Unsupervised Learning Techniques

9.520 Class 07, 1 March 2006

Andrea Caponnetto

About this class

Goal To introduce some methods for unsupervised learning: Gaussian Mixtures, K-Means, ISOMAP, HLLE, Laplacian Eigenmaps.

Unsupervised learning

Only u i.i.d. samples drawn on X from the unknown marginal distribution $p(x)$

$$\{x_1, x_2, \ldots, x_u\}.$$

The goal is to infer properties of this probability density.

In low-dimension many *nonparametric* methods allow direct estimation of $p(x)$ itself. Owing to the *curse of dimensionality*, this methods fail in high dimension.

One must settle for estimation of *crude global models*.

Unsupervised learning (cont.)

Different types of simple descriptive statistics that characterize aspects of $p(x)$

- *mixture modelling*

  representation of $p(x)$ by a mixture of simple densities representing different types or classes of observations [eg. Gaussian mixtures]

- *combinatorial clustering*

  attempt to find multiple regions of X that contain modes of X [eg. K-Means]

- *dimensionality reduction*

  attempt to identify low-dimensional manifolds in X that represent high data density [eg. ISOMAP,HLLE, Laplacian Eigenmaps]

- *manifold learning*

  attempt to determine very specific geometrical or topological invariants of $p(x)$ [eg. Homology learning]

Limited formalization

With *supervised* and *semi-supervised* learning there is a clear measure of effectiveness of different methods. The *expected loss* of various estimators $I[f_S]$ can be estimated on *validation set*.

In the context of unsupervised learning, it is *difficult to find such a direct measure of success*.

This situation has led to **proliferation of proposed methods**.

Mixture Modelling

Assumption that data is i.i.d. sampled from some probability distribution $p(x)$.

$p(x)$ is modelled as a mixture of component density functions, each component corresponding to a *cluster* or *mode*.

The free parameters of the model are fit to the data by *maximum likelihood*.

Gaussian Mixtures

We first choose a **parametric model** P_θ for the unknown density $p(x)$, hence maximize the **likelihood** of our data relative to the parameters θ.

Example: two-component gaussian mixture model with parameters

$$\theta = (\pi, \mu_1, \Sigma_1, \mu_2, \Sigma_2).$$

The model:

$$P_\theta(x) = (1 - \pi) G_{\Sigma_1}(x - \mu_1) + \pi G_{\Sigma_2}(x - \mu_2)$$

Maximize the log-likelihood

$$\ell(\theta | \{x_1, \ldots, x_u\}) = \sum_{i=1}^{u} \log P_\theta(x_i)$$

The EM algorithm

Maximization of $\ell(\theta|\{x_1,\ldots,x_u\})$ is a difficult problem. Iterative maximization strategies, as the EM algorithm, can be used in practice to get *local maxima*.

1. **Expectation:** compute the responsibilities

$$\gamma_i = \frac{\pi G_{\Sigma_2}(x_i-\mu_2)}{(1-\pi)G_{\Sigma_1}(x_i-\mu_1)+\pi G_{\Sigma_2}(x_i-\mu_2)}$$

2. **Maximization:** compute means and variances

$$\mu_2 = \frac{\sum_i \gamma_i x_i}{\sum_i \gamma_i}, \qquad \Sigma_2 = \frac{\sum_i \gamma_i (x_i-\mu_2)(x_i-\mu_2)^T}{\sum_i \gamma_i}, \qquad etc$$

   and the mixing probability $\pi = \frac{1}{u}\sum_i \gamma_i$.

3. Iterate until convergence

Combinatorial Clustering

Algorithms in this class work on the data *without any reference to an underlying probability model*.

The goal is assigning each data point x_i to a cluster k belonging a predefined set $\{1, 2, \ldots, K\}$

$$C(i) = k, \qquad i = 1, 2, \ldots, u$$

The optimal *encoder* $C^*(i)$ minimizes the overall dissimilarities $d(x_i, x_j)$ between points x_i, x_j assigned to the same cluster

$$W(C) = \frac{1}{2} \sum_{k=1}^{K} \sum_{C(i)=k} \sum_{C(j)=k} d(x_i, x_j)$$

The simplest choice for the dissimilarity $d(\cdot, \cdot)$ is the squared Euclidean distance in X

Combinatorial Clustering (cont.)

The minimization of the *within-cluster point scatter* $W(C)$ is straightforward in principle, but...

the number of distinct assignments *grows exponentially* with the number of data points u

$$S(u,K) = \frac{1}{K!}\sum_{k=1}^{K}(-1)^{K-k}\binom{K}{k}k^{u}$$

already $S(19,4) \simeq 10^{10}$!

In practice, clustering algorithms look for good suboptimal solutions.

Most popular algorithms are based on *iterative descent strategies*. Convergence to *local* optima.

K-Means

If $d(x_i, x_j) = \|x_i - x_j\|^2$, introducing the mean vectors $\bar{x}_k$ associated to the k-th cluster, the within-cluster point scatter $W(C)$ can be rewritten as

$$W(C) = \frac{1}{2} \sum_{k=1}^{K} \sum_{C(i)=k} \sum_{C(j)=k} \|x_i - x_j\|^2 = \sum_{k=1}^{K} \sum_{C(i)=k} \|x_i - \bar{x}_k\|^2.$$

Exploiting this representation one can easily verify that the optimal encoder C^* is the solution of the *enlarged minimization problem*

$$\min_{C,(m_1,\ldots,m_K)} \sum_{k=1}^{K} \sum_{C(i)=k} \|x_i - m_k\|^2.$$

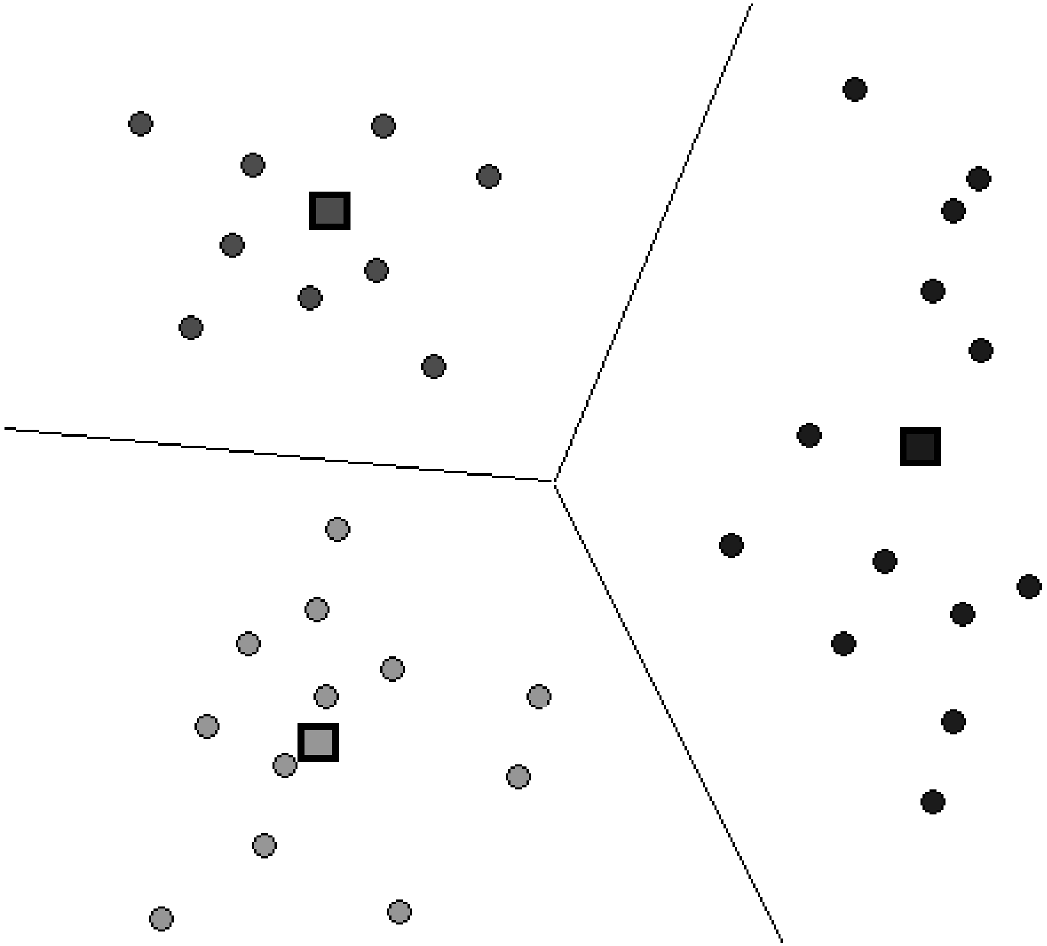
K-Means (cont.)

K-Means attempts the minimization of the enlarged problem by an iterative alternating procedure. Each step 1 and 2 reduces the objective function, so **convergence is assured**.

1. minimization with respect to $(m_1, \ldots, m_K)$, getting
$$m_k = \bar{x}_k$$
2. minimization with respect to C, getting
$$C(i) = \arg \min_{1 \le k \le K} \|x_i - m_k\|$$
3. do until C does not change

One should compare solutions derived from **different initial random means**, and choose best local minimum.

Voronoi tessellation

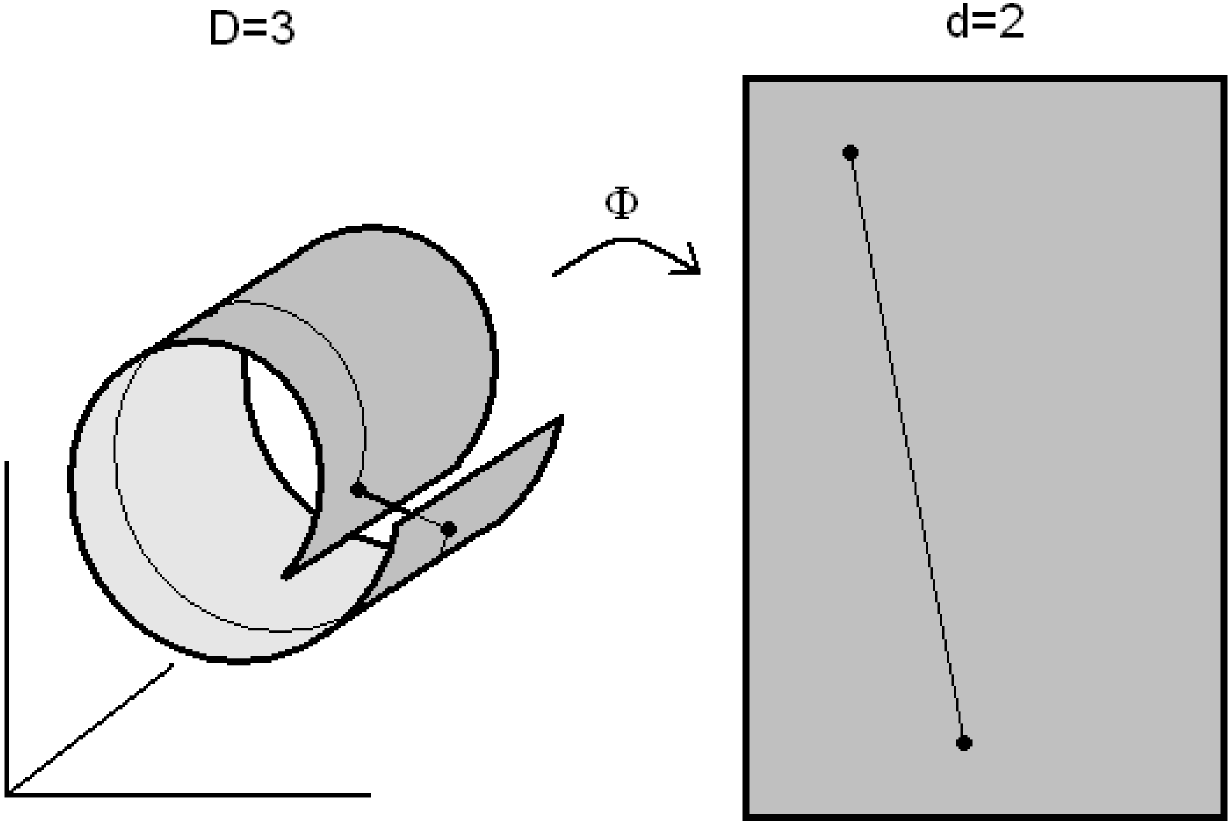


Dimensionality reduction

Often reducing the dimensionality of a problem is an effective preliminary step toward the actual solution of a regression or classification problem.

We look for a mapping Φ from the high dimensional space $\mathbb{R}^D$ to the low dimensional space $\mathbb{R}^d$ which preserves some relevant geometrical structure of our problem.

Dimensionality reduction



Principal Component Analysis (PCA)

Trying to approximate data $\{x_1, \ldots, x_u\}$ in $\mathbb{R}^D$ by a d-*dimensional hyperplane*

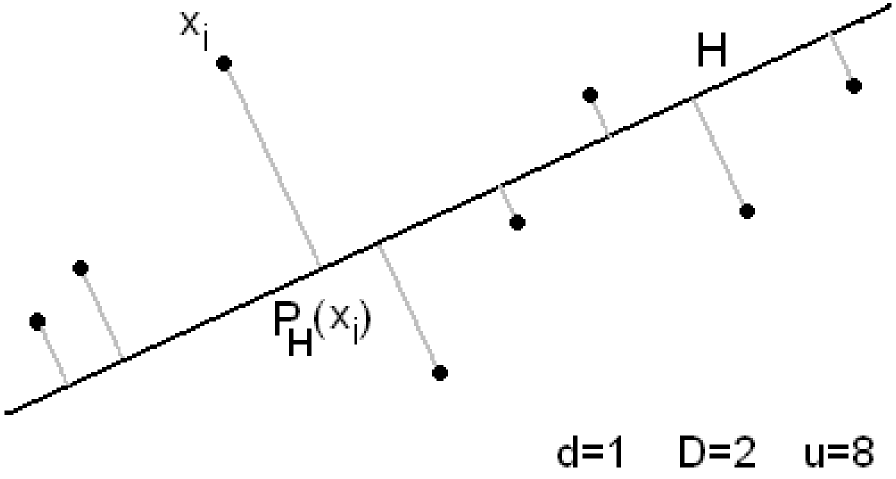
$$H = \{\mathbf{c} + \mathbf{V}\theta | \theta \in \mathbb{R}^d\}$$

$\mathbf{c}$ vector in $\mathbb{R}^D$, θ coordinates vector in $\mathbb{R}^d$ and $\mathbf{V} = (\mathbf{v}_1, \ldots, \mathbf{v}_d)$, $D \times d$ matrix with $\{\mathbf{v}_i\}$ orthonormal system of vectors in $\mathbb{R}^D$.

Problem: find H which minimizes sum of squared distances of data points x_i from H

$$H^* = \arg\min_H \sum_{i=1}^{u} \|x_i - P_H(x_i)\|^2$$

Linear approximation



PCA: the algorithm

1. center data points: $\sum_{i=1}^{u} x_i = 0$

2. define $u \times D$ matrix $\mathbf{X} = (x_1, \ldots, x_u)^T$

3. construct *singular value decomposition* $\mathbf{X} = \mathbf{U}\mathbf{\Sigma}\mathbf{W}^T$

   - $D \times D$ matrix $\mathbf{W} = (\mathbf{w}_1, \ldots, \mathbf{w}_D)$, with $\{\mathbf{w}_i\}$ *right eigenvectors* of $\mathbf{X}$

   - $u \times D$ matrix $\mathbf{U} = (\mathbf{u}_1, \ldots, \mathbf{u}_D)$, with $\{\mathbf{u}_i\}$ *left eigenvectors* of $\mathbf{X}$

   - $D \times D$ matrix $\mathbf{\Sigma} = diag(\sigma_1, \ldots, \sigma_D)$, with $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_D \geq 0$ *singular eigenvectors* of $\mathbf{X}$

4. answer: $\mathbf{V} = (\mathbf{w}_1, \ldots, \mathbf{w}_d)$

Sketch of proof

- Rewrite the minimization problem

$$\min_{\mathbf{c},\mathbf{V},\{\theta_i\}} \sum_{i=1}^{u} \|x_i - c - \mathbf{V}\theta_i\|^2$$

- Centering and minimizing with respect to $\mathbf{c}$ and θ_i gives

$$\mathbf{c} = 0, \qquad \theta_i = \mathbf{V}^T x_i$$

- Plugging into the minimization problem

$$\begin{aligned}\arg\min_{\mathbf{V}} \sum_{i=1}^{u} \|x_i - \mathbf{V}\mathbf{V}^T x_i\|^2 &= \arg\max_{\mathbf{V}} \sum_{i=1}^{u} x_i^T \mathbf{V}\mathbf{V}^T x_i \\ &= \arg\max_{\mathbf{V}} \sum_{j=1}^{d} \mathbf{v}_j^T \mathbf{X}^T \mathbf{X} \mathbf{v}_j\end{aligned}$$

hence $(\mathbf{v}_1, \ldots, \mathbf{v}_d)$ are the first d eigenvectors of $\mathbf{X}^T\mathbf{X}$: $(\mathbf{w}_1, \ldots, \mathbf{w}_d)$

Mercer's Theorem

Consider the pd kernel $K(x, x')$ on $X \times X$, and the probability distribution $p(x)$ on X.

Define the integral operator L_K

$$(L_K\ f)(x) = \int_X K(x, x')f(x')dp(x').$$

Mercer's Theorem states that

$$K(x, x') = \sum_i \lambda_i \phi_i(x)\phi_i(x')$$

where $(\lambda_i, \phi_i)_i$ is the eigensystem of L_K.

Feature Map

From Mercer's Theorem, the mapping Φ defined over X

$$\Phi(x) = (\sqrt{\lambda_1}\phi_1(x), \sqrt{\lambda_2}\phi_2(x), \ldots)$$

is such that

$$K(x, x') = \Phi(x)^T \Phi(x).$$

- $K(x, x')$ can be interpreted as the dot product in the "feature space".

- given a mapping of X into an Euclidean space, we can construct a pd kernel $X \times X$.

Kernelization

Algorithms that depend on the data, only through the dot products $x_i^T x_j$, can be easily kernelized:

1. Choose pd kernel $K(\cdot, \cdot)$

2. Replace $x_i^T x_j$ with $K(x_i, x_j)$

Example: PCA can be kernelized computing the eigenvectors of the matrix

$$\mathbf{M}_{ij} = K(x_i, x_j)$$

instead of those of the matrix $\mathbf{X}^T\mathbf{X}$.

ISOMAP [*]

- **Assumption:** the *support* of the marginal distribution $p(x)$ is a *convex region of* $\mathbb{R}^d$ (our manifold $\mathcal{M}$) isometrically embedded in $\mathbb{R}^D$.

- **Goal:** constructing a map $\Phi : \mathbb{R}^D \to \mathbb{R}^d$ which "transforms" geodesic distances in $\mathcal{M}$ into Euclidean distances in $\mathbb{R}^d$

- **Construction 1:** approximate the matrix $\mathbf{d}_{\mathcal{M}}$ of pairwise geodesic distances between data points, estimating the *shortest distances* d_{ij} over the *neighborhood graph*.

[*] Tenenbaum, et al, 00

- **Construction 2:** compute the $u \times u$ "kernel matrix"

$$\mathbf{K} = -\frac{1}{2}\mathbf{HDH}, \qquad \mathbf{H} = \mathbf{I} - \frac{1}{u}\mathbb{1}\mathbb{1}^T,$$

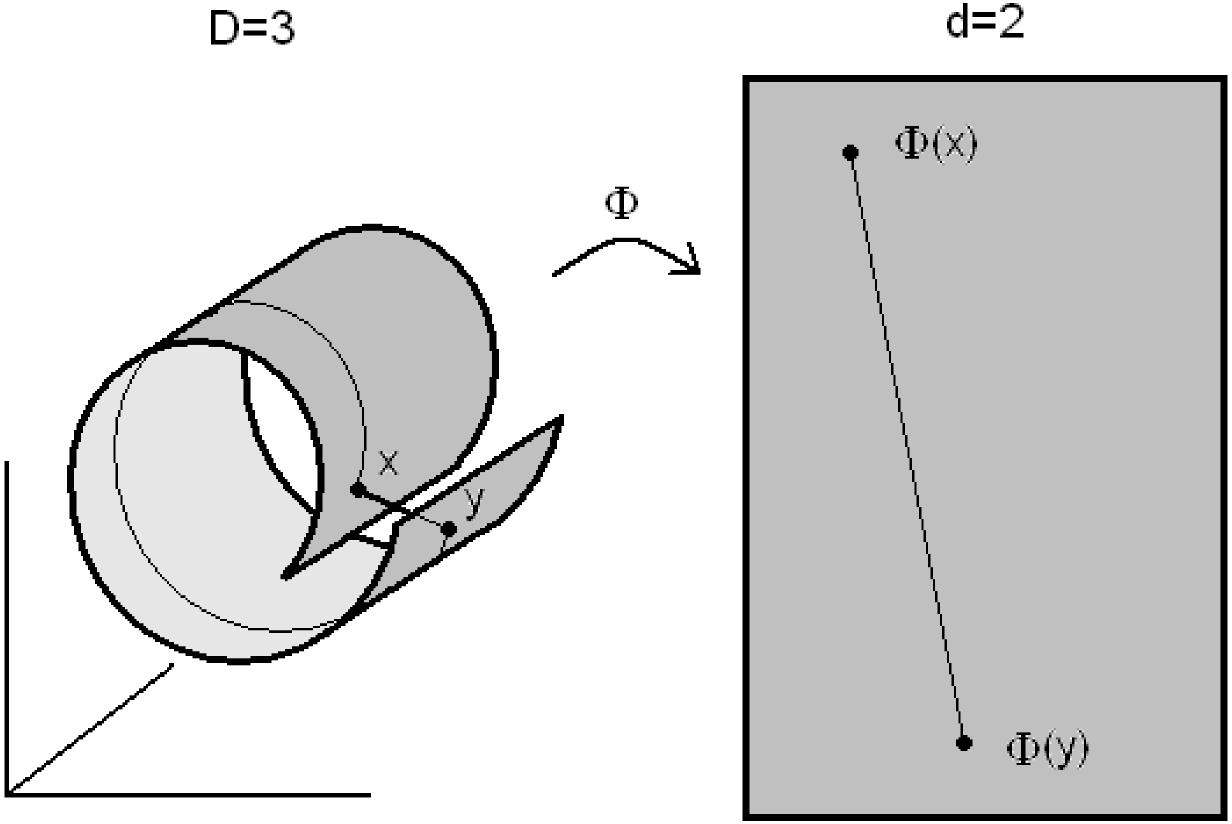
with $\mathbb{1}$ the u-dimensional column vector $(1, 1, \ldots, 1)$, and $\mathbf{D}$ the matrix of squared distances, that is: $\mathbf{D}_{ij} = d_{ij}^2$.

- **Result:** let $(\lambda_a, \mathbf{u}_a)_{a=1}^u$ be the eigensystem of $\mathbf{K}$. The embedding Φ, of $\{x_i\}_{i=1}^u$

$$\Phi(x_i) = (\sqrt{\lambda_1}(\mathbf{u}_1)_i, \sqrt{\lambda_2}(\mathbf{u}_2)_i, \ldots, \sqrt{\lambda_d}(\mathbf{u}_d)_i),$$

is the isometry we were looking for.

ISOMAP global isometry



$$K(x,y)=\Phi(x)^T\Phi(y)$$

Explaining ISOMAP

Firstly, we have to verify that the matrix $\mathbf{K}$ is a genuine pd kernel on the data points.

1. **Symmetry:** since both $\mathbf{H}$ and $\mathbf{D}$ are symmetric, $\mathbf{K} = -\frac{1}{2}\mathbf{H}^T\mathbf{D}\mathbf{H}$, hence $\mathbf{K}^T = \mathbf{K}$.

2. **Positivity:** Note that, by assumption, there exist vectors $\{\phi_i\}_{i=1}^u$, such that $d_{ij} = \|\phi_i - \phi_j\|$. For all $\mathbf{c} = (c_1, \ldots, c_u)$, defining ing $\mathbf{c}' = \mathbf{c} - \frac{1}{u}\sum_{i=1}^u c_i \mathbb{1}$, we get

$$\begin{aligned} \mathbf{c}^T\mathbf{K}\mathbf{c} &= -\frac{1}{2}(H\mathbf{c})^T\mathbf{D}(H\mathbf{c}) = -\frac{1}{2}\mathbf{c}'^T\mathbf{D}\mathbf{c}' \\ [\mathbf{D}_{ij} = \|\phi_i - \phi_j\|^2] &= -\frac{1}{2}\sum_{ij} c_i'(\phi_i^T\phi_i + \phi_j^T\phi_j - 2\phi_i^T\phi_j)c_j' \\ [\textstyle\sum_i c_i' = 0] &= (\sum_i c_i'\phi_i)^T(\sum_i c_i'\phi_i) \geq 0. \end{aligned}$$

Explaining ISOMAP (cont.)

- We must prove that the pd kernel $\mathbf{K}_{ij}$ induces the correct pairwise distances d_{ij} between data points

  $$d_{ij}^2 = \mathbf{K}_{ii} + \mathbf{K}_{jj} - 2\mathbf{K}_{ij}.$$

  This can be verified by direct computation.

- By Mercer's Theorem, the *feature map*

  $$\Phi_0(x_i) = (\sqrt{\lambda_1}(\mathbf{u}_1)_i, \sqrt{\lambda_2}(\mathbf{u}_2)_i, \ldots, \sqrt{\lambda_u}(\mathbf{u}_u)_i),$$

  is an isometry. If the manifold $\mathcal{M}$ is d-dimensional, $\lambda_a = 0$ for $a > d$, and we can use the truncated mapping Φ.

Hessian Locally Linear Embedding (HLLE)*

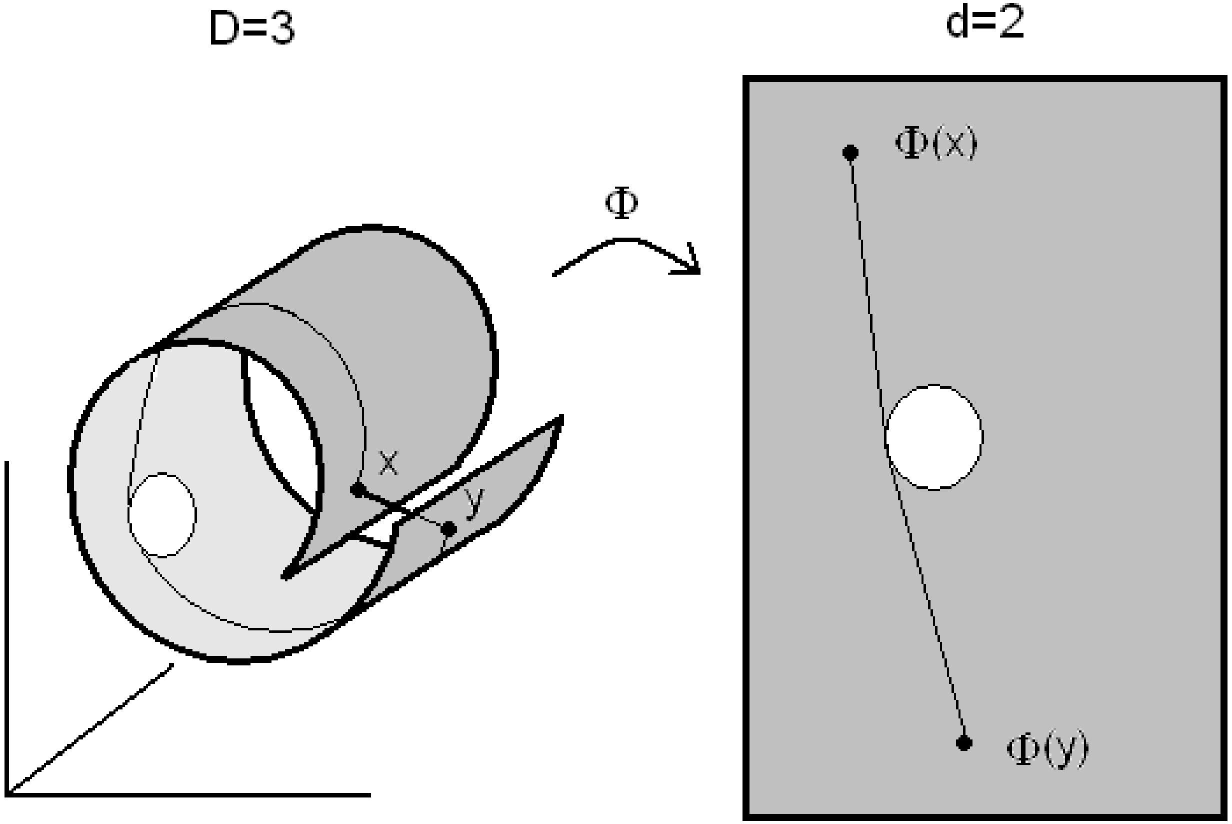
ISOMAP outputs an embedding of the data points $\{x_i\}_{i=1}^{u}$ into $\mathbb{R}^d$, attempting to preserve pairwise distances on the underlying manifold $\mathcal{M}$. The method gives guarantees of convergence if $\mathcal{M}$ is isometric to a convex region in $\mathbb{R}^d$.

Convexity is a very strong hypothesis. Typically, linear combinations of images are not reasonable images!

HLLE gives guarantees of convergence while relaxing the convexity hypothesis.

*Hessian Eigenmaps; Donoho, Grimes 03

HLLE local isometry



Core idea of HLLE

For every point $x \in \mathcal{M}$ and system of coordinates $(\xi_1, \ldots, \xi_d)$ on its tangent space, the Hessian at x of a function $f : \mathcal{M} \to \mathbb{R}$, is the matrix of second derivatives

$$(H_f(x))_{ij} = \frac{\partial}{\partial \xi_i} \frac{\partial}{\partial \xi_j} f(x), \qquad i, j = 1, \ldots, d$$

The core idea of HLLE is that the null space of the quadratic form

$$\mathcal{H}(f) = \int_{\mathcal{M}} \sum_{ij} (H_f(x))^2_{ij}$$

is **independent** of the choice of local coordinates ξ_i.

The null space of $\mathcal{H}$ is the d-dimensional linear space spanned by the *global cartesian coordinates*

Computing the Hessian

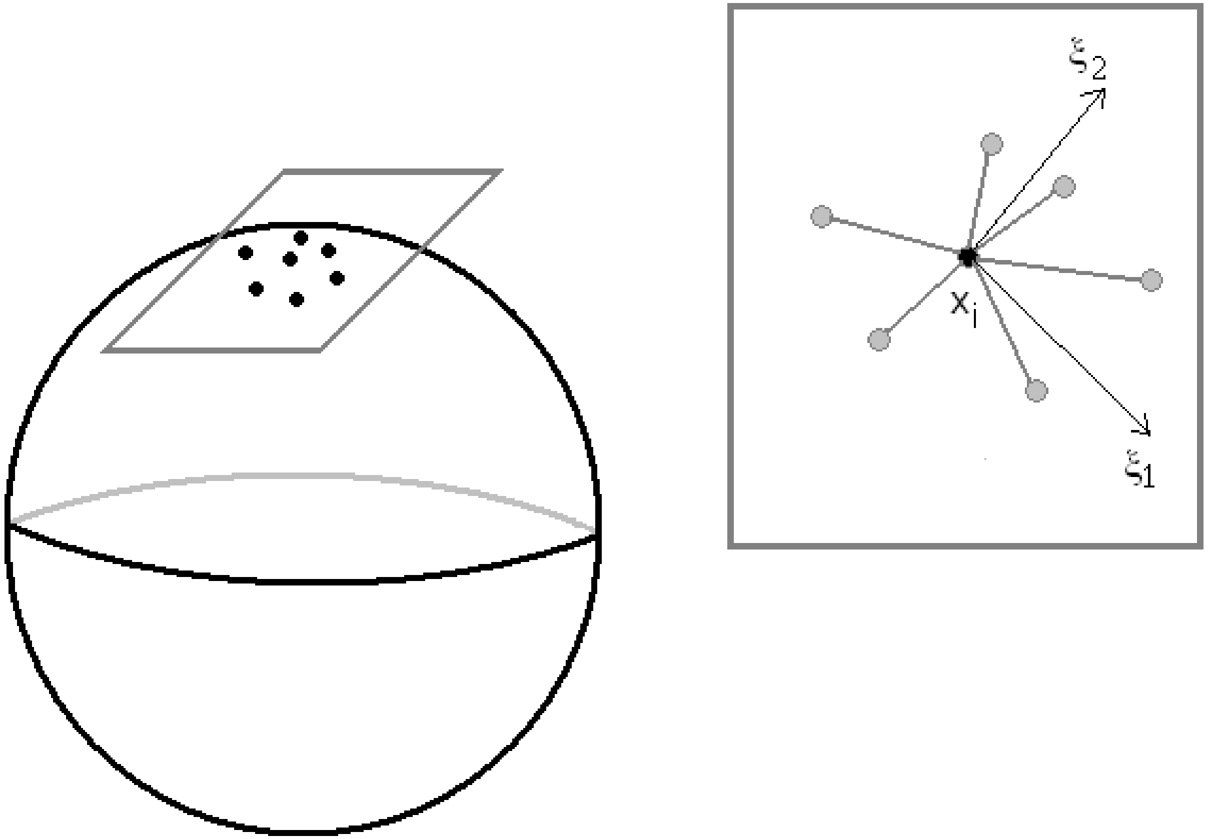
In order to implement this idea, HLLE has to evaluate the quadratic form $\mathcal{H}$ using the data points x_i.

1. construct proxies for the tangent spaces using the k-nearest neighborhood graph

2. implement a finite differences scheme to evaluate second derivatives

3. compute eigensystem of approximation of $\mathcal{H}$. Use d eigenvectors with smallest eigenvalues as embedding coordinates.

Local Linear Neighborhood



Laplacian based methods [*]

Unsupervised methods based on the eigensystem of the Laplacian on the neighborhood graph with weights W_{ij}.

- **Dimensionality Reduction:** consider the solutions of the eigenvector problem $(0 = \lambda_0 \leq \lambda_1 \leq \cdots \leq \lambda_{u-1})$

  $$\mathbf{L}\mathbf{f}_a = \lambda_a \mathbf{D}\mathbf{f}_a$$

  where $\mathbf{D} = diag(D_{11}, \ldots, D_{uu})$. The considered embedding into the d-dimensional Euclidean space is

  $$\Phi(x_i) = ((\mathbf{f}_1)_i, \ldots, (\mathbf{f}_d)_i).$$

- **Spectral Clustering:** use sign of components $(\mathbf{f}_1)_j$ to define two clusters: *connection to min cut problem*.

[*] Belkin, Niyogi, 02