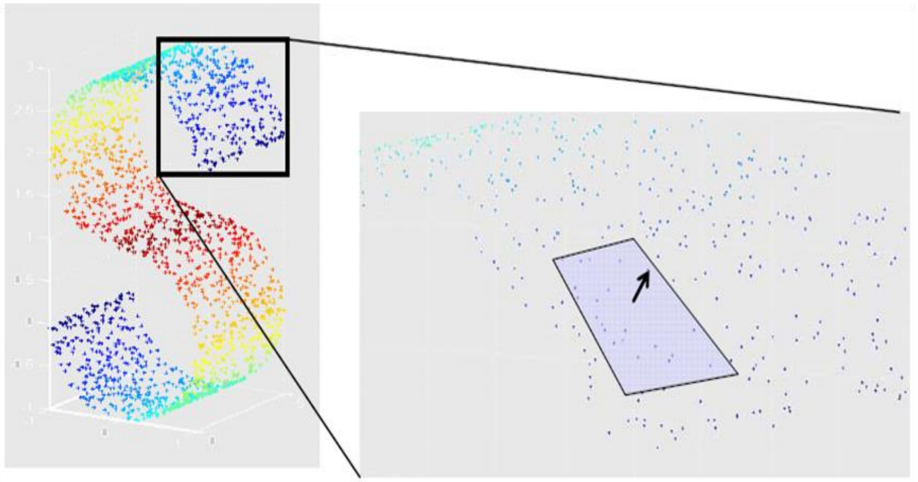


ISOMAP:

- extends the MDS to nonlinear case using **geodesics** instead of Euclidean distances in MDS
- preserves the data manifold geometry by capturing the geodesic distances

- for neighboring points, Euclidean distance is a good approximation to the geodesic distance

.



- for distant points, estimating their geodesic distance by “a chain of short paths” between neighboring points

Step 1: constructing small neighborhoods

- X_i - selected sample point,
- $U(X_i)$ - small neighborhood of the point X_i excluding the point X_i itself

ε-neighborhood:

$$U(X_i) = \{X' \in \mathbf{X}_n : \|X' - X\| \leq \varepsilon\}$$

k Nearest Neighbors

$$X_{(1)}, X_{(2)}, \ldots , X_{(n)} \in \mathbf{X}_n :$$

$$\|X_{(1)} - X\| \leq \|X_{(2)} - X\| \leq \ldots \leq \|X_{(n)} - X\|$$

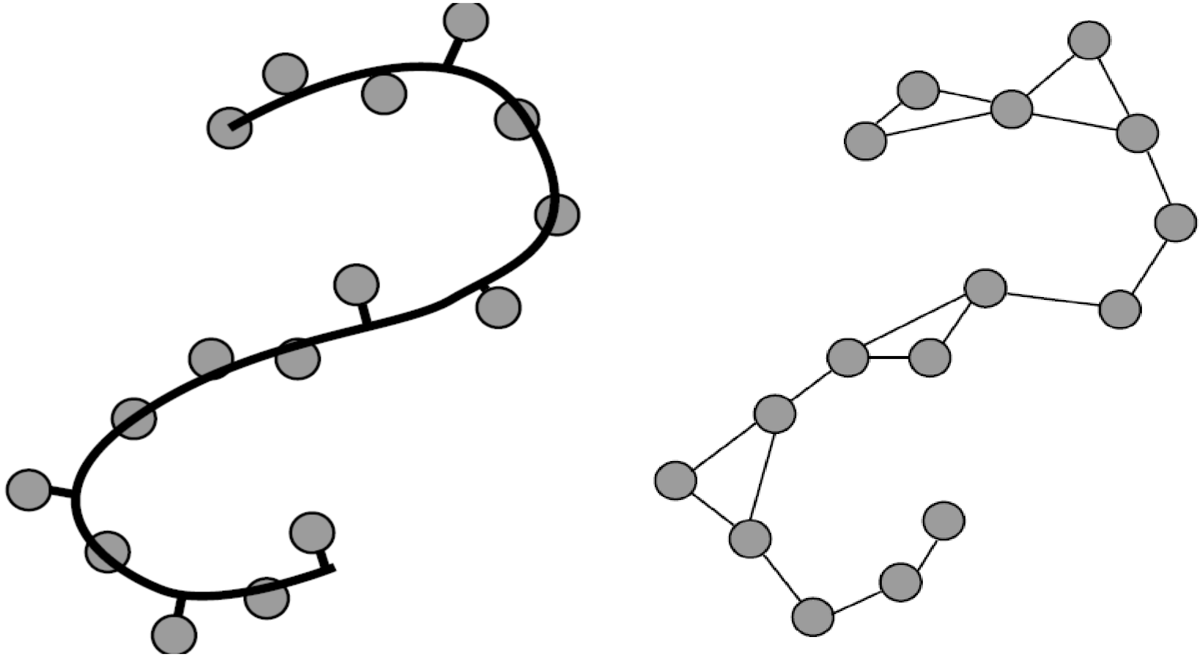
$$U(X) = \{X_{(1)}, X_{(2)}, \ldots , X_{(k)}\}$$

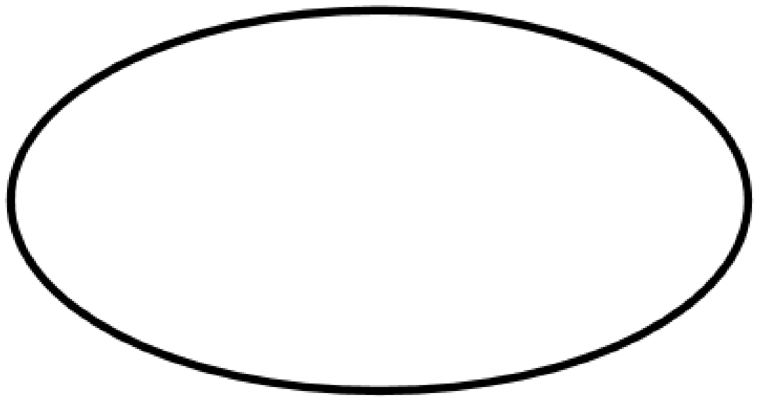
Step 2: Constructing the Adjacency graph (common for many methods)

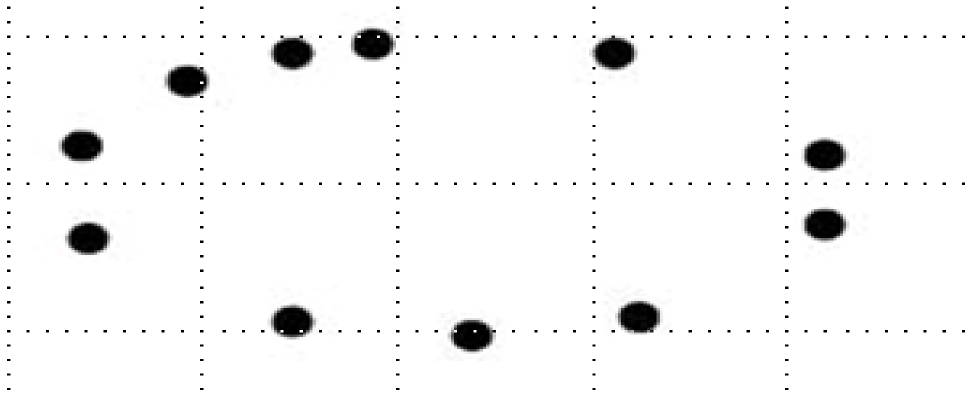
Weighted undirected sample graph $\Gamma(\mathbf{X}_n)$:

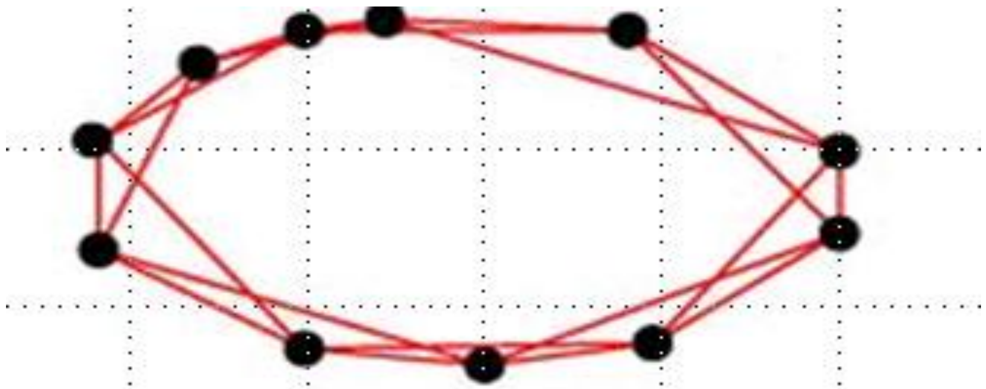
- the sample points $\{X_i\}$ are nodes
- the edges connect the nodes X_i and X_j if and only when

$$X_i \in U(X_j) \text{ and } X_j \in U(X_i)$$

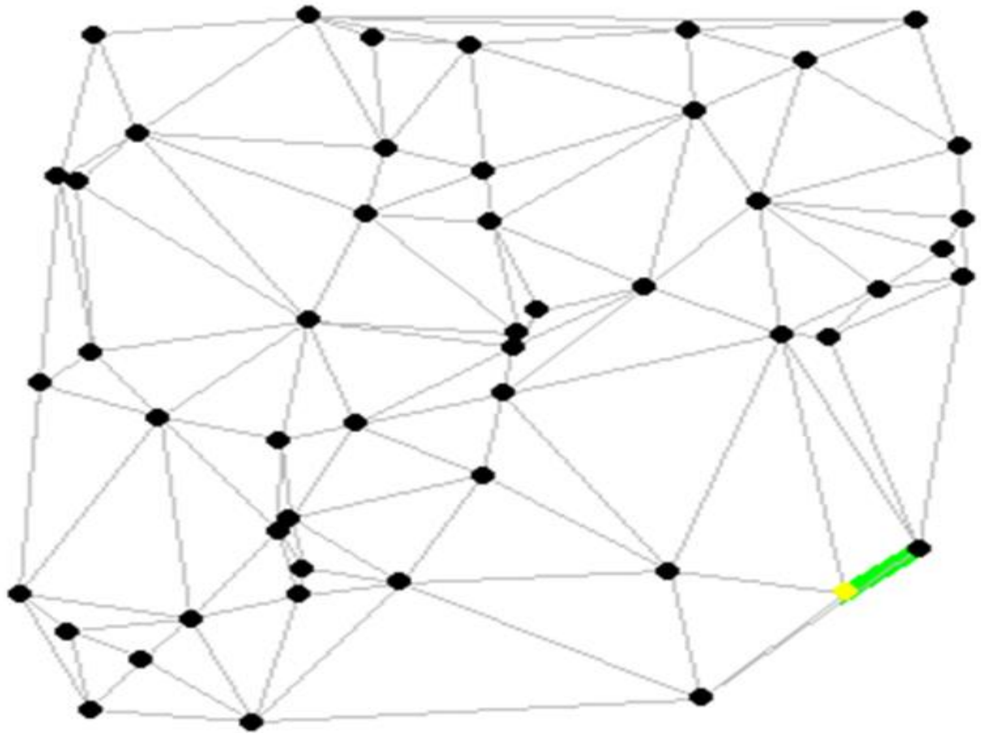


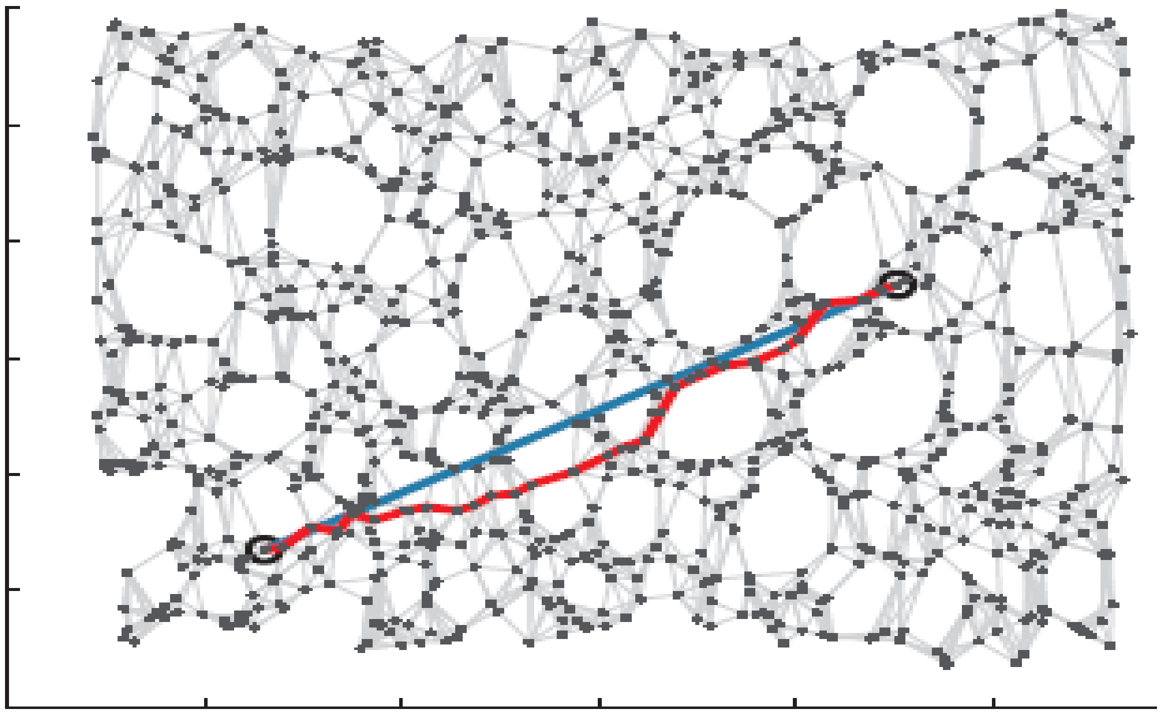






Step 3. Compute the shortest-distance paths between all the graph vertices (Dijkstra's algorithm)





$D(X_i, X_j)$ - the lengths of the shortest "geodesic paths" between the points X_i and X_j

The “shortest path” between two points on the graph G approximates **geodesic lines** between these points on the Data manifold: for arbitrary given $\lambda > 0$ and $\mu > 0$, for sufficiently large sample size n, we have the relation

$$1 - \lambda \leq \frac{\text{Recovered distance}}{\text{Original distance}} \leq 1 + \lambda$$

with probability at least $(1 - \mu)$

Step 4. In Metric MDS, the averaged pairwise distances

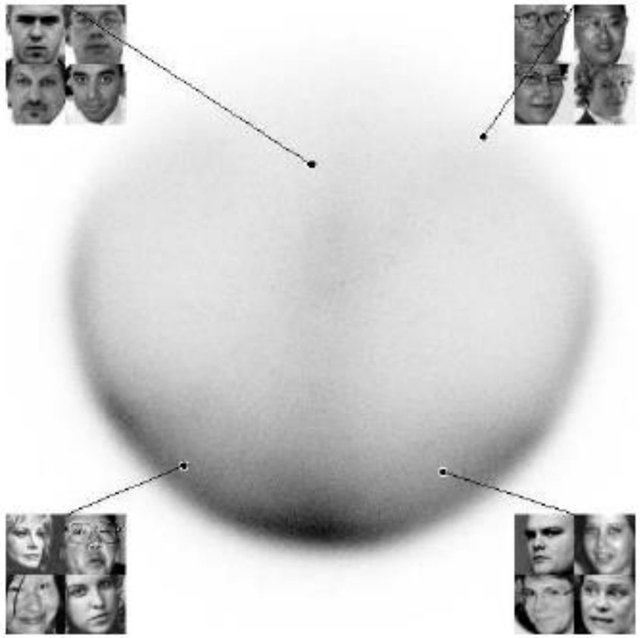
$$\Delta_{\text{MetricMDS}} = \sum_{i,j=1}^{n} \left(\|X_i - X_j\|^2 - \|y_i - y_j\|^2 \right)^2$$

is replaced by a quantity

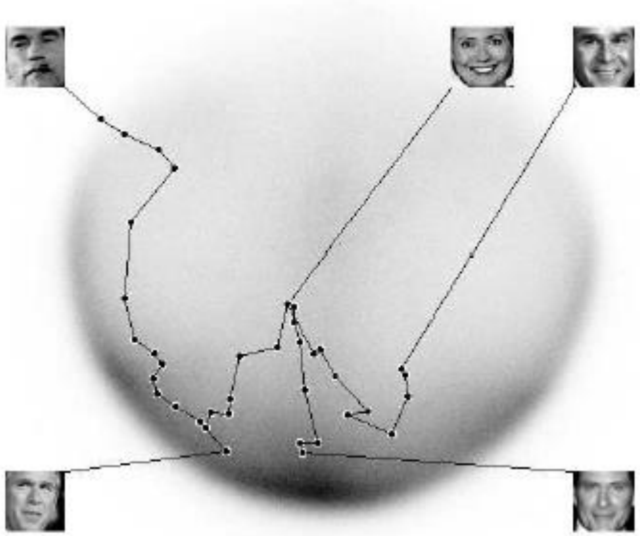
$$\Delta_{\text{ISOMAP}} = \sum_{i,j=1}^{n} \left(\left(D(X_i, X_j) \right)^2 - \|y_i - y_j\|^2 \right)^2$$

The remaining MDS-procedures are unchanged

Face samples at different points on the Image manifold



Approximate geodesic paths between different faces



The faces on the “shortest-paths”

Classification (supervised learning)

original dataset consists of labeled examples $\{(X_i , \lambda_i)\}$:

- inputs $\{X_1, X_2, \ldots , X_n\}$
- outputs (labels) $\Lambda_n = \{\lambda_1, \lambda_2, \ldots , \lambda_n\}$, $\lambda \in \{1, 2, \ldots , m\}$, $m \geq 2$, known for the inputs

The problem: to generalize a function (mapping) F from inputs to outputs

$$F: X \rightarrow \lambda = \lambda(X) \in \{1, 2, \ldots , m\}$$

which can then be used to generate an output for a **previously unseen input** $X \in \mathbf{X}$

Embedding: original high-dimensional inputs $\mathbf{X}_n \quad \to \quad$ low-dimensional features $\mathbf{Y}_n = \{y_1, y_2, \ldots, y_n\}$

Reduced classification problem for reduced dataset $\{(y_i, \lambda_i)\}$: f – 'reduced' solution

Use the reduced solution for the solution of original problem:

$$F(X) = f(y),\ y = h(X)\ -$$

low-dimensional representation $y = h(X)$ of Out-of-Sample input $X \in \mathbf{X} / \mathbf{X}_n$ Is required

Out-of-Sample extension for Embedding method

Out-of-Sample extension for Embedding method

Naïve solution: applying the Embedding technique $h_{(n+1)}$ to the dataset $\mathbf{X}_{n+1} = \mathbf{X}_n \cup X$ resulting in low-dimensional features $h_{(n+1)}(\mathbf{X}_{n+1}) = \{y_{1(n+1)}, y_{2(n+1)}, \dots, y_{n(n+1)}, y\}$

But: $h_{(n+1)}(\mathbf{X}_n) = \{y_{1(n+1)}, y_{2(n+1)}, \dots, y_{n(n+1)}\} \neq h_{(n)}(\mathbf{X}_n) = \{y_{1(n)}, y_{2(n)}, \dots, y_{n(n)}\}$

Kernel PCA

Dataset $\mathbf{X}_{(n)} = \{X_1, X_2, \ldots , X_n\}, X \in R^p \to \Phi(X)$ - desired transform

$\mathbf{X}_{(n)} \to$ transformed dataset $\mathbf{\Phi}_{(n)} = \{\Phi_i = \Phi(X_i), i = 1, 2, \ldots , n\}$

PCA-solution applied to the transformed dataset $\mathbf{\Phi}_{(n)}$ depends on the dataset only through the inner products $K(X_i, X_j) = (\bar{\Phi}(X_i), \bar{\Phi}(X_j))$ of the centered transformed data (kernel functions)

PCA-solution based on kernels $\{K(X_i, X_j)\}$, without performing the transformation $\Phi(X)$ - **kernel trick**

Kernel PCA: is based on the solution to eigenvector problem for centered matrix $\bar{K} = \|\bar{K}(X_i, X_j)\|$:

eigenvectors $\bar{K} \times \alpha_k = \gamma_k \times \alpha_k$, $k = 1, 2, \ldots , q$, correspond to largest eigenvalues $\gamma_1 \geq \gamma_2 \geq \ldots \geq \gamma_q$

Kernel PCA - solution: Embedding $y = \begin{pmatrix} y_1 \\ \cdots \\ y_q \end{pmatrix}$ of arbitrary vector X (sample vector / **OoS**) has coordinates: $y_k = \gamma_k^{1/2} \times \sum_{j=1}^{n} \alpha_{kj} \times K(X_j, X)$, $k = 1, 2, \ldots, q$

LLE and ISOMAP solutions are the Kernel PCA solutions for specific kernels

$K_{LLE}(X_i, X_j)$ and $K_{ISOMAP}(X_i, X_j)$

LLE

The LLE-solution corresponds to the eigenvectors of the $n \times n$ matrix $M = (I_n - W)^T \times (I_n - W)$ with the smallest nonzero eigenvalues $\{\lambda_1, \lambda_2, \dots, \lambda_q\}$ and to the Kernel PCA solution with a kernel

$$K_{LLE}(X_i, X_j) = W_{ij} + W_{ji} - \sum_{k=1}^{n} W_{ki} \times W_{kj}$$

ISOMAP

The Isomap-solution corresponds to MDS with the distances $D(X_i, X_j)$ - the estimated lengths of the shortest geodesic paths between the points X_i and X_j and to the Kernel PCA solution with a kernel

$$K_{ISOMAP}(X_i, X_j) = - D^2(X_i, X_j) - \sum_{k=1}^{n} D^2(X_k, X_i) - \sum_{k=1}^{n} D^2\left(X_j, X_k\right) + \sum_{k,s=1}^{n} D^2(X_s, X_k)$$