BLG453E COMPUTER VISION

Fall 2018 Term

Week 14

Istanbul Technical University
Computer Engineering Department

Instructor: Prof. Gözde ÜNAL

Teaching Assistant: Enes ALBAY

Learning Outcomes of the Course

Students will be able to:

1. Discuss the main problems of computer (artificial) vision, its uses and applications
2. Design and implement various image transforms: point-wise transforms, neighborhood operation-based spatial filters, and geometric transforms over images
3. Define and construct segmentation, feature extraction, and visual motion estimation algorithms to extract relevant information from images
4. Construct least squares solutions to problems in computer vision
5. Describe the idea behind dimensionality reduction and how it is used in data processing
6. Apply object and shape recognition approaches to problems in computer vision

Week : Dimensionality Reduction and its use in Computer Vision

At the end of Week: Students will be able to:

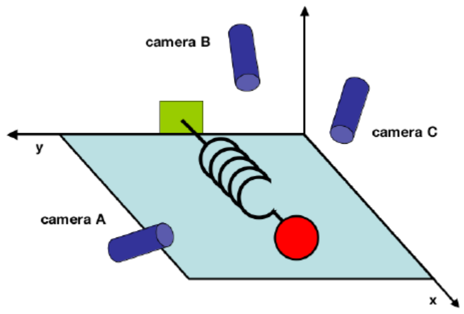
5. Describe the idea behind dimensionality reduction and how it is used in data processing

Dimension: no of variables measured on each observation

Intuition: Not all the measured variables are “important” for understanding the underlying phenomena of interest

Example Toy Problem

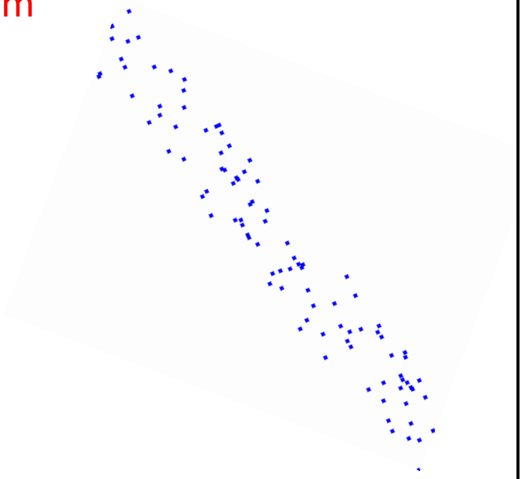
- Suppose, want to study motion of the *ideal spring*: Ball of mass m attached to it, stretch the spring, it will oscillate indefinitely along the x-axis



❑ Say we record the ball’s 2D position from three cameras for 10 mins at 120Hz, we have 10*60*120=72,000 measurements or observations

Example Toy Problem

Q: What is the data dimensionality ?



- In fact, the spring travels in a straight line: →any spread deviating from the straight line must be noise
- Hence, directions with largest variances in our measurement vector space contains the dynamics of interest

Dimensionality Reduction

64x64 sized images → dimension = 4096

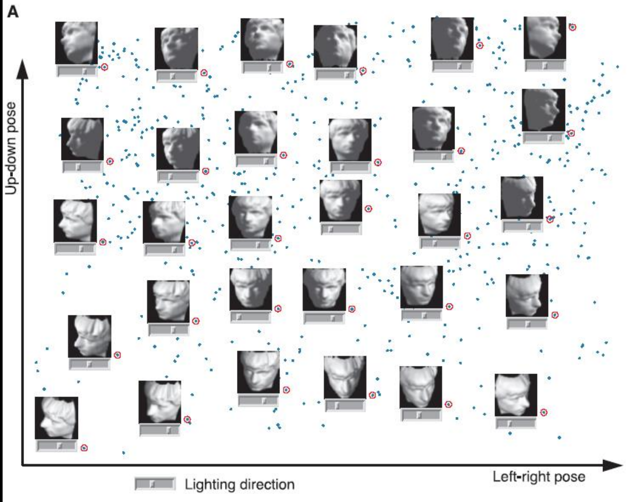
From face database: olivettifaces

Dimensionality Reduction

- Need to analyze large amounts multivariate data:
  - Human Faces, Medical images, speech signals
  - Linguistics: Syntactic language analysis
  - Climate and atmospheric patterns and data analysis
  - Gene Distributions
- Difficult to visualize data in dimensions just greater than three.
- Discover compact representations of high dimensional data.
  - Better Modeling and Recognition
  - Probably meaningful dimensions
  - Visualization
  - Compression

Typically, if 2-3 dimensions are enough to explain the variability in the data, we can do a visual analysis

A



For example:

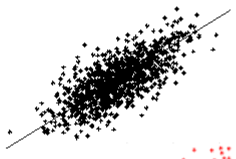
- 64X64 Input Images form

  4096-dimensional vectors
- Intrinsically, three dimensions is enough for presentations:
- Two pose parameters and azimuthal lighting angle

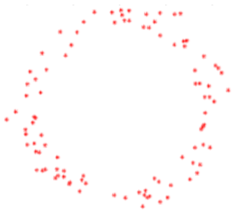
Tennenbaum|Silva|Langford: “A Global Geometric Framework for Nonlinear Dimensionality Reduction (Isomap)”

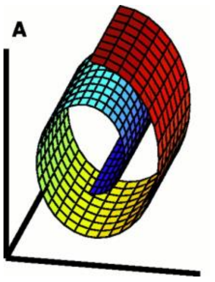
Types of Structure in Multivariate Data

– Linear

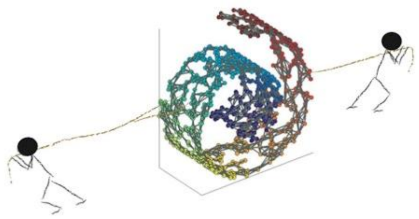


– Non-Linear





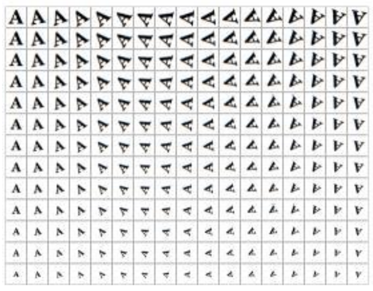
Q: Can you unroll the non-linear data to a simpler structure?



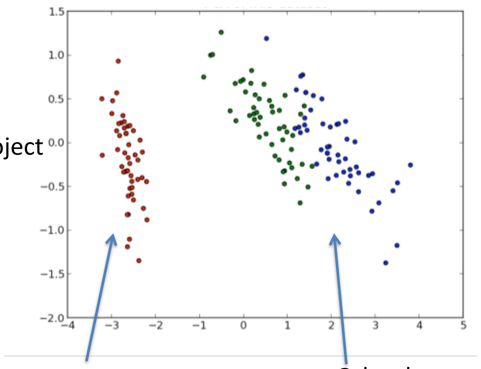
http://www.cse.wustl.edu/~kilian/research/manifold/manifold.html

Concept of Dimensionality Reduction:

Embed data in a higher dimensional space to a lower dimensional manifold



Project



Question: Are there projections that can produce this 2D mapping?

Dimensionality Reduction

Goal:

High-dimensional observations/data are projected onto "meaningful" low-dimensional space

- Classical techniques
  - Principle Component Analysis—maximizes/preserves the variance
  - Multidimensional Scaling—preserves inter-point distances

Overview

- **Linear Dimensionality Reduction**

  Principal Component Analysis (PCA)

  Multidimensional Scaling (MDS)
- **Applications of PCA**
- Nonlinear Dimensionality Reduction (advanced topic, we'll cover briefly if time permits)
  - Isomap
  - Locally Linear Embedding
  - Laplacian Embedding

References:

General Ref book: E. Alpaydın, "Introduction to Machine Learning", 2010, Chapter 6

– Tennenbaum&Silva&Langford	[Isomap]
– Roweis&Saul	[Locally Linear Embedding]
– Belkin&Niyogi	[Laplacian Eigenmaps]

Idea in Dimensionality Reduction:

Linear Approach:

want to find a mapping y = W^T x, with a linear transformation:
W is kxd dimensions, k << d

$$\mathbf{y} = \mathbf{W}^T \mathbf{x} \qquad \mathbf{W} = [\mathbf{w}_1 \quad \mathbf{w}_2 \quad \dots \quad \mathbf{w}_k]$$

i.e. write the new variable y (in a low dimension) as a linear combination of original variables:

$$y_i = w_{i1}x_1 + w_{i2}x_2 + \dots + w_{id}x_d, \quad i = 1, \dots, k$$

$$y_i = \mathbf{w}_i^T \mathbf{x}$$

Note: Each x is d-dimensional vector, y is k-dimensional vector

Linear Dimensionality Reduction:

Derive on board

Overview of Principal Component Analysis

- Principal component analysis (PCA) is a classical way to reduce data dimensionality
- PCA projects high dimensional data to a lower dimension
- PCA projects the data in the least square sense– it captures big (principal) variability in the data and ignores other small variabilities

Principal Component Analysis (PCA)

$$\mathbf{X}_{d\times N} = \begin{bmatrix} \mathbf{x}_1 & \mathbf{x}_2 & \cdots & \mathbf{x}_N \end{bmatrix}$$

These are Centered Data Points, i.e. mean is subtracted from each data point:

$$\mathbf{X}_i \rightarrow \mathbf{X}_i - \mathbf{X}_{mean}$$

Calcuate Covariance matrix S of the data:

$$\mathbf{S} = \mathbf{X}\mathbf{X}^T$$

Perform Eigen Value Decomposition on Data Covariance matrix S, which is symmetric :

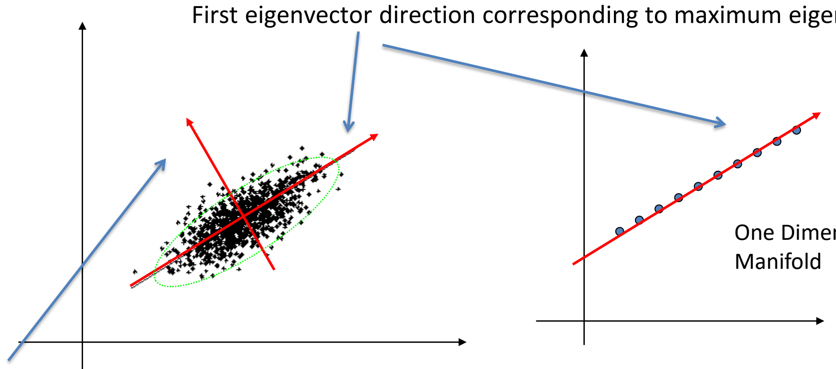
$$\mathbf{S} = \mathbf{V}\mathrm{L}\,\mathbf{V}^T$$

Eigenvector matrix

Diagonal Eigenvalue matrix

Principal Component Analysis (PCA)

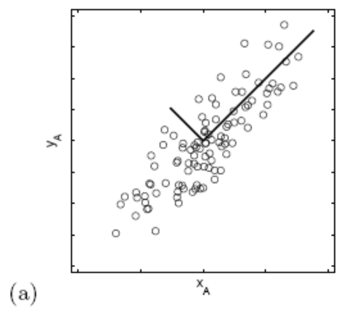
First eigenvector direction corresponding to maximum eigenvalue



Second eigenvector direction corresponding to second maximum eigenvalue (if it is 2D data, this is the only 2nd eigenvalue)

→Maximizing the data variance corresponds to

Finding the appropriate rotation of the canonical basis



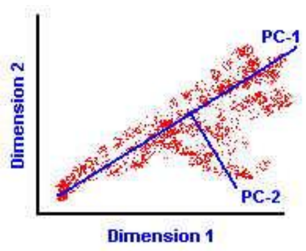


Fig (a): Independent data: one can not predict r1 from r2 (e.g. plot of x_A vs. Humidity)

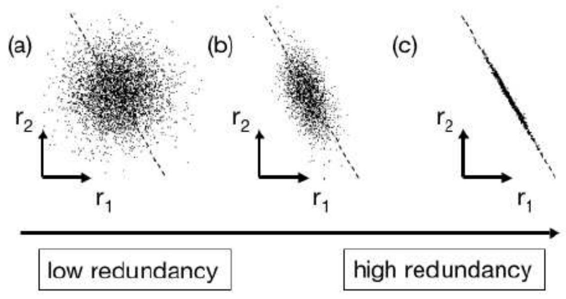


FIG. 3 A spectrum of possible redundancies in data from the two separate recordings r_1 and r_2 (e.g. x_A, y_B). The best-fit line $r_2 = kr_1$ is indicated by the dashed line.

PCA: Mathematical Derivation – Least Squares (You are not responsible from this derivation)

Let us say we have x_i, i=1…N data points in *p* dimensions (*p* is large)

If we want to represent the data set by a single point x_0, then

$$\mathbf{x}_0 = \mathbf{m} = \frac{1}{N}\sum_{i=1}^{N}\mathbf{x}_i \quad \longleftarrow \text{Sample mean}$$

Can we justify this choice mathematically?

$$J_0(\mathbf{x}_0) = \sum_{i=1}^{N}\left\|\mathbf{x}_i - \mathbf{x}_0\right\|^2$$

It turns out that if you minimize J_0, you get the above solution, *i.e,* sample mean

PCA: Mathematical Derivation

Representing the data set x$_i$, i=1…*N* by its mean is quite uninformative

So lets try to represent the data by a straight line of the form:

$$\mathbf{x} = \mathbf{m} + a\mathbf{e}$$

This is equation of a straight line that says that it passes through m

e is a unit vector along the straight line

The training points projected on this straight line would be

$$\mathbf{x}_i = \mathbf{m} + a_i\mathbf{e}, \quad i = 1 \ldots N$$

PCA: Mathematical Derivation

Let's now determine a$_i$'s

$$\boxed{J_1(a_1, a_2, \ldots, a_N, \mathbf{e}) = \sum_{i=1}^{N} \left\| \mathbf{m} + a_i\mathbf{e} - \mathbf{x}_i \right\|^2}$$

Expand

$$J_1 = \sum_{i=1}^{N} a_i^2 \|\mathbf{e}\|^2 - 2\sum_{i=1}^{N} a_i \mathbf{e}^T(\mathbf{x}_i - \mathbf{m}) + \sum_{i=1}^{N} \|\mathbf{x}_i - \mathbf{m}\|^2$$

$$= \sum_{i=1}^{N} a_i^2 - 2\sum_{i=1}^{N} a_i \mathbf{e}^T(\mathbf{x}_i - \mathbf{m}) + \sum_{i=1}^{N} \|\mathbf{x}_i - \mathbf{m}\|^2$$

Partially differentiating with respect to a_i we get:

$$a_i = \mathbf{e}^T(\mathbf{x}_i - \mathbf{m})$$

Plugging in this expression for a$_i$ in J$_1$ (3rd line above) we get:

$$J_1(\mathbf{e}) = -\sum_{i=1}^{N} \mathbf{e}^T(\mathbf{x}_i - \mathbf{m})(\mathbf{x}_i - \mathbf{m})^T\mathbf{e} + \sum_{i=1}^{N} \|\mathbf{x}_i - \mathbf{m}\|^2 = -\mathbf{e}^T S\mathbf{e} + \sum_{i=1}^{N} \|\mathbf{x}_i - \mathbf{m}\|^2$$

where $\boxed{S = \sum_{i=1}^{N} (\mathbf{x}_i - \mathbf{m})(\mathbf{x}_i - \mathbf{m})^T}$ is called the sample covariance matrix

PCA: Mathematical Derivation

So minimizing J_1 is equivalent to maximizing: $\mathbf{e}^T S\mathbf{e}$

Subject to the constraint that e is a unit vector: $\mathbf{e}^T\mathbf{e} = 1$

Use Lagrange multiplier method to form the objective function:

$$\max_{\mathrm{e}} \quad \mathbf{e}^T S\mathbf{e} - \lambda(\mathbf{e}^T\mathbf{e} - 1)$$

Differentiate to obtain the equation: $2S\mathbf{e} - 2\lambda\mathbf{e} = \mathbf{0} \; or \; \mathbf{S}\mathbf{e} = \lambda\mathbf{e}$

Solution is that e is the eigenvector of S corresponding to the largest eigen value

PCA: Mathematical Derivation (Extra for interested)

The preceding analysis can be extended in the following way.

Instead of projecting the data points on to a straight line, we may

now want to project them on a d-dimensional plane of the form:

$$\mathbf{x} = \mathbf{m} + a_1\mathbf{e}_1 + \cdots + a_d\mathbf{e}_d$$

d is much smaller than the original dimension *p*

In this case one can form the objective function: $J_d = \sum_{i=1}^{N} \| (\mathbf{m} + \sum_{k=1}^{d} a_{ik}\mathbf{e}_k) - \mathbf{x}_i \|^2$

It can also be shown that the vectors e_1, e_2, …, e_d are *d* eigenvectors

corresponding to *d* largest eigen values of the scatter matrix = sample covariance

PCA: Summary

- Reduce the number of dimensions of the data points "x_i" to k << d, where d is the dimension of points in the original space
- Search in R^d for the direction of the unit vector v such that the projection of the set of N data points x_n (n=1,...N) to this direction leads to the scatter of N points with highest dispersion
- To keep 1 component, pick the one that best separates all the points, ie.has the highest variance: This is achieved by picking the eigenvector of largest eigenvalue
- You can keep d components by picking d eigenvectors that correspond to d largest eigen values.

k=1, d=2

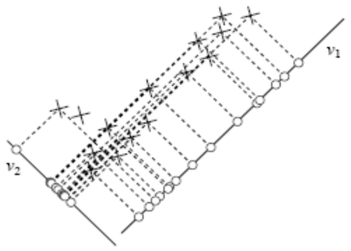


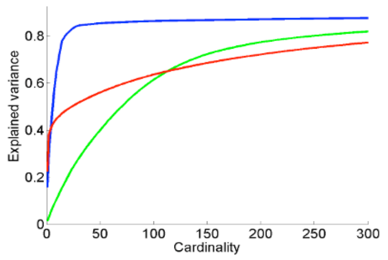
Figure 12.30 – *Projecting the samples for the directions* $\mathbf{v}_1$ *and* $\mathbf{v}_2$*: the dispersion of the projected points is more favorable to an analysis for the vector* $\mathbf{v}_1$ *than it is for* $\mathbf{v}_2$

Explained Variance by the k eigenvalues out of d

Eigenvalues are sorted in descending order $/_1 > /_2 > \ldots > /_k$

Proportion (or percent) of variance=100* $\dfrac{/_1 + /_2 + \ldots + /_k}{/_1 + /_2 + \ldots + /_k + \ldots + /_d}$

Desired: % variance is large while dimension k is much smaller than d

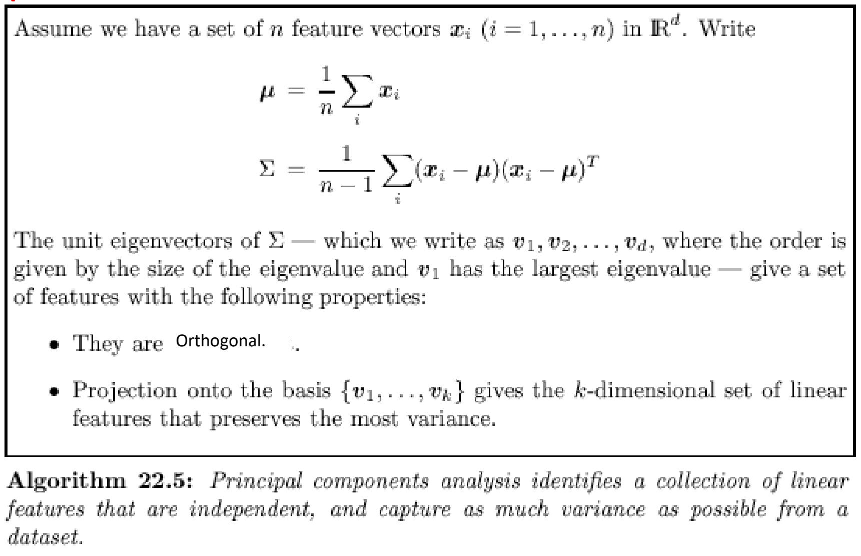


No of eigenvalues

Curves with different colors correspond to different datasets

PCA Applications

PCA



Assume we have a set of n feature vectors $\boldsymbol{x}_i$ $(i = 1, \dots, n)$ in $\mathbb{R}^d$. Write

$$\boldsymbol{\mu} = \frac{1}{n}\sum_i \boldsymbol{x}_i$$

$$\Sigma = \frac{1}{n-1}\sum_i (\boldsymbol{x}_i - \boldsymbol{\mu})(\boldsymbol{x}_i - \boldsymbol{\mu})^T$$

The unit eigenvectors of Σ — which we write as $\boldsymbol{v}_1, \boldsymbol{v}_2, \dots, \boldsymbol{v}_d$, where the order is given by the size of the eigenvalue and $\boldsymbol{v}_1$ has the largest eigenvalue — give a set of features with the following properties:

- They are Orthogonal.
- Projection onto the basis $\{\boldsymbol{v}_1, \dots, \boldsymbol{v}_k\}$ gives the k-dimensional set of linear features that preserves the most variance.

Algorithm 22.5: *Principal components analysis identifies a collection of linear features that are independent, and capture as much variance as possible from a dataset.*

Computer Vision - A Modern Approach
Slides by D.A. Forsyth

Principal Component Analysis

PCA algorithm

Input: Datamatrix X

Output: Eigen Vectors $B_1, \dots, B_k$

1. Compute the average image:

N: # data points Xi that are all aligned

$$\boxed{\overline{X} = \frac{1}{N} \Sigma X_i}$$

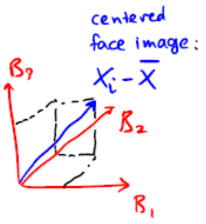
2. Subtract the average from each X_i: $\boxed{Z_i = X_i - \overline{X}}$

3. Define $\boxed{Z = [Z_1 \dots Z_N]}$

4. $\boxed{B_1, \dots, B_k = \text{eigenvectors of matrix } ZZ^T \text{ with the } k \text{ largest eigenvalues}}$

Application: Face recognition & compression using Eigenfaces

- 250x350 pixel image of a face = 75,000-dimensional vector X_i
- $\overline{X}$ = "mean" face image
- $B_1, \dots, B_k$: ($k < 20$ usually) the "eigenfaces"
- Each face image represented as linear combination of eigenfaces

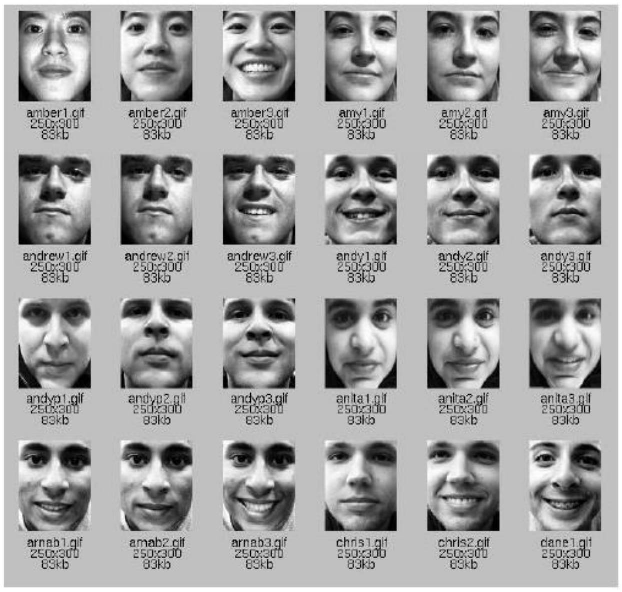


Source: IAPR PCA Lecture Notes

Principal Component Analysis: Results

The input photographs (they are all approximately aligned)

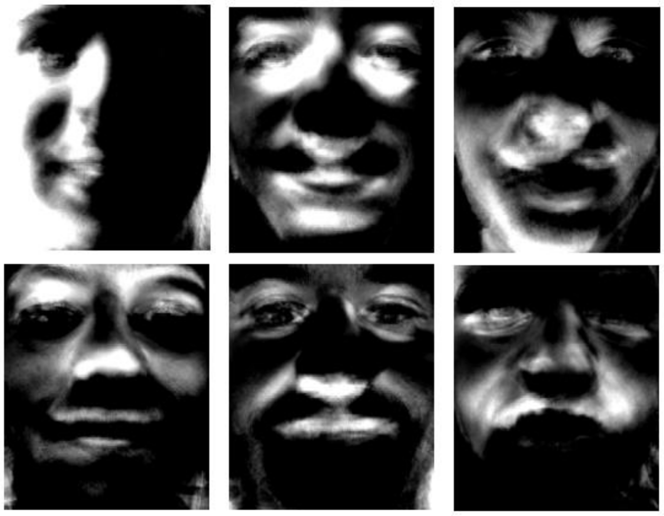
Pre-alignment is important!



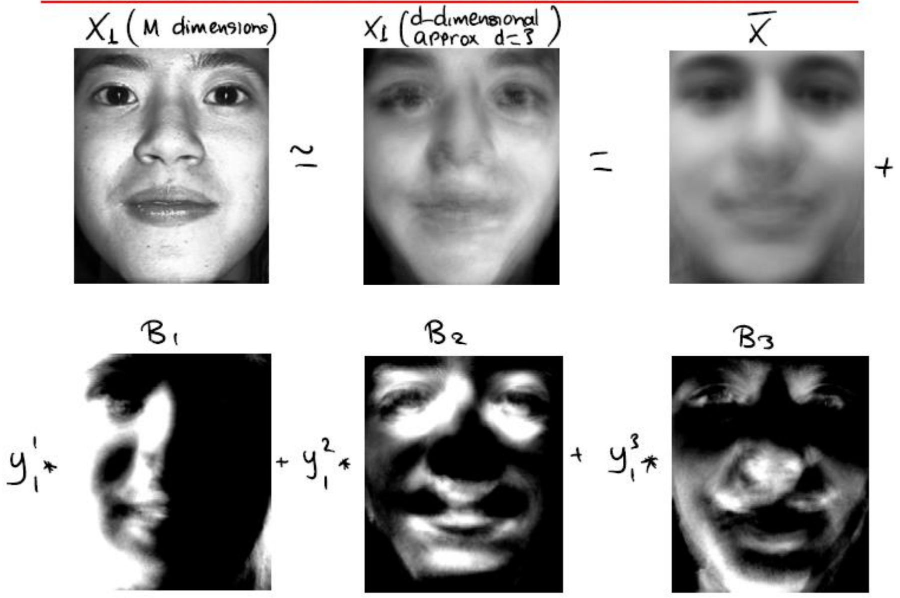
Source: IAPR PCA Lecture Notes

Principal Component Analysis: Results

The top 6 eigenvectors (eigenfaces):

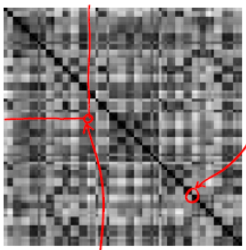


Principal Component Analysis: Results



Face Recognition Using PCA (Eigenfaces)

Distance matrix



yi: coordinates of Yi in the reduced space

Recognition
(Given: Query image T & Database)

$\mathbf{W} = [\mathbf{B}_1 \quad \mathbf{B}_2 \quad \dots \quad \mathbf{B}_k]$: Linear transform matrix

① Compute coordinates of T in basis $B_1, \dots, B_k$

$$\boxed{t^j = \mathbf{W}_j^T(\mathbf{T} - \bar{\mathbf{X}})}$$

② Find the vector Y_i that is closest to vector $\begin{bmatrix} t^1 \\ \vdots \\ t^d \end{bmatrix}$

③ Return face image X_i

$$\mathbf{X}_i = \bar{\mathbf{X}} + \mathbf{W}\mathbf{Y}_i$$

Face Recognition databases: another example

64x64 sized images → dimension = 4096

From face database: olivettifaces

Face Recognition

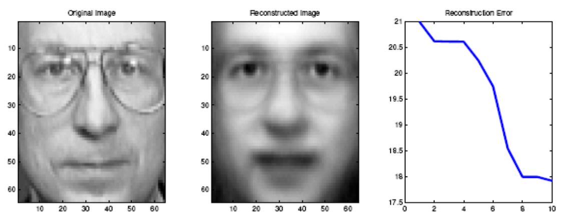


Figure 5: The original image (left) and the reconstructed image (middle) after ten principal components have been employed. The right hand plot shows how the error has decreased for this particular face over the ten PC's employed.

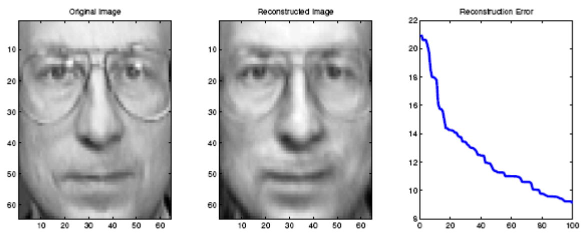
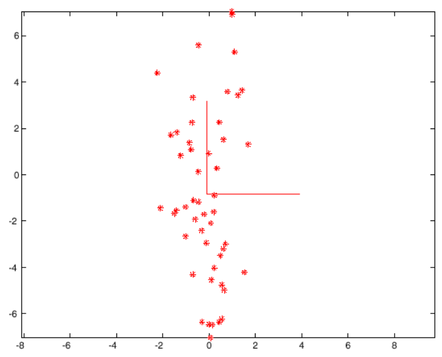
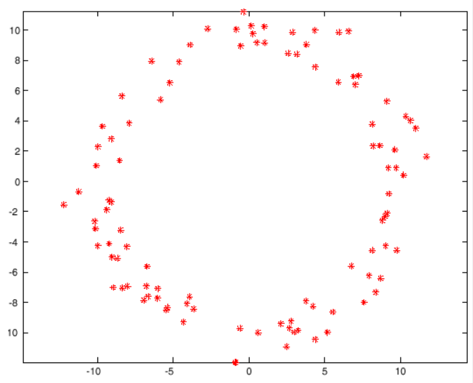


Figure 6: The original image (left) and the reconstructed image (middle) after one hundred principal components have been employed. The right hand plot shows how the error has decreased for this particular face over the one hundred PC's employed.



PCA gives a good representation of this data set



PCA will give a very poor representation of this data set

Difficulties with PCA

- Data may lie on more complex manifolds, e.g. the swiss roll, or the data on previous slide
- Projection may suppress important detail
  - Smallest variance directions may not be unimportant
  - The task we are interested in may not correlate with picking the largest variance directions
- Then you can resort to MDS or Nonlinear Dimensionality reduction techniques (not covered in this class) or other such more advanced techniques

END OF LECTURE

Recall Learning objectives of Week : Students are able to:

LO5: Describe the idea behind dimensionality reduction and how it is used in data processing

LO6: Apply object and shape recognition approaches to problems in computer vision

Work on your last Homework Assignment

EXTRA MATERIAL: Slides on/after this one are for your reference: You are not responsible in our class

- Linear Dimensionality Reduction
  Principal Component Analysis (PCA)
  Multidimensional Scaling (MDS)
- Applications of PCA
- Nonlinear Dimensionality Reduction (advanced topic)
  - Isomap
  - Locally Linear Embedding
  - Laplacian Embedding

Overview: you are responsible from only bold items below

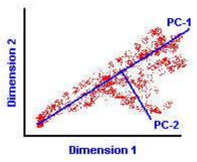
- Linear Dimensionality Reduction

  Principal Component Analysis (PCA)
  Multidimensional Scaling (MDS)

- Applications of PCA
- Nonlinear Dimensionality Reduction
  - Isomap (Tennenbaum&Silva&Langford)
  - Locally Linear Embedding (Roweis&Saul)
  - Laplacian Eigenmaps (Belkin&Niyogi)

Linear Dimensionality Reduction

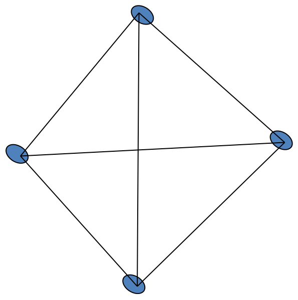
- PCA
  - Finds a low-dimensional embedding of the data points that best preserves their variance as measured in the high-dimensional input space



- MDS
  - Finds an embedding that preserves the inter-point distances, similar to PCA when the points are given rather than distances between points.

Multidimensional Scaling (MDS)

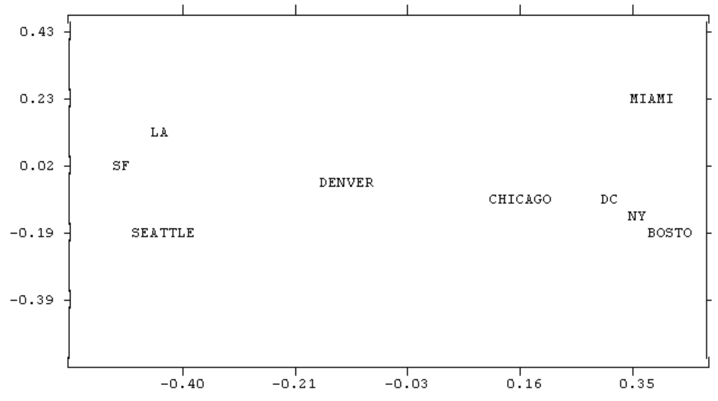
- Here we are given pairwise distances instead of the actual data points
  - First convert the pairwise distance matrix into the dot product matrix XX^T
  - Then, proceed similar to PCA



MDS: Example

```
                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
```

- Given road travel distances between cities, we try to get an approximation to the map
- Map deviates from bird-flight path (Euclidean distance) due to geographical obstacles (lakes, mountains ..)

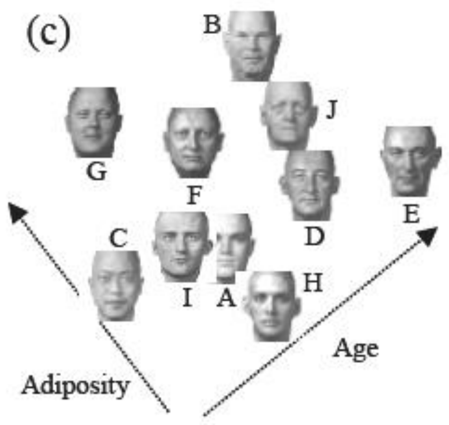


MDS is more general

- When the distances are Euclidean, MDS is equivalent to PCA
- In MDS: Instead of pairwise distances we can use paiwise "dissimilarities".

Eg. Face recognition:

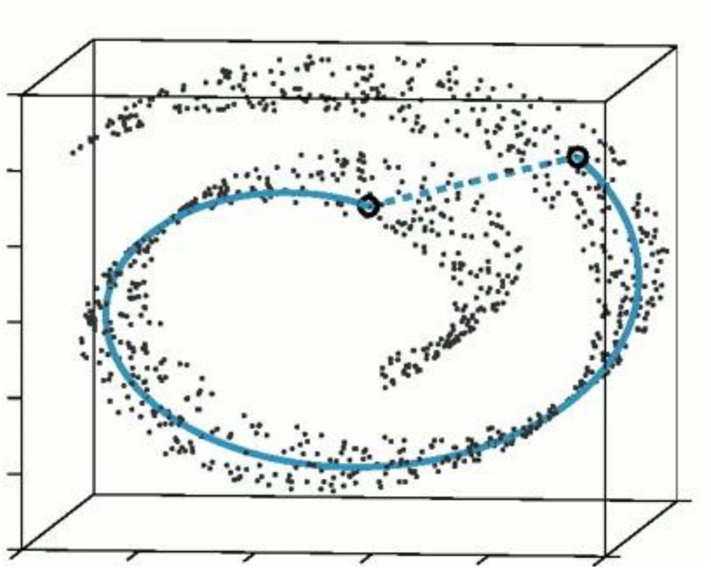
May get some significant cognitive dimensions (not always true)



Nonlinear Dimensionality Reduction

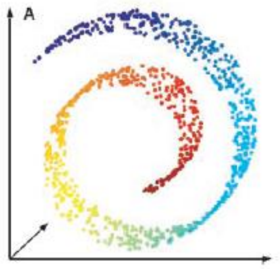
- Many data sets contain essential nonlinear structures that can not be recovered by PCA and MDS
- May need to resort to some nonlinear dimensionality reduction approaches

To preserve structure, preserve the geodesic distance and not the Euclidean distance

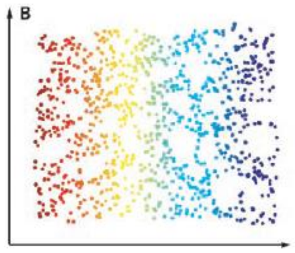


Example of a Nonlinear Structure

Swiss Roll

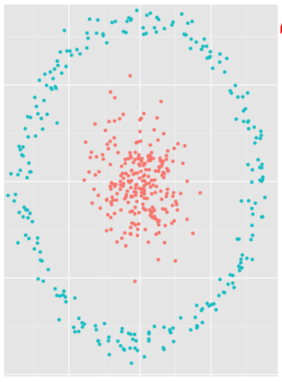


Unfold the nonlinear manifold structure

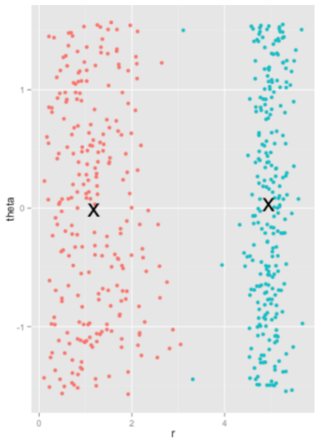


e.g. Obtain an embedding in a low dimensional space

Another Example of a Nonlinear Structure



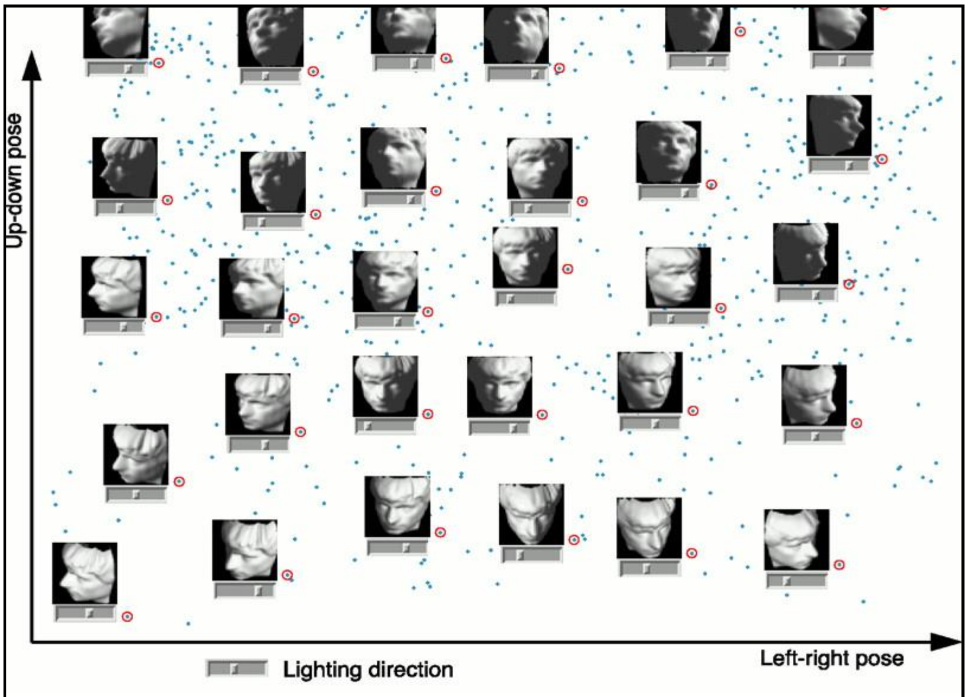
Unfold the nonlinear manifold structure into a linear one:

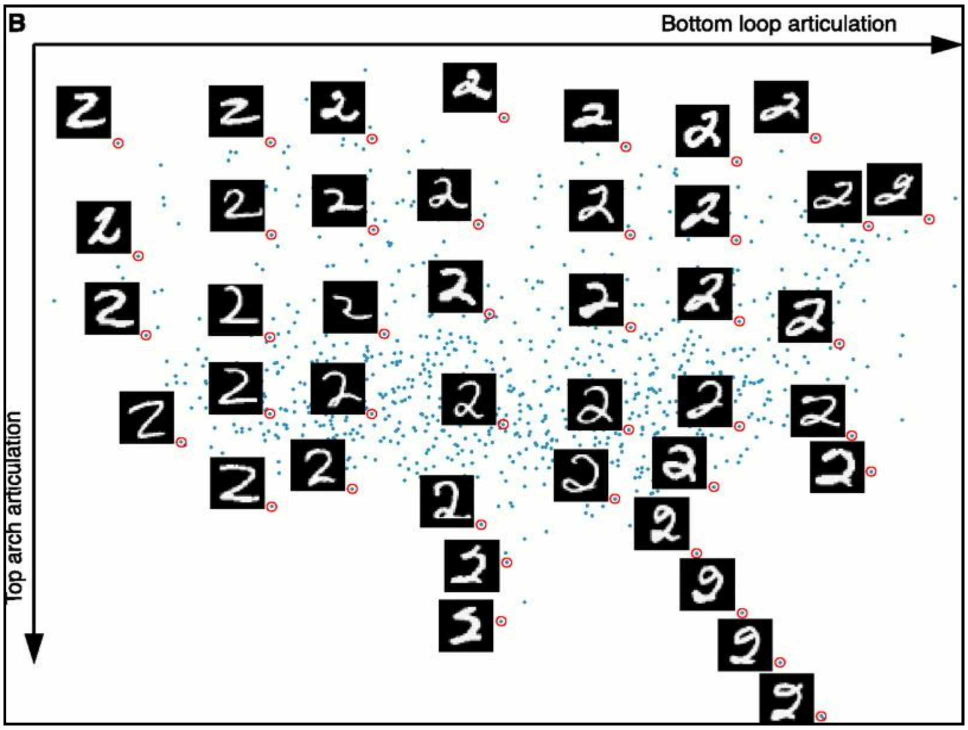


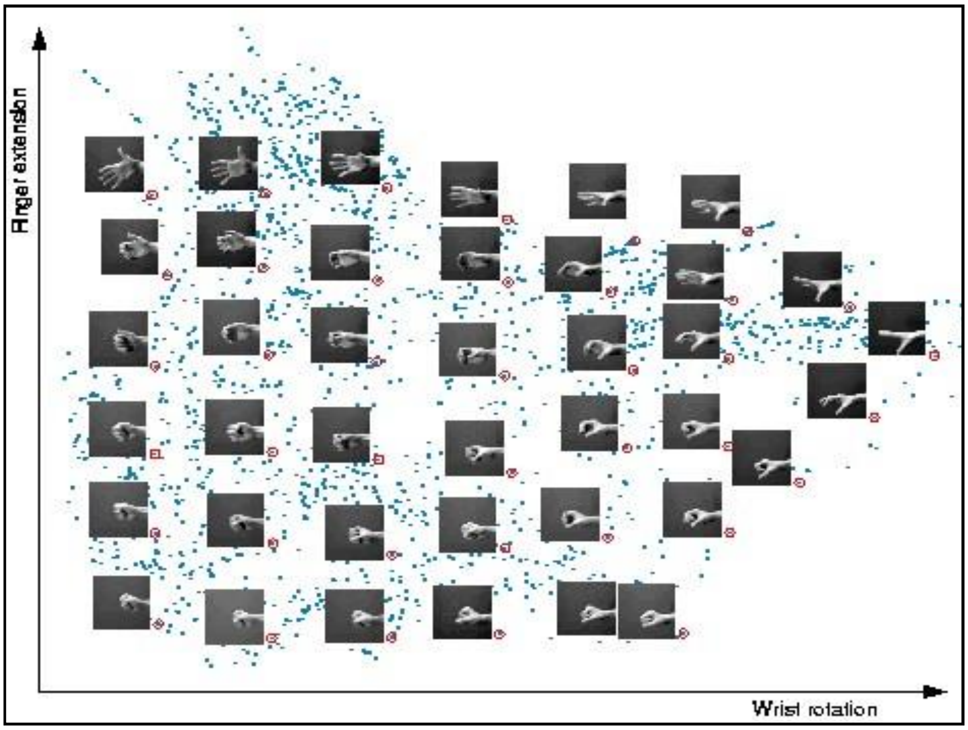
For your future reference: You are not responsible in this class from the following:

State-of-the Art Nonlinear Methods

- Tenenbaum et.al's Isomap Algorithm
  - Global approach: Uses MDS with geodesic distances
  - On a low dimensional embedding
    - Nearby points should be nearby.
    - Faraway points should be faraway.
- Roweis and Saul's Locally Linear Embedding Algorithm
  - Local approach
    - Nearby points nearby
- Belkin and Niyogi's Laplacian Eigenmaps for Dimensionality Reduction and Data Representation, "Neural Computation", 2003; 15(6):1373-1396

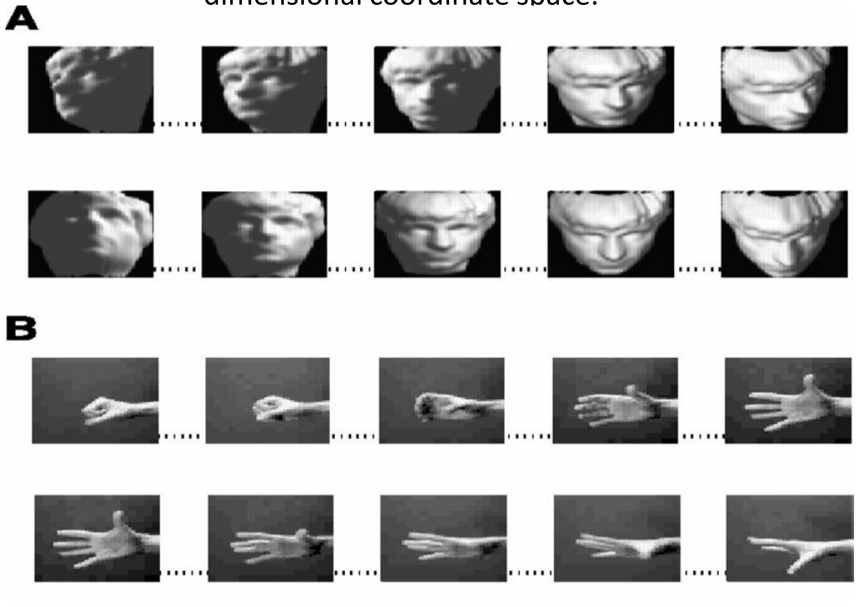



B
Bottom loop articulation
Top arch articulation


Finger extension
Wrist rotation

Example applications

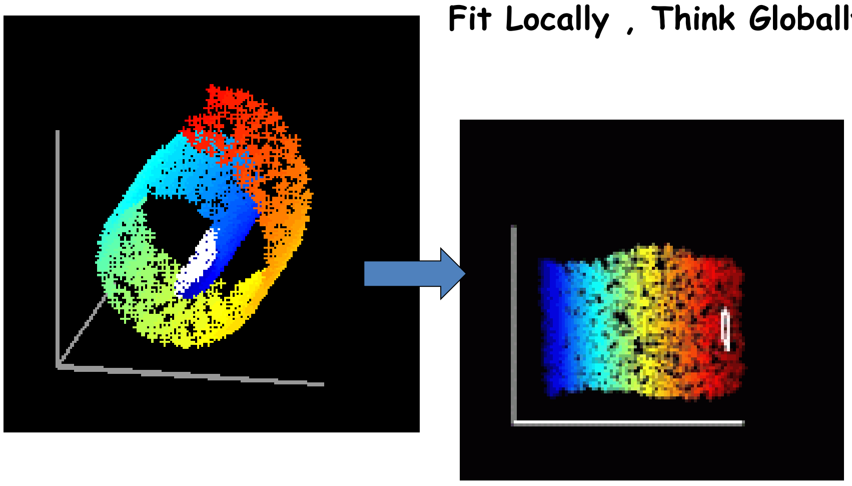
Interpolations between distant points in the low-dimensional coordinate space.



Locally Linear Embedding

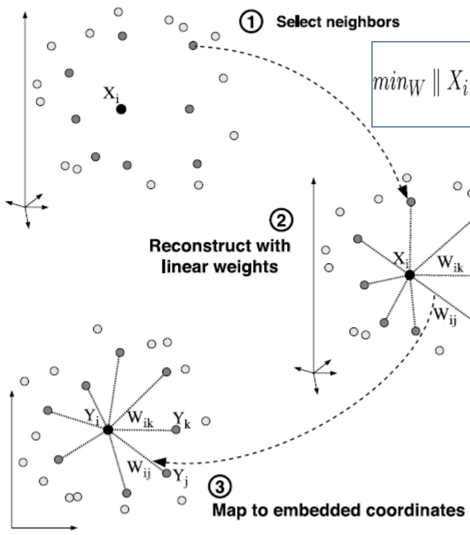
A Manifold is a topological space which is locally Euclidean."

Fit Locally , Think Globally



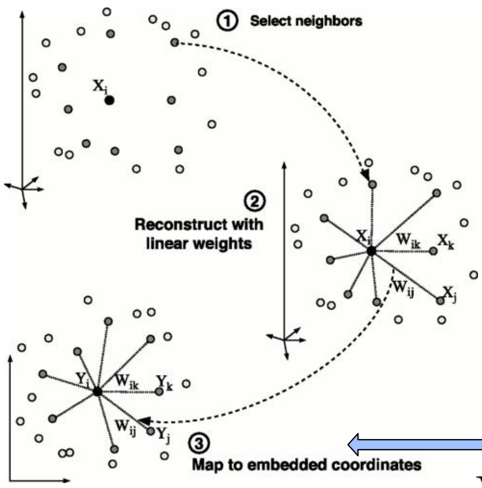
Fit locally …

Fig. 2. Steps of locally linear embedding: (1) Assign neighbors to each data point $\vec{X}_i$ (for example by using the K nearest neighbors). (2) Compute the weights W_{ij} that best linearly reconstruct $\vec{X}_i$ from its neighbors, solving the constrained least-squares problem in Eq. 1. (3) Compute the low-dimensional embedding vectors $\vec{Y}_i$ best reconstructed by W_{ij}, minimizing Eq. 2 by finding the smallest eigenmodes of the sparse symmetric matrix in Eq. 3. Although the weights W_{ij} and vectors Y_i are computed by methods in linear algebra, the constraint that points are only reconstructed from neighbors can result in highly nonlinear embeddings.


1 Select neighbors
X_i
2 Reconstruct with linear weights
X_i W_{ik} X_k W_{ij} X_j
3 Map to embedded coordinates
Y_i W_{ik} Y_k W_{ij} Y_j

$$min_W \parallel X_i - \sum_{j=1}^{K} W_{ij}X_j \parallel^2 \qquad (1)$$

Nonlinear Dimensionality Reduction by Locally Linear Embedding. Sam T. Roweis, *et al. Science 290, 2323 (2000);*


1 Select neighbors
X_i
2 Reconstruct with linear weights
X_i W_{ik} X_k W_{ij} X_j
3 Map to embedded coordinates
Y_i W_{ik} Y_k W_{ij} Y_j

Think Globally…

$$min_Y \sum_{i=1}^{N} \parallel Y_i - YW_i \parallel^2$$

$$Y_{d\times N} = [Y_1|Y_2|...|Y_N]$$

Properties of Locally Linear Embedding Method (Not linear globally)

- ❑ The same weights that reconstruct the data points in d-dimensions should reconstruct it in the manifold in k- dimensions
  - The weights characterize the intrinsic geometric properties of each neighborhood
- ❑ The weights that minimize the reconstruction errors are invariant to rotation, rescaling and translation of the data points
  - Invariance to translation is enforced by adding the constraint that the weights sum to one

Examples : 2-D embedding of faces

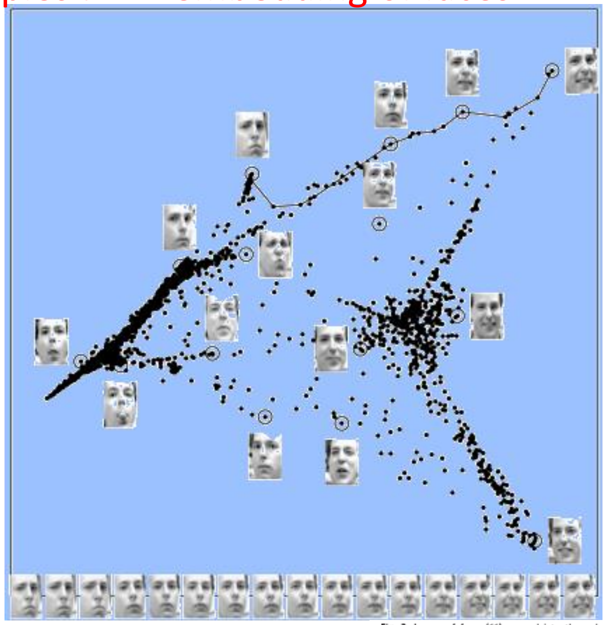
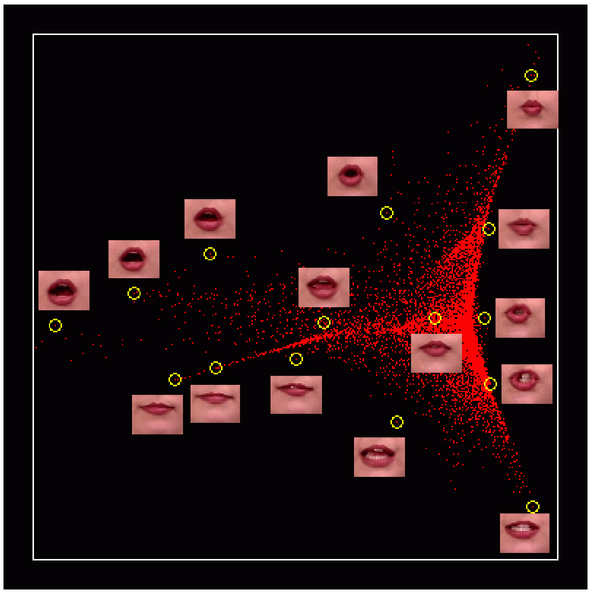


Fig. 3. Images of faces (11) mapped into the embedding space described by the first two coordinates of LLE. Representative faces are shown next to circled points in different parts of the space. The bottom images correspond to points along the top-right path (linked by solid line), illustrating one particular mode of variability in pose and expression.



Short circuit problem

There is a free parameter:
How many neighbours?

- How to choose neighborhoods:

Susceptible to short-circuit errors
if neighborhood is larger than the folds in the manifold

If nbhd is small, we get isolated patches