

[70240413 Statistical Machine Learning, Spring, 2017]

# **Unsupervised Learning (II)**

## **Dimension Reduction**

**Jun Zhu**

dcszj@mail.tsinghua.edu.cn

<http://bigml.cs.Tsinghua.edu.cn/~jun>

State Key Lab of Intelligent Technology & Systems

Tsinghua University

April 11, 2017

# Outline

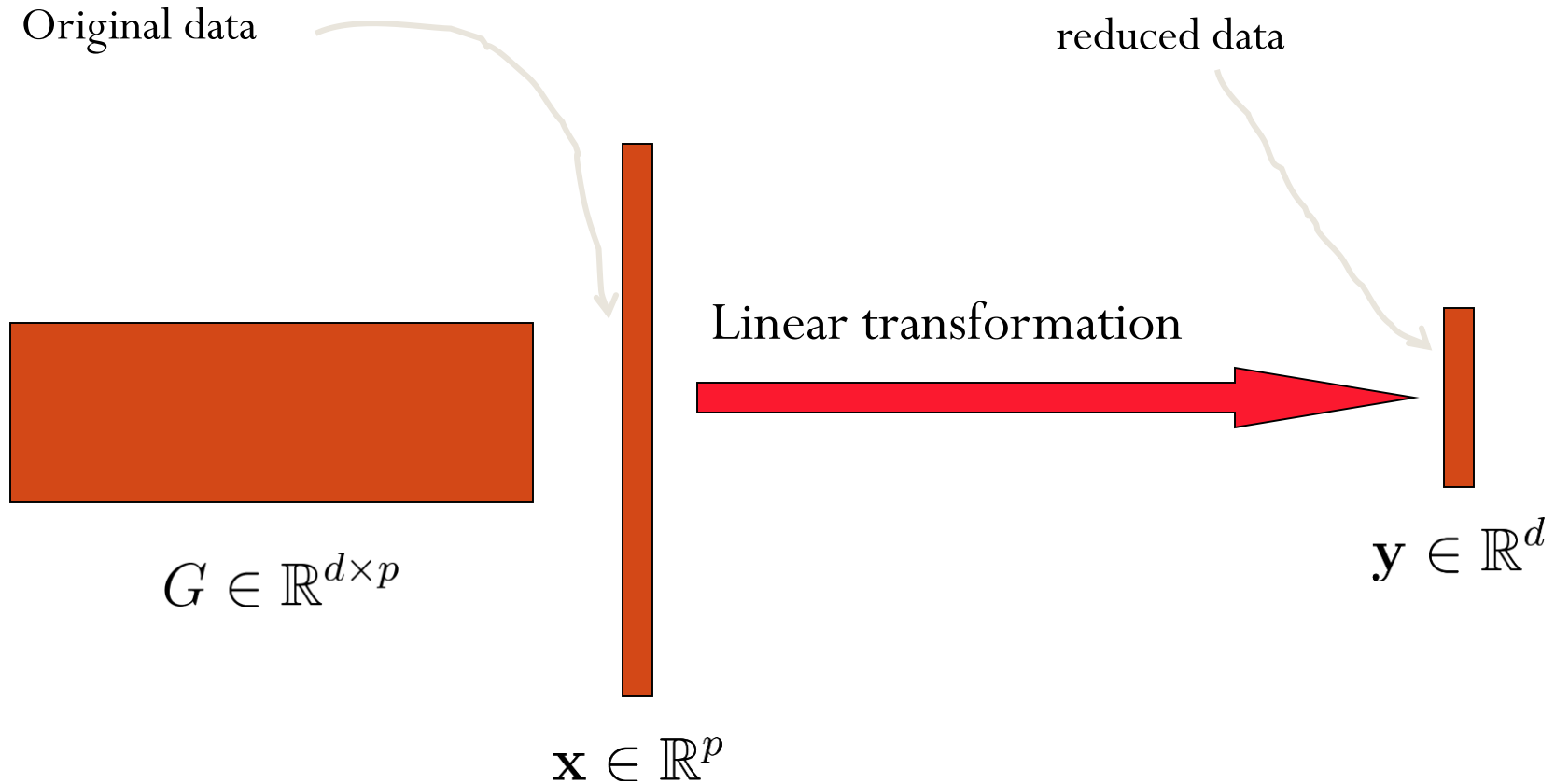
- ◆ What is dimension reduction?
- ◆ Why dimension reduction?
- ◆ Dimension reduction algorithms
- ◆ Principal Component Analysis (PCA)
- ◆ Local linear embedding
- ◆ Feature selection

# What is dimension reduction?

- ◆ Dimension reduction refers to the mapping of the original high-dim data onto a lower-dim space
  - Criterion for dimension reduction can be different based on different problem settings
    - Unsupervised setting: minimize the information loss
    - Supervised setting: maximize the class discrimination
- ◆ Given a set of data points of  $p$  variables  
Compute the linear transformation (projection)

$$G \in \mathbb{R}^{d \times p} : \mathbf{x} \rightarrow \mathbf{y} = G\mathbf{x} \in \mathbb{R}^d$$

# What is dimension reduction? – linear case



$$G \in \mathbb{R}^{d \times p} : \mathbf{x} \rightarrow \mathbf{y} = G\mathbf{x} \in \mathbb{R}^d$$

# Why dimension reduction?

- ◆ Most machine learning and data mining techniques may not be effective for high-dimensional data
  - **Curse of Dimensionality**
  - Query accuracy and efficiency degrade rapidly as the dimension increases.
- ◆ The **intrinsic** dimension may be small.
  - For example, the number of genes responsible for a certain type of disease may be small.

# Why dimension reduction?

- ◆ **Visualization**: projection of high-dimensional data onto 2D or 3D.
- ◆ **Data compression**: efficient storage and retrieval.
- ◆ **Noise removal**: positive effect on query accuracy.

# Example: a job satisfaction questionnaire

## ◆ A questionnaire with 7 items

Please respond to each of the following statements by placing a rating in the space to the left of the statement. In making your ratings, use any number from 1 to 7 in which 1="strongly disagree" and 7="strongly agree."

- \_\_\_\_\_ 1. My supervisor treats me with consideration.
- \_\_\_\_\_ 2. My supervisor consults me concerning important decisions that affect my work.
- \_\_\_\_\_ 3. My supervisors give me recognition when I do a good job.
- \_\_\_\_\_ 4. My supervisor gives me the support I need to do my job well.
- \_\_\_\_\_ 5. My pay is fair.
- \_\_\_\_\_ 6. My pay is appropriate, given the amount of responsibility that comes with my job.
- \_\_\_\_\_ 7. My pay is comparable to the pay earned by other employees whose jobs are similar to mine.

# Example: a job satisfaction questionnaire

- ◆ A questionnaire with 7 items, each item corresponds to a variable
- ◆  $N = 200$  (participants)

Variable	Correlations						
	1	2	3	4	5	6	7
1	1.00						
2	.75	1.00					
3	.83	.82	1.00				
4	.68	.92	.88	1.00			
5	.03	.01	.04	.01	1.00		
6	.05	.02	.05	.07	.89	1.00	
7	.02	.06	.00	.03	.91	.76	1.00

**Strong  
correlation  
means high  
redundancy**

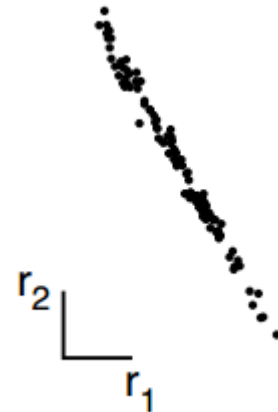
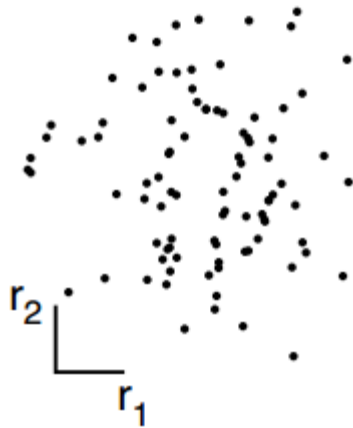
satisfaction with supervision

satisfaction with pay



# Redundant?

◆ which one is redundant?



- highly redundant data are likely to be compressible -- essential idea for dimension reduction

# More examples

## ◆ Face recognition:

- **Representation**: a high-dimensional vector (e.g.,  $20 \times 28 = 560$ ) where each dimension represents the brightness of one pixel



- **Underlying structure parameters**: different camera angles, pose and lighting condition, face expression, etc.

# More examples

## ◆ Character recognition:

- **Representation**: a high-dimensional vector (e.g.,  $28 \times 28 = 784$ ) where each dimension represents the brightness of one pixel



- **Underlying structure parameters**: orientation, curvature, style (e.g., 2 with/without loops)

# More examples

## ◆ Text document analysis:

- **Representation**: a high-dimensional vector (e.g., 10K) of term frequency over the vocabulary of the word



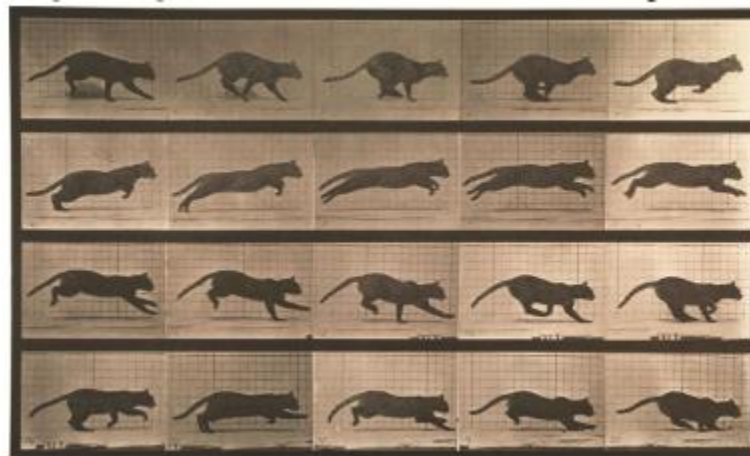
Term	D1	D2
game	1	0
decision	0	0
theory	2	0
probability	0	3
analysis	0	2
...		

- **Underlying structure parameters**: topic proportions, clustering structure

# More examples

## ◆ Motion capture:

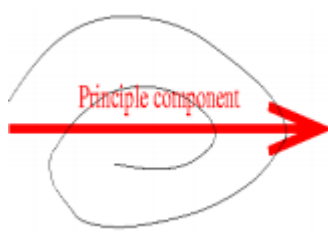
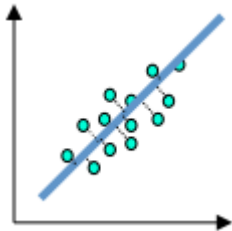
- **Representation:** pose is determined, e.g., by the 3D coordinates of multiple points on the body



- **Underlying structure parameters:** pose type
- Motion can be viewed as a trajectory on the manifold

# Dimension reduction algorithms

◆ Many methods have been developed



	Unsupervised	Supervised
Linear	PCA, ICA, SVD, LSA (LSI)	LDA, CCA, PLS
Non-linear	Isomap, LLE, MDR	Learning with Non-linear kernels

◆ We will cover PCA and LLE as examples

# PCA: Principal Component Analysis

- ◆ probably the most widely-used and well-known of the “standard” multivariate methods
- ◆ invented by Karl Pearson (1901) and independently developed by Harold Hotelling (1933)
- ◆ first applied in ecology by Goodall (1954) under the name “factor analysis” (“principal factor analysis” is a synonym of PCA).

# Review: Eigenvector, Eigenvalue

◆ For a square matrix  $A$  ( $p \times p$ ), the eigenvector is defined as

$$A\boldsymbol{\mu} = \lambda\boldsymbol{\mu}$$

□ where  $u$  is an eigenvector and  $\lambda$  is the corresponding eigenvalue

◆ Put in a matrix form

$$AU = U\Lambda$$

$$U = [\boldsymbol{\mu}_1, \cdots, \boldsymbol{\mu}_p] \quad \Lambda = \text{diag}(\lambda_1, \cdots, \lambda_p)$$

◆ For symmetric matrices, the eigenvectors can be orthogonal

$$UU^\top = U^\top U = I$$

□ Thus:

$$U^\top AU = \Lambda \quad A = U\Lambda U^\top$$



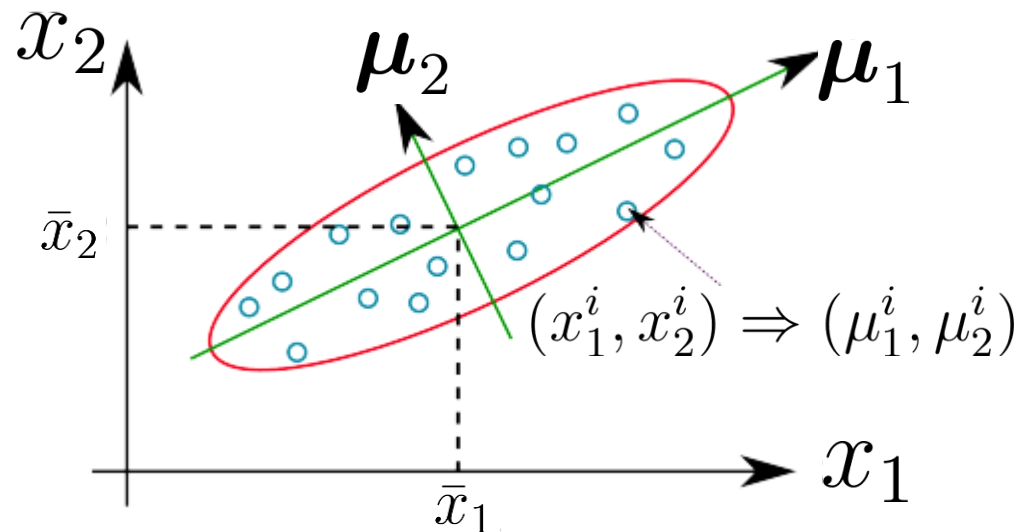
# PCA for dimension reduction

An eigen-decomposition process to data covariance matrix

- ◆ Minus the empirical mean to get centered data
- ◆ Compute the covariance
$$S = \frac{1}{N} \sum_n \mathbf{x}_n \mathbf{x}_n^\top$$
- ◆ Doing eigenvalue decomposition
  - Let  $U$  be the eigenvectors of  $S$  corresponding to the top  $d$  eigenvalues
- ◆ Encode data  $Y = U^\top X$
- ◆ Reconstruct data  $\hat{X} = UY = UU^\top X$

# Apply to data covariance -- eigensystem

- ◆ The eigenvectors of the covariance  $\Sigma$  define a new axis system

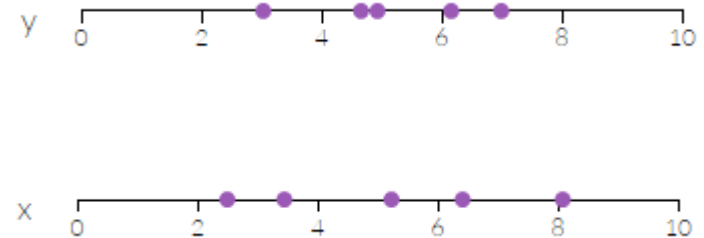
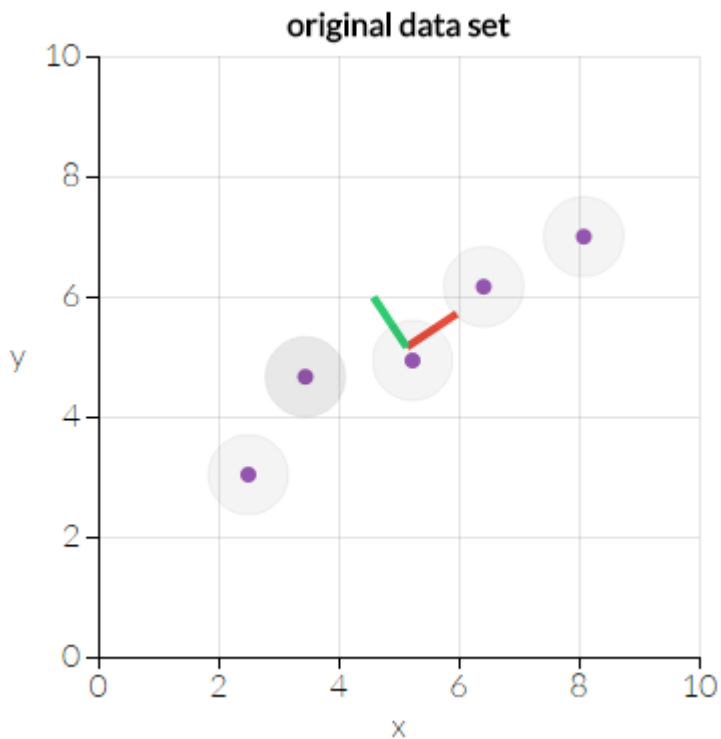


- Any point  $p_x$  in the X-axis system,  $\bar{\mathbf{x}}$  is the data mean, the coordinate in the U-axis system is:

$$p_\mu = U^\top (p_x - \bar{\mathbf{x}})$$

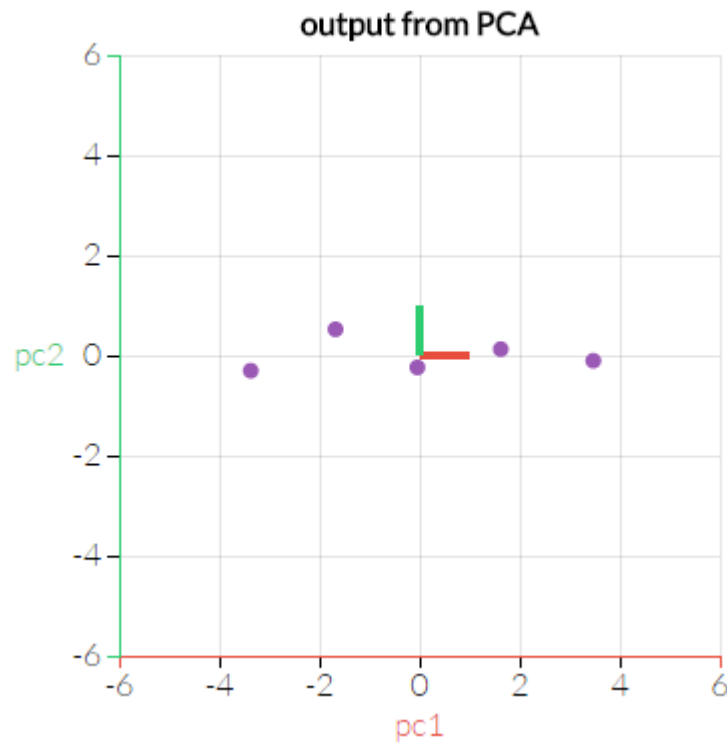
# A 2D Example

◆ 2D data represented in 1D dimensions



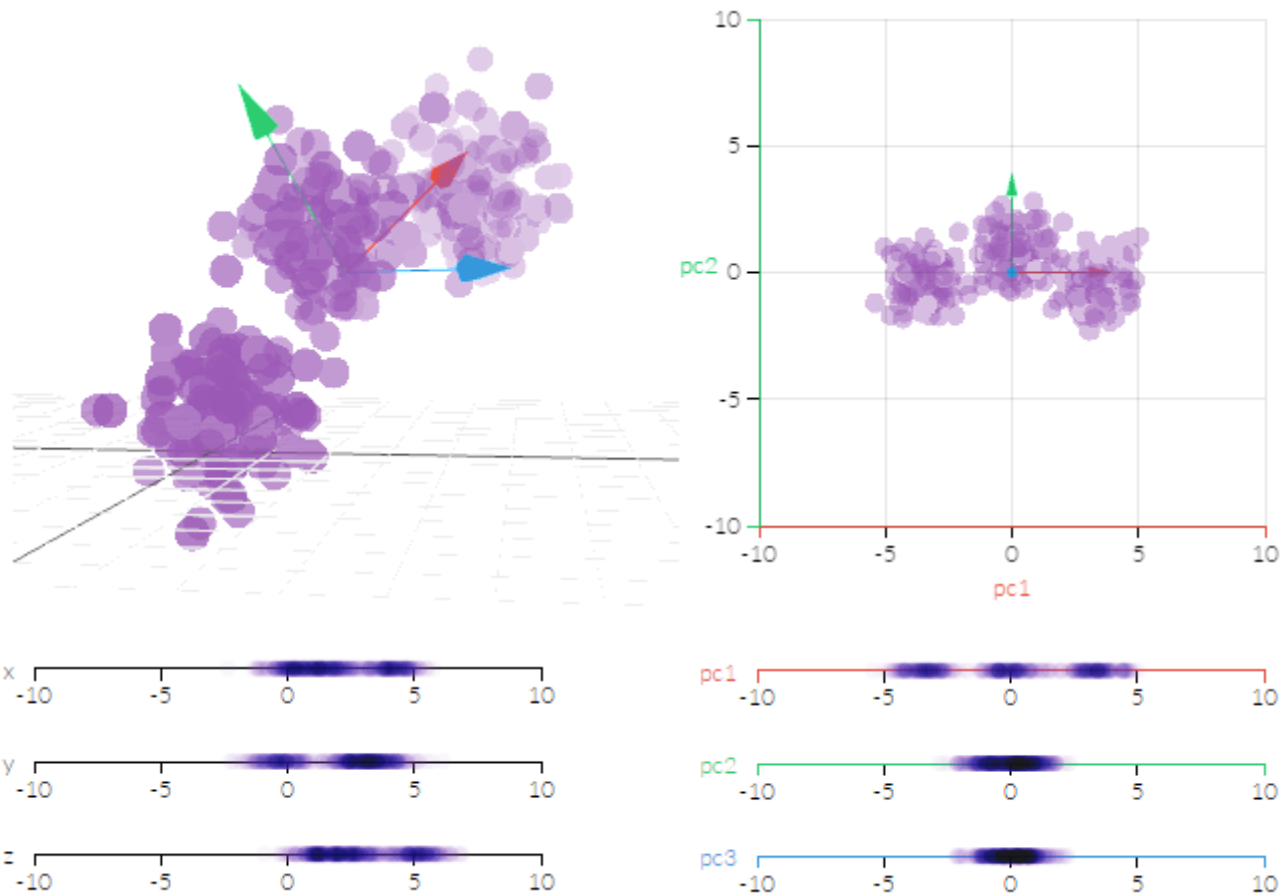
# A 2D Example

◆ 2D data represented in 1D dimensions



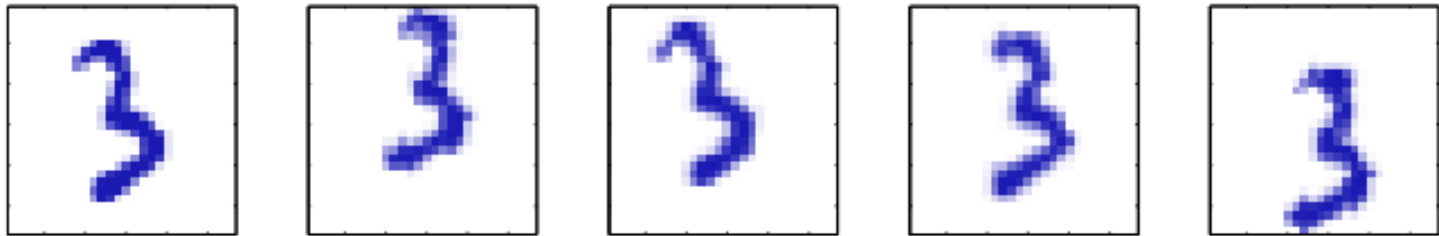
# A 3D Example

◆ 3D data represented in 2D dimensions

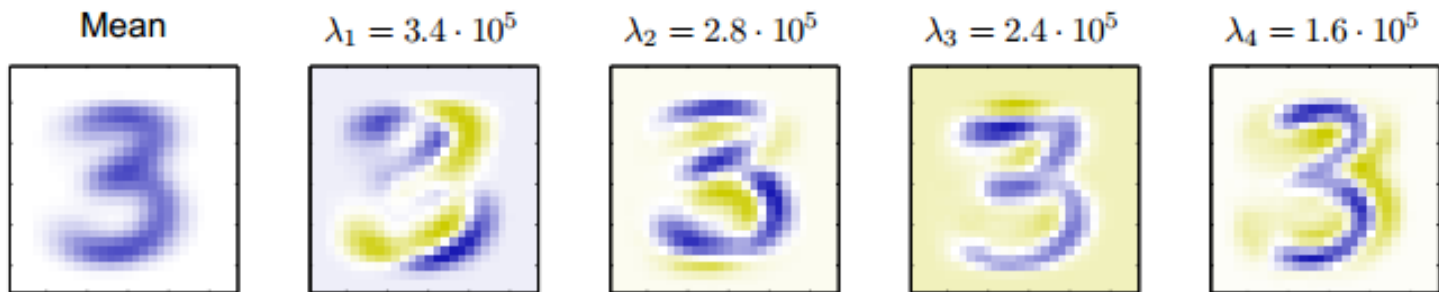


[<http://setosa.io/ev/principal-component-analysis/>]

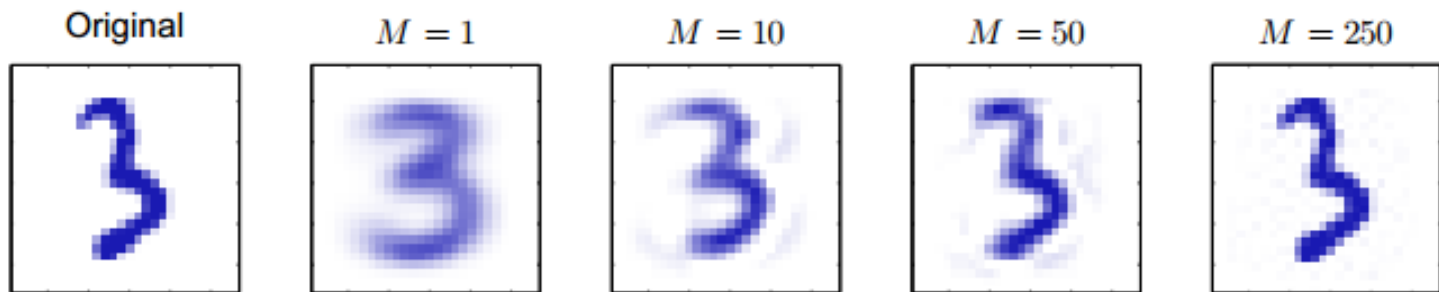
# A high-dimensional Example



(a) Samples of digit '3'

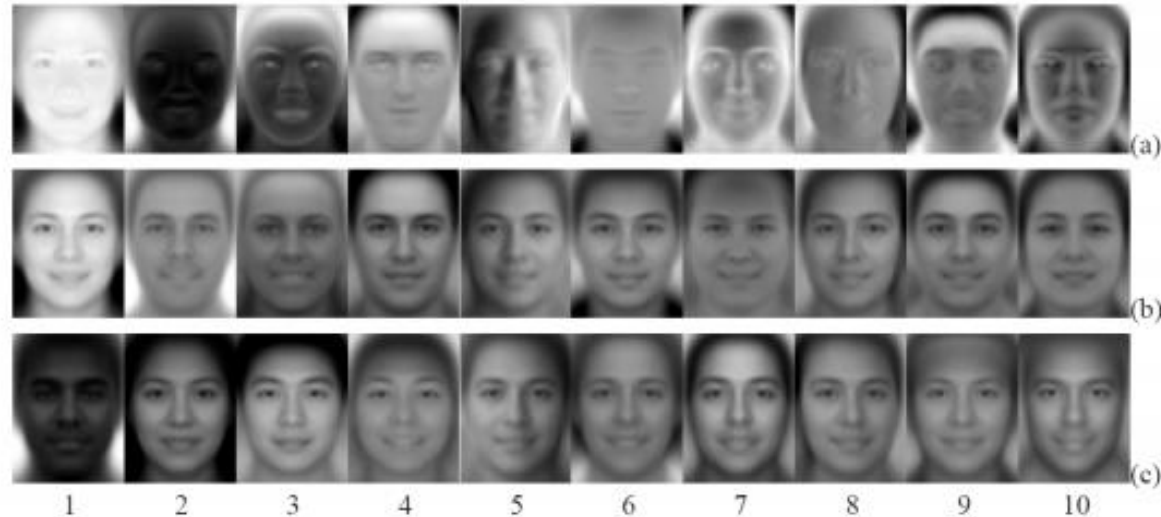


(b) Eigenvectors and corresponding eigenvalues

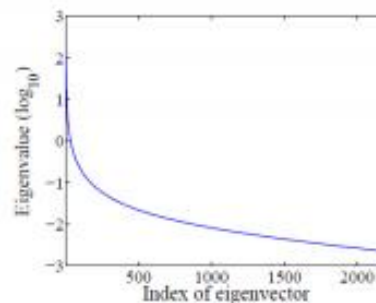


(c) PCA Reconstruction

# Eigenfaces



(d)



(e)

(a) Top 10 eigenvectors corresponding to the 10 largest eigenvalues. (b) Eigenvectors (eigenfaces) are multiplied by  $3\sigma$  where  $\sigma$  is the square root of eigenvalue and added to the mean face. (c) Eigenvectors are multiplied by  $3\sigma$  and added to the mean face. (d) Mean face. (e) The logarithm of eigenvalues.

# How to choose $d$ ?

◆ Measure the total variance accounted for by the  $d$  principal components

- the percentage of the variance accounted for by the  $i$ -th eigenvector:

$$r_i = \frac{\lambda_i}{\sum_{j=1}^p \lambda_j} \times 100$$

- Account for a minimum percentage of total variance, e.g., 95%:

$$\sum_{i=1}^d r_i \geq 95$$



# Theory of PCA

◆ There are three types of interpretation

- Maximum variance
- Least reconstruct error
- Probabilistic model

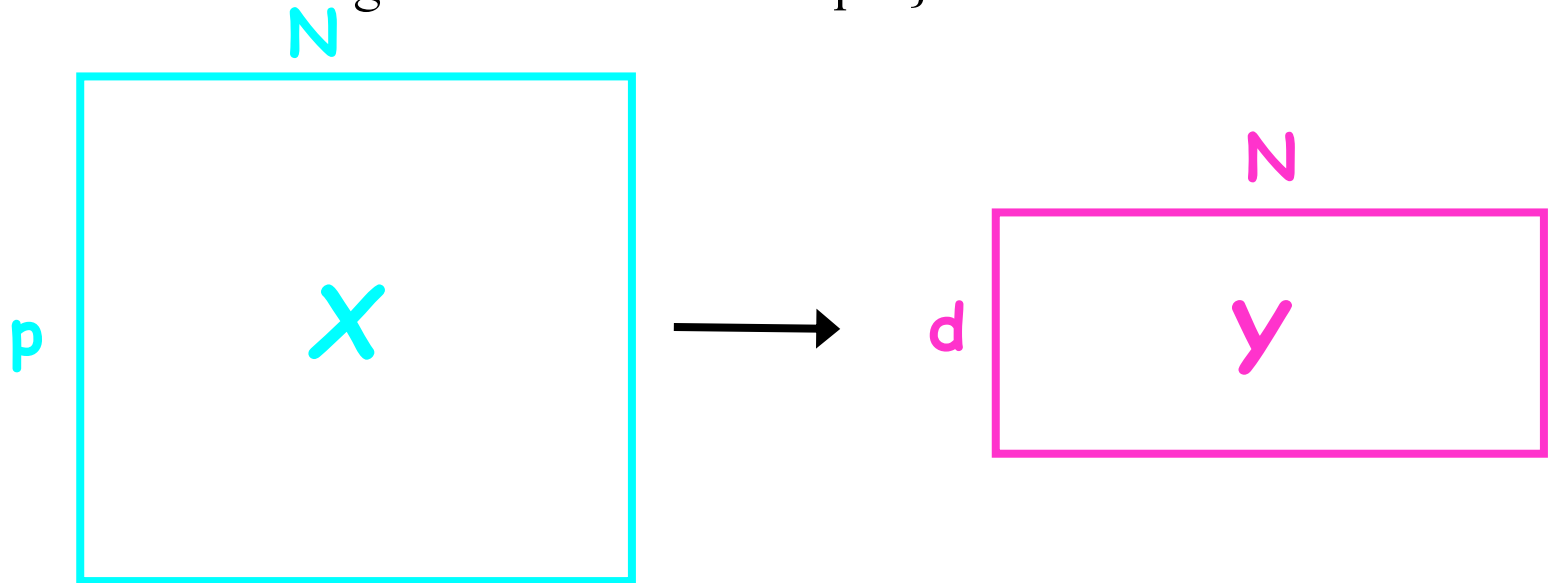
# Maximum Variance Formulation

◆ Given a set of data points  $\{\mathbf{x}_n\}$ ,  $n = 1, \dots, N$

$$\mathbf{x}_n \in \mathbb{R}^p$$

◆ **Goal:**

- Project the data into an  $d$ -dimensional ( $d < p$ ) space while maximizing the variance of the projected data



# Maximum Variance Formulation

- ◆ Let's start with the 1-dimensional projection, i.e.,  $d = 1$
- ◆ We only care about the projection direction, not the scale, so we assume

$$\mu_1^\top \mu_1 = 1$$

- ◆ The projection is

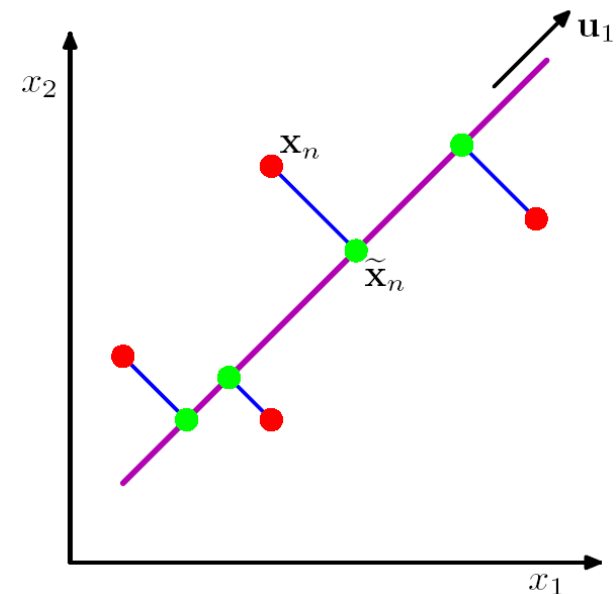
$$y_n = \mu_1^\top \mathbf{x}_n$$

- ◆ Mean and variance of projected data:

$$\bar{y} = \mu_1^\top \bar{\mathbf{x}}, \quad \text{where } \bar{\mathbf{x}} = \frac{1}{N} \sum_n \mathbf{x}_n$$

$$\text{var}(y) = \frac{1}{N} \sum_n (\mu_1^\top \mathbf{x}_n - \mu_1^\top \bar{\mathbf{x}})^2 = \mu_1^\top S \mu_1$$

□ sample covariance 
$$S = \frac{1}{N} \sum_n (\mathbf{x}_n - \bar{\mathbf{x}})(\mathbf{x}_n - \bar{\mathbf{x}})^\top$$



# Maximum Variance Formulation

◆ Now, we get a **constrained optimization** problem

$$\max_{\mu_1} \text{var}(y) = \frac{1}{N} \sum_n (\mu_1^\top \mathbf{x}_n - \mu_1^\top \bar{\mathbf{x}})^2 = \mu_1^\top S \mu_1$$

□ where  $\mu_1^\top \mu_1 = 1$

◆ Solve it using **Lagrangian methods**, we get

□ The **eigenvector** problem

$$S \mu_1 = \lambda_1 \mu_1$$

□ The lagrange multiplier is the **eigenvalue**

$$\mu_1^\top S \mu_1 = \lambda_1$$

□ The eigenvector corresponds to largest eigenvalue is **1<sup>st</sup> PC**.

$$\bar{\mathbf{x}} = \frac{1}{N} \sum_n \mathbf{x}_n \quad S = \frac{1}{N} \sum_n (\mathbf{x}_n - \bar{\mathbf{x}})(\mathbf{x}_n - \bar{\mathbf{x}})^\top$$

# Maximum Variance Formulation

- ◆ Additional components can be incrementally found

$$\max_{\mu_2} \text{var}(y) = \frac{1}{N} \sum_n (\mu_2^\top \mathbf{x}_n - \mu_2^\top \bar{\mathbf{x}})^2 = \mu_2^\top S \mu_2$$

- where  $\mu_2^\top \mu_2 = 1$  and  $\mu_1^\top \mu_2 = 0$

- ◆ Solve this problem with Lagrangian method, we have

$$2S\mu_2 - 2\lambda_2\mu_2 - \gamma\mu_1 = 0$$

- which leads to

$$S\mu_2 - \lambda_2\mu_2 - \gamma\mu_1 = 0$$

- Left multiplying  $\mu_1^\top$ , we get (remember  $\mu_1$  is eigenvector)

$$\gamma = \mu_1^\top S\mu_2 = \lambda_1 \mu_1^\top \mu_2 = 0$$

- Thus,  $S\mu_2 = \lambda_2\mu_2$        $\mu_2^\top S\mu_2 = \lambda_2$

# Maximum Variance Formulation

- ◆ For the general case of an  $d$  dimensional subspace, it is obtained by the  $d$  eigenvectors  $\mu_1, \mu_2, \dots, \mu_d$  of the data covariance matrix  $S$  corresponding to the  $d$  largest eigenvalues  $\lambda_1, \lambda_2, \dots, \lambda_d$

# Minimum Error Formulation

- ◆ A set of **complete orthonormal** basis

$$\{\mu_i\}, \quad i = 1, \dots, p$$

$$\mu_i^\top \mu_j = \delta_{ij}$$

- ◆ Each data point can be represented as

$$\mathbf{x}_n = \sum_i \alpha_{ni} \mu_i$$

- Due to the **orthonormal property**, we can get

$$\alpha_{ni} = \mathbf{x}_n^\top \mu_i$$

$$\mathbf{x}_n = \sum_i (\mathbf{x}_n^\top \mu_i) \mu_i$$

# Minimum Error Formulation

- ◆ A set of complete orthonormal basis

$$\{\mu_i\}, \quad i = 1, \dots, p$$

$$\mu_i^\top \mu_j = \delta_{ij}$$

- ◆ We consider a low-dimensional approximation

$$\tilde{\mathbf{x}}_n = \sum_{i=1}^d z_{ni} \mu_i + \sum_{i=d+1}^p b_i \mu_i$$

- where  $b_i$  are constants for all data points

- ◆ The best approximation is to minimize the error

$$\min_{U, \mathbf{z}, \mathbf{b}} J := \frac{1}{N} \sum_{n=1}^N \|\mathbf{x}_n - \tilde{\mathbf{x}}_n\|^2$$



# Minimum Error Formulation

- ◆ A set of complete orthonormal basis

$$\{\mu_i\}, \quad i = 1, \dots, p \quad \mu_i^\top \mu_j = \delta_{ij}$$

- ◆ The best approximation is to minimize the error

$$J = \frac{1}{N} \sum_{n=1}^N \|\mathbf{x}_n - \tilde{\mathbf{x}}_n\|^2 \quad \tilde{\mathbf{x}}_n = \sum_{i=1}^d z_{ni} \mu_i + \sum_{i=d+1}^p b_i \mu_i$$

- we get (*proof?*)

$$z_{ni} = \mathbf{x}_n^\top \mu_i, \quad i = 1, \dots, d \quad b_i = \bar{\mathbf{x}}^\top \mu_i, \quad i = d+1, \dots, p$$

- Use the general representation  $\mathbf{x}_n = \sum_i (\mathbf{x}_n^\top \mu_i) \mu_i$ , we get that the displacement lines in the orthogonal subspace

$$\mathbf{x}_n - \tilde{\mathbf{x}}_n = \sum_{i=d+1}^p \{(\mathbf{x}_n - \bar{\mathbf{x}})^\top \mu_i\} \mu_i$$

# Minimum Error Formulation

◆ With the result

$$\mathbf{x}_n - \tilde{\mathbf{x}}_n = \sum_{i=d+1}^p \{(\mathbf{x}_n - \bar{\mathbf{x}})^\top \mu_i\} \mu_i$$

◆ We get the error

$$\begin{aligned} J &= \frac{1}{N} \sum_{n=1}^N \|\mathbf{x}_n - \tilde{\mathbf{x}}_n\|^2 = \frac{1}{N} \sum_n \sum_{i=d+1}^p (\mathbf{x}_n^\top \mu_i - \bar{\mathbf{x}}^\top \mu_i)^2 \\ &= \sum_{i=d+1}^p \mu_i^\top S \mu_i \end{aligned}$$

◆ The optimization problem

$$\min_{\mu_i} J$$

□ where  $\mu_i^\top \mu_i = 1$

# Minimum Error Formulation

- ◆ Consider a 2-dimensional space ( $p=2$ ) and a 1-dimensional principal subspace ( $d=1$ ). Then, we choose  $\mu_2$  that minimizes

$$\min_{\mu_2} J = \mu_2^\top S \mu_2$$

$$\text{s.t.: } \mu_2^\top \mu_2 = 1$$

- We have:

$$S \mu_2 = \lambda_2 \mu_2$$

- ◆ We therefore obtain the minimum value of  $J$  by choosing  $\mu_2$  as the eigenvector corresponding to the smaller eigenvalue
- ◆ We choose the principal subspace by the eigenvector with the large eigenvalue

# Minimum Error Formulation

- ◆ The general solution is to choose the eigenvectors of the covariance matrix with  $d$  largest eigenvalues

$$S\mu_i = \lambda_i\mu_i$$

□ where  $i = 1, \dots, d$

- ◆ The distortion measure (i.e., reconstruction error) becomes

$$J = \sum_{i=d+1}^p \lambda_i$$

# PCA Reconstruction

- ◆ By the minimum error formulation, the PCA approximation can be written as:

$$\tilde{\mathbf{x}}_n = \sum_{i=1}^d z_{ni} \mu_i + \sum_{i=d+1}^p b_i \mu_i$$

$$z_{ni} = \mathbf{x}_n^\top \mu_i, \quad i = 1, \dots, d \quad b_i = \bar{\mathbf{x}}^\top \mu_i, \quad i = d+1, \dots, p$$

- ◆ We have

$$\begin{aligned} \tilde{\mathbf{x}}_n &= \sum_{i=1}^d (\mathbf{x}_n^\top \mu_i) \mu_i + \sum_{i=d+1}^p (\bar{\mathbf{x}}^\top \mu_i) \mu_i \\ &= \sum_{i=1}^d (\mathbf{x}_n^\top \mu_i + \bar{\mathbf{x}}^\top \mu_i - \bar{\mathbf{x}}^\top \mu_i) \mu_i + \sum_{i=d+1}^p (\bar{\mathbf{x}}^\top \mu_i) \mu_i \\ &= \bar{\mathbf{x}} + \sum_{i=1}^d (\mathbf{x}_n^\top \mu_i - \bar{\mathbf{x}}^\top \mu_i) \mu_i \end{aligned}$$

- ◆ Essentially, this representation implies compression of p-dim data into a d-dim vector with components  $(\mathbf{x}_n^\top \mu_i - \bar{\mathbf{x}}^\top \mu_i)$

# Probabilistic PCA

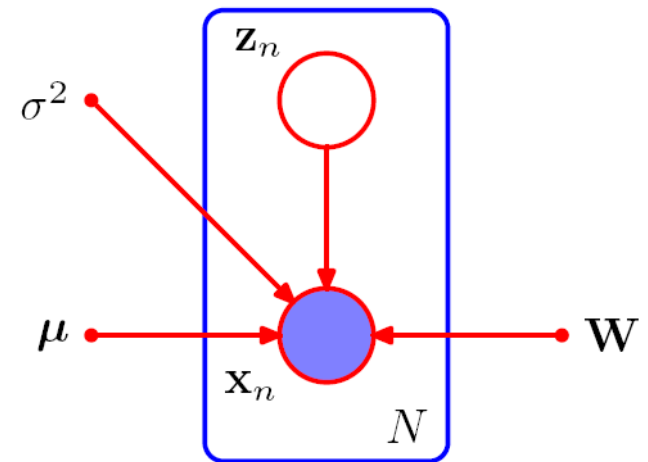
- ◆ A simple **linear-Gaussian** model
- ◆ Let  $\mathbf{z}$  be a latent feature vector  $\mathbf{z} \in \mathbb{R}^d$ 
  - In Bayesian, we assume it's prior  $\mathbf{z} \sim \mathcal{N}(0, I)$
- ◆ A linear-Gaussian model

$$\mathbf{x} = W\mathbf{z} + \mu + \epsilon \quad \epsilon \sim \mathcal{N}(0, \sigma^2 I)$$

- this gives the likelihood

$$p(\mathbf{x}|\mathbf{z}) = \mathcal{N}(\mathbf{x}|W\mathbf{z} + \mu, \sigma^2 I)$$

- the columns of  $W$  span a linear subspace

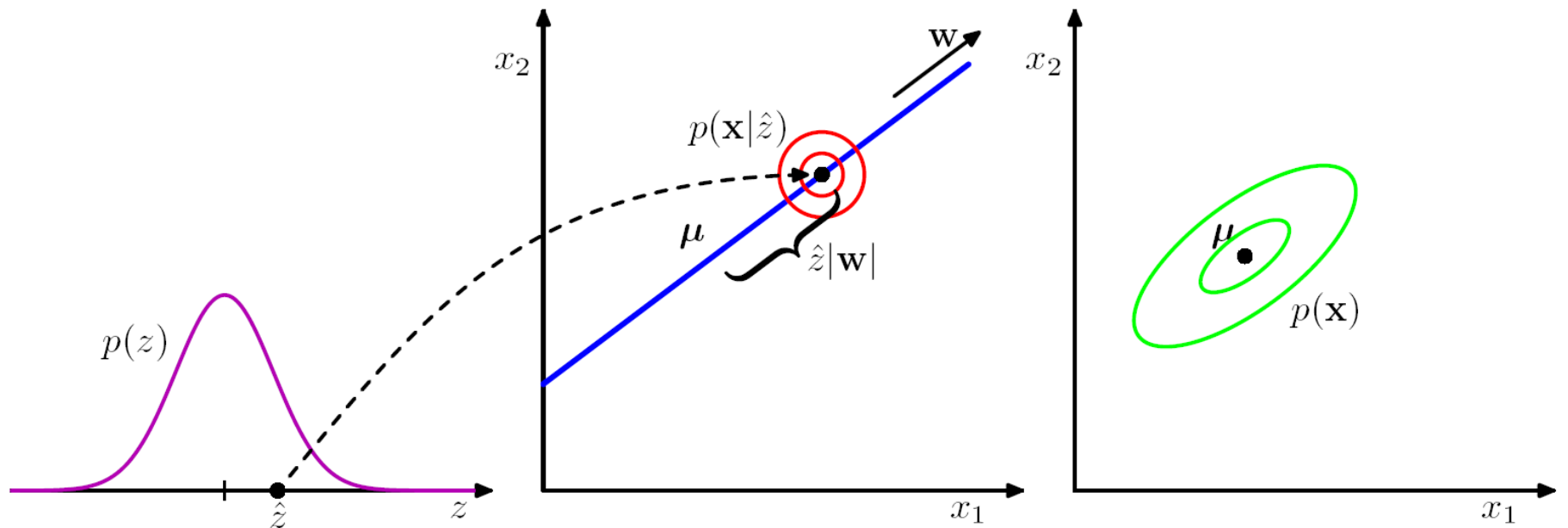


# Probabilistic PCA

◆ By the properties of Gaussian, we can get the marginal

$$p(\mathbf{x}) = \int p(\mathbf{z})p(\mathbf{x}|\mathbf{z})d\mathbf{z} = \mathcal{N}(\mathbf{x}|\mu, C)$$

$$C = WW^\top + \sigma^2 I$$



# Unidentifiability issue

- ◆ Any rotation of the latent dimensions leads to invariance of the predictive distribution

$$p(\mathbf{x}) = \int p(\mathbf{z})p(\mathbf{x}|\mathbf{z})d\mathbf{z} = \mathcal{N}(\mathbf{x}|\mu, C)$$

$$C = WW^\top + \sigma^2 I$$

- Let  $R$  be an orthogonal matrix with  $RR^\top = I$
- Define  $\widetilde{W} = WR$
- Then, we have

$$\widetilde{W}\widetilde{W}^\top = WRR^\top W^\top = WW^\top$$

- which is independent of  $R$



# Inverse of the Covariance matrix

- ◆ Evaluating the inverse of the covariance matrix  $C$  has complexity  $O(p^3)$ . We can do inversion as follows

$$C^{-1} = \sigma^{-2}I - \sigma^{-2}WM^{-1}W^{\top}$$

- where the  $d \times d$  matrix  $M$  is:

$$M = W^{\top}W + \sigma^2I$$

- ◆ Evaluating the inverse of  $M$  has complexity  $O(d^3)$

# Probabilistic PCA

◆ By the properties of Gaussian, we can get the posterior

$$p(\mathbf{z}|\mathbf{x}) = \mathcal{N}(\mathbf{z}|M^{-1}W^{\top}(\mathbf{x} - \mu), \sigma^{-2}M)$$

$$M = W^{\top}W + \sigma^2I$$

- The posterior mean **depends** on  $\mathbf{x}$  (**a linear projection of  $\mathbf{x}$** )

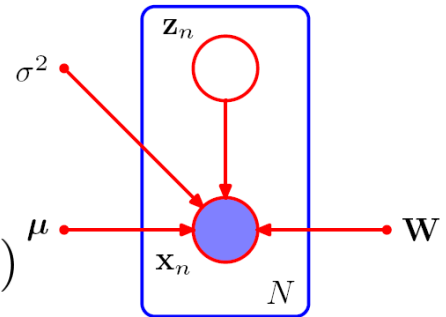
$$\mathbb{E}[\mathbf{z}|\mathbf{x}] = M^{-1}W^{\top}(\mathbf{x} - \mu)$$

- Posterior covariance is **independent** of  $\mathbf{x}$

# Maximum Likelihood PCA

◆ Given a set of observations  $X = \{\mathbf{x}_n\}$ , the log-likelihood is

$$\begin{aligned}\log p(X|\mu, W, \sigma^2) &= \sum_n \log p(\mathbf{x}_n|W, \mu, \sigma^2) \\ &= -\frac{Np}{2} \log(2\pi) - \frac{N}{2} \log |C| - \frac{1}{2} \sum_n (\mathbf{x}_n - \mu)^\top C^{-1} (\mathbf{x}_n - \mu)\end{aligned}$$



◆ We get the MLE:  $\hat{\mu} = \bar{\mathbf{x}}$  and

$$\log p(X|\mu, W, \sigma^2) = -\frac{N}{2} \left( p \log(2\pi) + \log |C| + \text{Tr}(C^{-1}S) \right)$$

# Maximum Likelihood PCA

## ◆ Log-likelihood

$$\log p(X|\mu, W, \sigma^2) = -\frac{N}{2} \left( p \log(2\pi) + \log |C| + \text{Tr}(C^{-1}S) \right)$$

- The **stationary points** can be written as (Tipping & Bishop, 1999)

$$\hat{W} = U_d(L_d - \sigma^2 I)^{1/2} R$$

- $L_d$  is diagonal with eigenvalues  $\lambda_i$ ;  $R$  is an arbitrary  $d \times d$  orthogonal matrix;  $U_d$  is  $p \times d$  matrix whose columns are eigenvectors of  $S$
- The maximum of likelihood is obtained while the  $d$  eigenvectors are chosen to be those whose eigenvalues are the  $d$  largest

- MLE for  $\sigma^2$  is:

$$\hat{\sigma}^2 = \frac{1}{p-d} \sum_{i=d+1}^p \lambda_i$$

- The average variance associated with the discarded dimensions

# Maximum Likelihood PCA

◆ Since the choice of  $R$  doesn't affect the covariance matrix, we can simply choose  $R = I$

◆ Recover the conventional PCA

□ Take the limit  $\sigma^2 \rightarrow 0$ , we get the posterior mean

$$\mathbb{E}[\mathbf{z}|\mathbf{x}] = M^{-1}W^\top(\mathbf{x} - \mu) = (\hat{W}^\top \hat{W})^{-1}\hat{W}^\top(\mathbf{x} - \bar{\mathbf{x}})$$

□ which is an **orthogonal projection** of the data point into the latent space

□ So we **recover the standard PCA**

# EM Algorithm for PPCA

◆ **E-step:** evaluate expectation of complete likelihood

$$\begin{aligned}\mathbb{E}[\log p(X, Z|\Theta)] = & - \sum_n \left\{ \frac{p}{2} \log(2\pi\sigma^2) + \frac{1}{2} \text{Tr}(\mathbb{E}[\mathbf{z}_n \mathbf{z}_n^\top]) \right. \\ & \left. + \frac{1}{2\sigma^2} \|\mathbf{x}_n - \mu\|^2 - \frac{1}{\sigma^2} \mathbb{E}[\mathbf{z}_n]^\top W^\top (\mathbf{x}_n - \mu) + \frac{1}{2\sigma^2} \text{Tr}(\mathbb{E}[\mathbf{x}_n \mathbf{z}_n^\top] W^\top W) \right\}\end{aligned}$$

□ where

$$\mathbb{E}[\mathbf{z}_n] = M^{-1} W^\top (\mathbf{x}_n - \bar{\mathbf{x}})$$

$$\mathbb{E}[\mathbf{z}_n \mathbf{z}_n^\top] = \sigma^2 M^{-1} + \mathbb{E}[\mathbf{z}_n] \mathbb{E}[\mathbf{z}_n]^\top$$

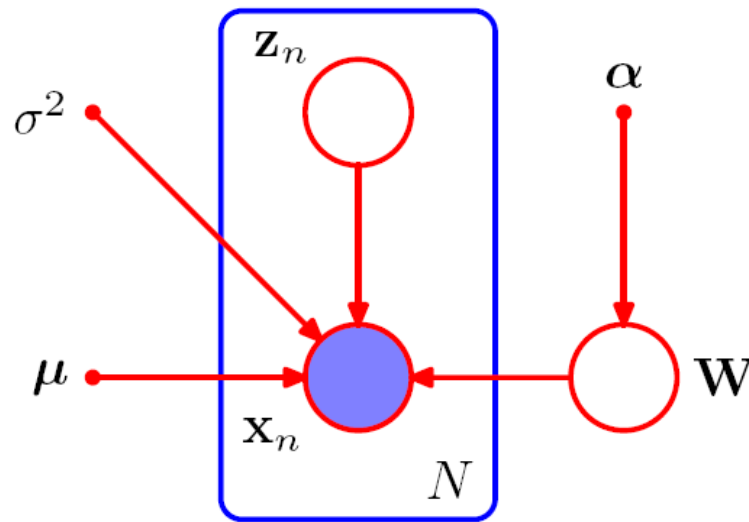
◆ **M-step:** optimizes over parameters

$$W = \left[ \sum_n (\mathbf{x}_n - \bar{\mathbf{x}}) \mathbb{E}[\mathbf{z}_n] \right]^\top \left[ \sum_n \mathbb{E}[\mathbf{z}_n \mathbf{z}_n^\top] \right]^{-1}$$

$$\sigma^2 = \frac{1}{Np} \sum_n \left\{ \|\mathbf{x}_n - \bar{\mathbf{x}}\|^2 - 2 \mathbb{E}[\mathbf{z}_n]^\top W^\top (\mathbf{x}_n - \bar{\mathbf{x}}) + \text{Tr}(\mathbb{E}[\mathbf{z}_n \mathbf{z}_n^\top] W^\top W) \right\}$$

# Bayesian PCA

- ◆ A prior is assumed on the parameters  $W$



$$p(W|\alpha) = \prod_{i=1}^d \left( \frac{\alpha_i}{2} \right)^{p/2} \exp \left\{ -\frac{1}{2} \alpha_i \mathbf{w}_i^\top \mathbf{w}_i \right\}$$

- ◆ Inference can be done in closed-form, as in GP regression
- ◆ Fully Bayesian treatment put priors on  $\mu, \sigma^2, \alpha$

# Factor Analysis

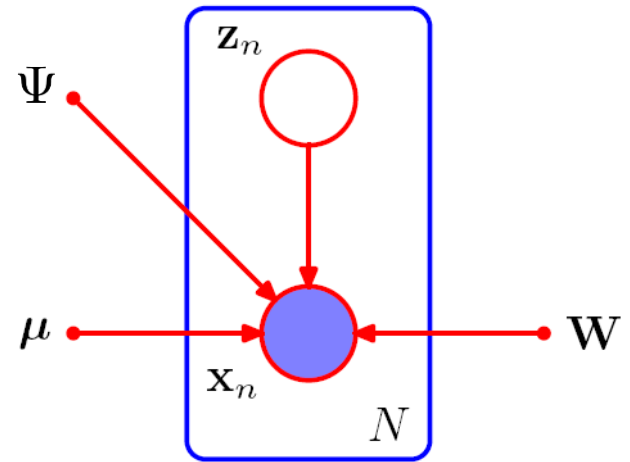
- ◆ Another simple linear-Gaussian model
- ◆ Let  $\mathbf{z}$  be a latent feature vector  $\mathbf{z} \in \mathbb{R}^d$ 
  - In Bayesian, we assume it's prior  $\mathbf{z} \sim \mathcal{N}(0, I)$
- ◆ A linear-Gaussian model

$$\mathbf{x} = W\mathbf{z} + \mu + \epsilon \quad \epsilon \sim \mathcal{N}(0, \Psi)$$

- $\Psi$  is a **diagonal matrix**
- this gives the likelihood

$$p(\mathbf{x}|\mathbf{z}) = \mathcal{N}(\mathbf{x}|W\mathbf{z} + \mu, \Psi)$$

- the columns of  $W$  span a linear subspace





# Factor Analysis

- ◆ We can the inference tasks almost the same as in PCA
- ◆ The predictive distribution is Gaussian
- ◆ EM algorithm can be applied to maximum likelihood estimation

# PCA in high-dimensions

◆ What is  $p$  is very large, e.g.,  $p \gg N$ ?

$$S = \frac{1}{N} X X^\top$$

- which is a  $p \times p$  matrix
- Finding the eigenvectors typically has complexity  $O(p^3)$ 
  - Computationally expensive
- The number of nonzero eigenvalues is no larger than  $N$ 
  - Waste of time to work with  $S$
- How about working with the  $N \times N$  full rank Gram matrix?

$$G = X^\top X$$

# Dual PCA – PCA in high-dimensions

◆ For **centered** data, we have

- $S = \frac{1}{N} X X^\top$  with eigenvalues and eigenvectors  $(\lambda_i, \mu_i)$
- $G = X^\top X$  with eigenvalues and eigenvectors  $(\gamma_i, \nu_i)$

◆ By left-multiplying  $X^\top$  to  $X X^\top \mu_i = N \lambda_i \mu_i$ , we get

$$X^\top X (X^\top \mu_i) = N \lambda_i (X^\top \mu_i) \quad , \quad \nu_i = X^\top \mu_i \quad \text{and} \quad \gamma_i = N \lambda_i$$

◆ Thus,

$$X \nu_i = X X^\top \mu_i = N \lambda_i \mu_i = \gamma_i \mu_i \quad \mu_i = \frac{1}{\gamma_i} X \nu_i$$

# Kernel PCA

- ◆ **PCA is linear**: the reduced dimension representation is generated by linear projections
- ◆ **Kernel PCA is nonlinear** by exploring kernel trick

$$\Phi : \mathcal{X} \rightarrow \mathcal{H}$$

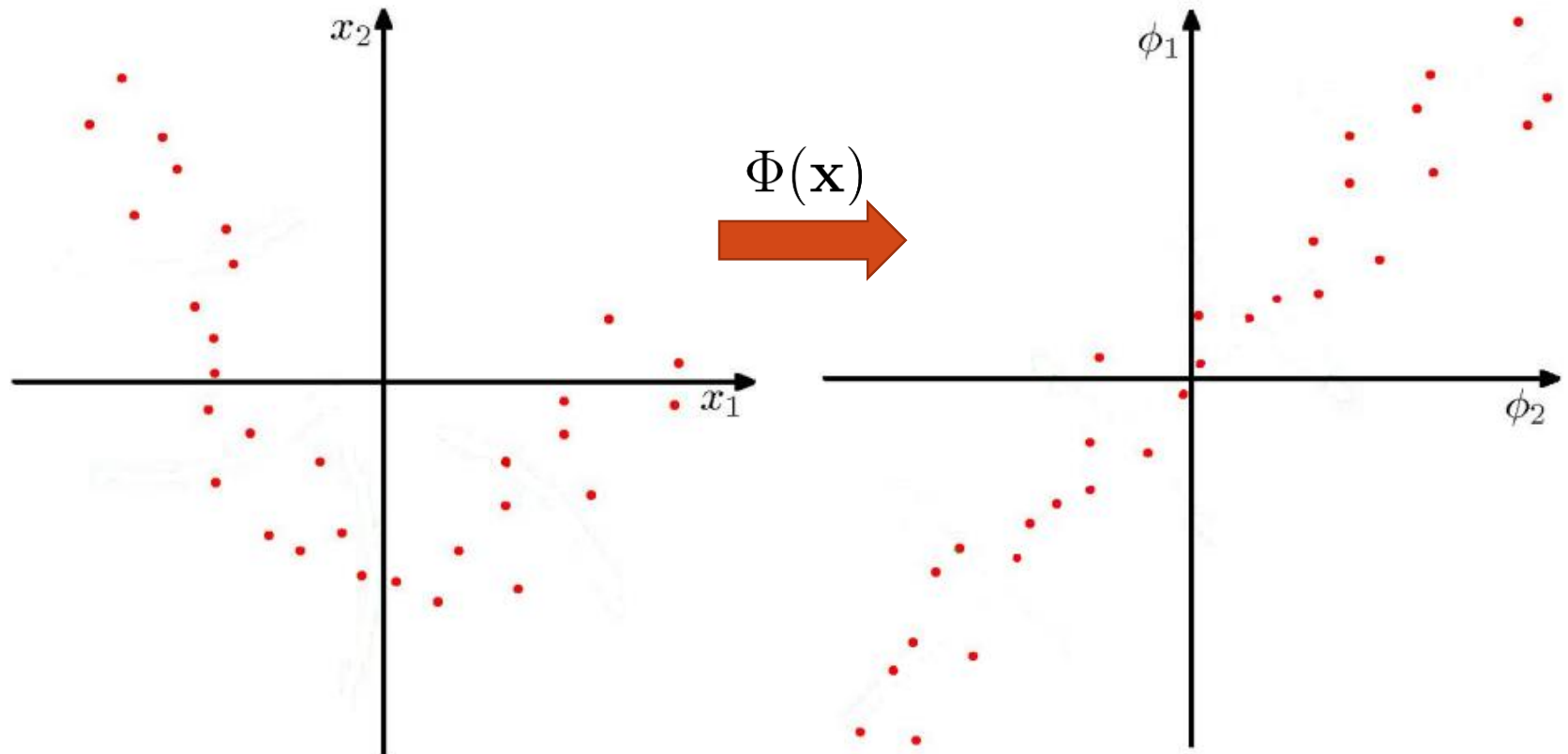
$$\mathbf{x} \mapsto \Phi(\mathbf{x})$$

- ◆ Apply dual PCA in the Hilbert space

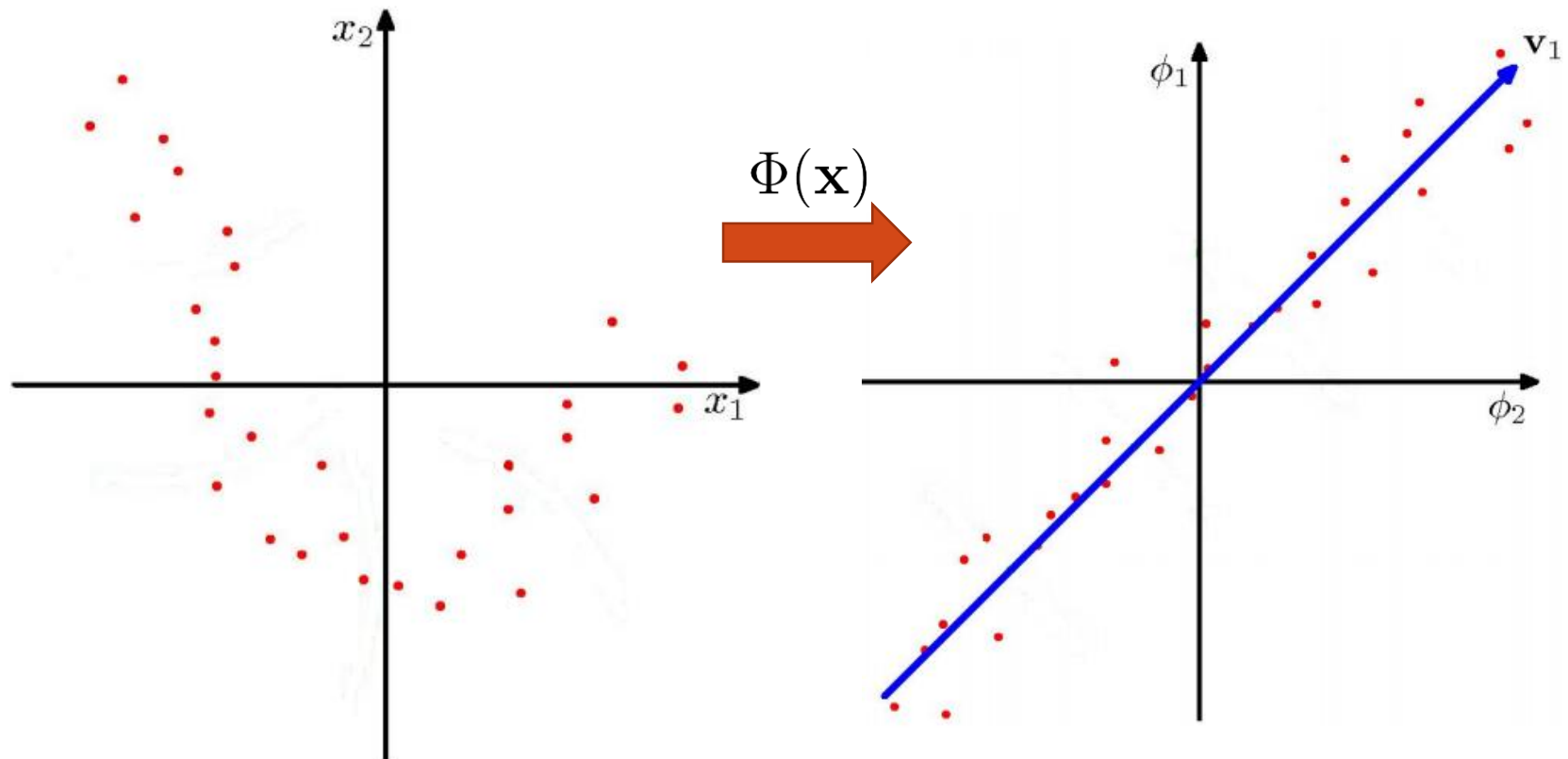
$$G = \Phi(X)^\top \Phi(X) = [k(\mathbf{x}_i, \mathbf{x}_j)]_{i,j}$$

- where  $k(.,.)$  is the reproducing kernel

# Example of Kernel PCA



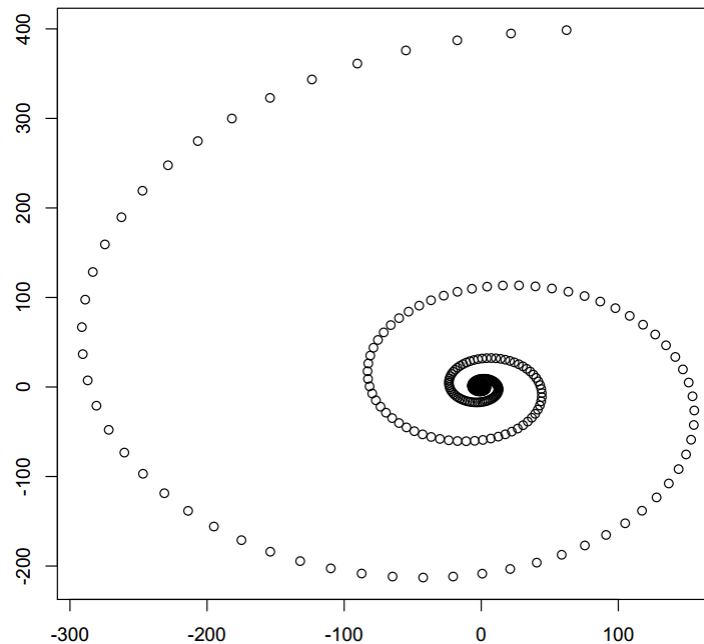
# Example of Kernel PCA



# Nonlinear Dimension Reduction (Manifold Learning)

# Manifold Learning

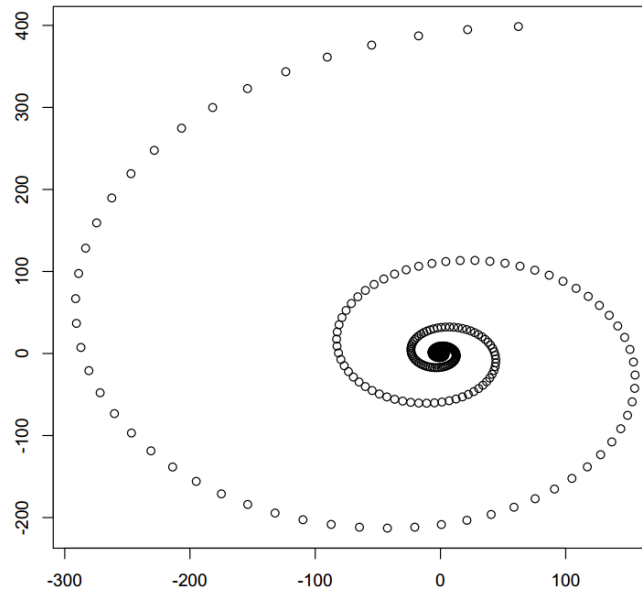
- ◆ **Manifold:** a smooth, curved subset of an Euclidean space, in which it is embedded



- ◆ A  $d$ -dim manifold can be arbitrarily well-approximated by a  $d$ -dim linear subspace, the tangent space, by taking a sufficiently small region about any point



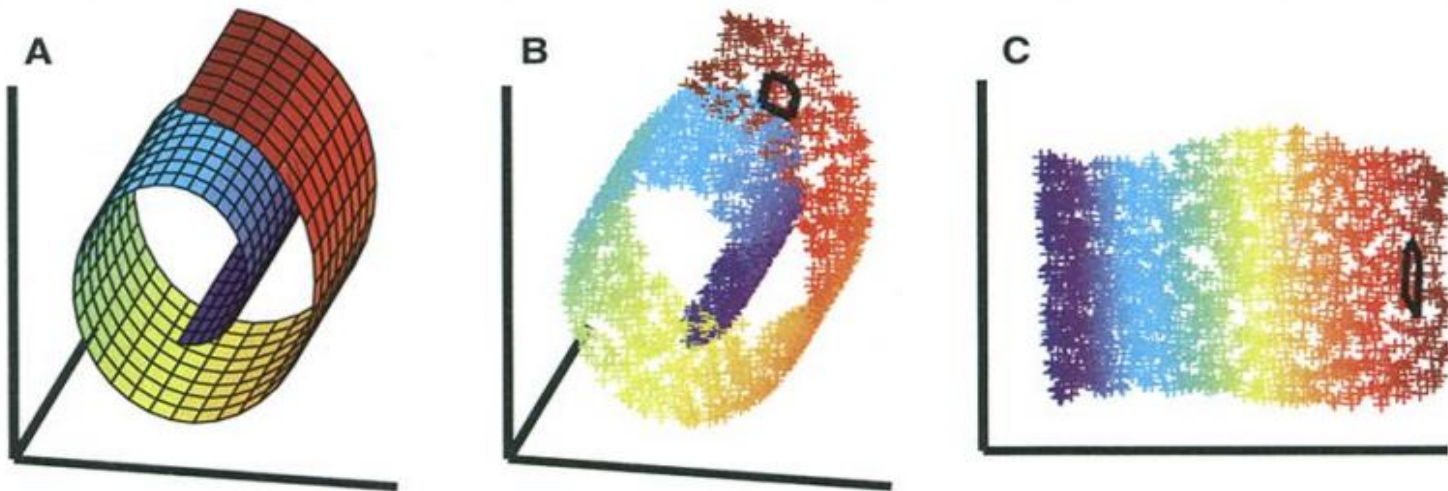
# Manifold Learning



- ◆ If our data come from a manifold, we should be able to do a local linear approximation around each part of the manifold, and then smoothly interpolate them together into a single global system
- ◆ To do dimension reduction, we want to find the global low-dim coordinates

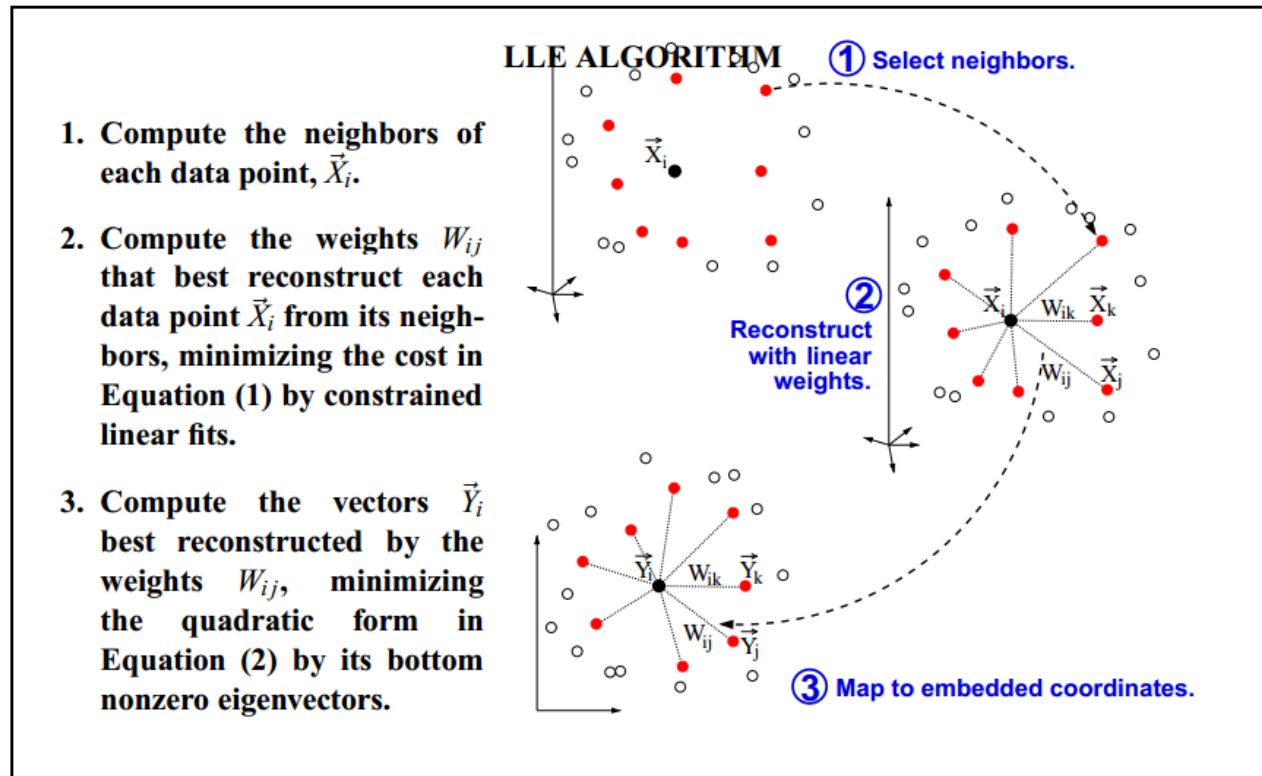
# Locally linear embedding (LLE)

- ◆ A nonlinear dimension reduction technique to preserve neighborhood structure



- ◆ **Intuition:** nearby points in the high dimensional space remain nearby and similarly co-located w.r.t one another in the low dimensional space

# How does LLE work?



**Step 2:** minimize reconstruction error

$$\begin{aligned} \min_W \epsilon(W) &= \sum_i \|\mathbf{x}_i - \sum_{j \in \mathcal{N}_i} W_{ij} \mathbf{x}_j\|_2^2 \\ \text{s.t.} \quad &\sum_j W_{ij} = 1, \quad \forall i \end{aligned}$$

**Step 3:** neighborhood-preserving embedding

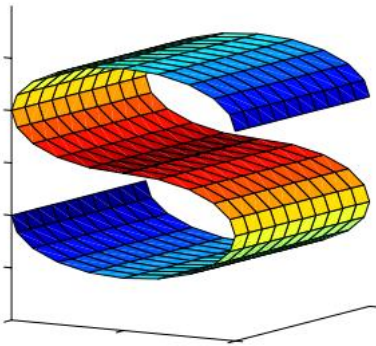
$$\min_Y \Phi(Y) = \sum_i \|\mathbf{y}_i - \sum_{j \in \mathcal{N}_i} W_{ij} \mathbf{y}_j\|_2^2$$

geometric structure  $W$

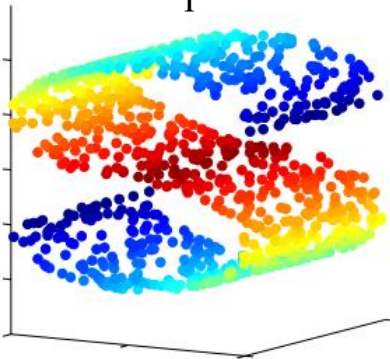
# Implementation details

- ◆ Free parameter:  $K$  – number of neighbors per data point

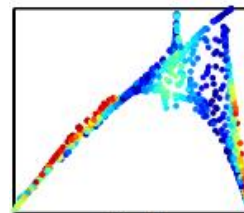
Original manifold



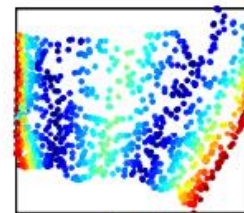
samples



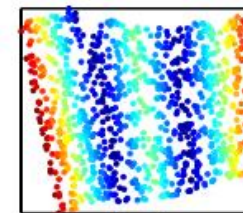
Embedding results by LLE with various  $K$



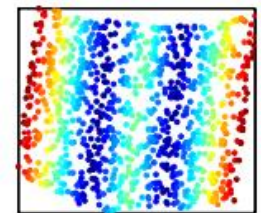
$K = 5$



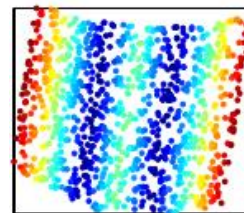
$K = 6$



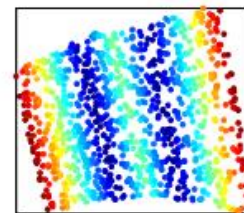
$K = 8$



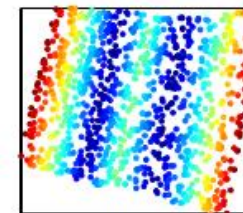
$K = 10$



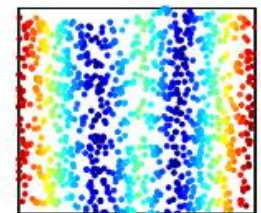
$K = 12$



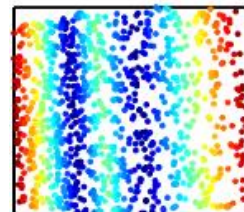
$K = 14$



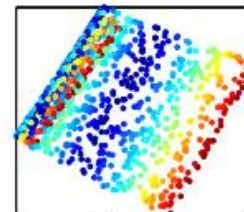
$K = 16$



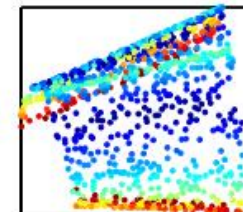
$K = 18$



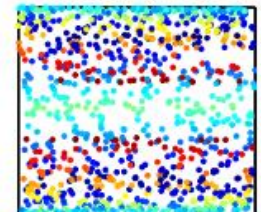
$K = 20$



$K = 30$



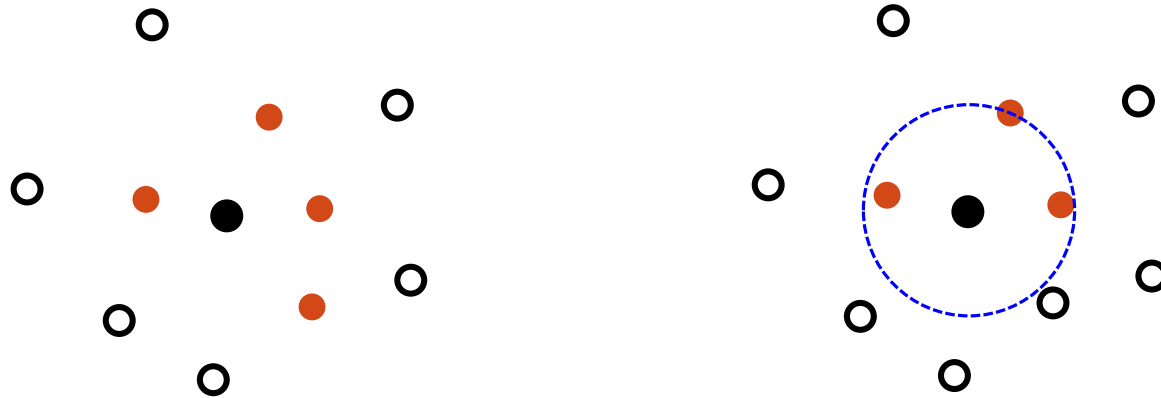
$K = 40$



$K = 60$

# Implementation details

◆ **Step 1:** choose neighborhood – many choices



◆ **Note:** different points can have different numbers of neighbors

# Implementation details

◆ Step 2: minimize reconstruction error

$$\begin{aligned} \min_W \quad & \epsilon(W) = \sum_i \|\mathbf{x}_i - \sum_{j \in \mathcal{N}_i} W_{ij} \mathbf{x}_j\|_2^2 \\ \text{s.t.:} \quad & \sum_{j \in \mathcal{N}_i} W_{ij} = 1, \quad \forall i \end{aligned}$$

□ each data point can be done in parallel – **locality**

$$\|\mathbf{x}_i - \sum_j W_{ij} \mathbf{x}_j\|_2^2 = \|\sum_j W_{ij} (\mathbf{x}_i - \mathbf{x}_j)\|_2^2 = \sum_{jk} W_{ij} W_{ik} G_{jk} = W_i^\top G W_i$$

$$G_{jk} = (\mathbf{x}_i - \mathbf{x}_j)^\top (\mathbf{x}_i - \mathbf{x}_k), \quad \forall j, k \in \mathcal{N}_i$$

□ Solution (Lagrange methods):

$$2GW_i - \lambda I = 0$$

$$\sum_j W_{ij} = 1$$



$$W_i = \frac{G^{-1} \mathbf{1}}{\mathbf{1}^\top G^{-1} \mathbf{1}}$$

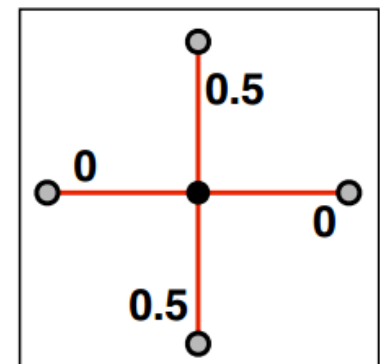
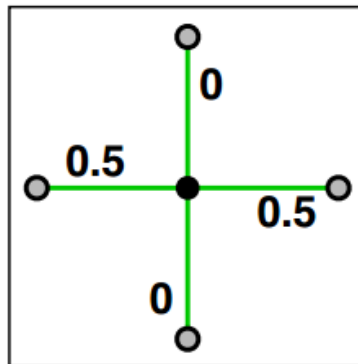
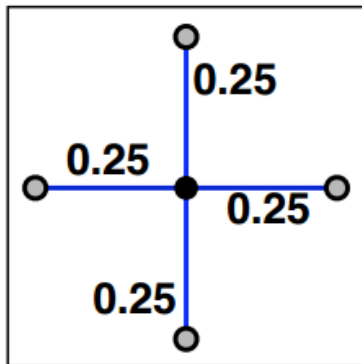
# Implementation details

## ◆ *What's happening if $K > p$ ?*

- The space spanned by  $k$  distinct vectors is the whole space
- Each data point can be perfectly reconstructed from its neighbors

$$\mathbf{x}_i = \sum_{j \in \mathcal{N}_i} W_{ij} \mathbf{x}_j$$

- $G$  is singular! (**fewer constraints than parameters**)
  - The reconstruction weights are no longer uniquely defined
- Example ( $D=2, K=4$ )



# Implementation details

## ◆ *What's happening if $K > p$ ?*

- The space spanned by  $k$  distinct vectors is the whole space
- Each data point can be perfectly reconstructed from its neighbors

$$\mathbf{x}_i = \sum_{j \in \mathcal{N}_i} W_{ij} \mathbf{x}_j$$

- $G$  is singular!
- The reconstruction weights are no longer uniquely defined

## ◆ Regularized opt. problem: (save ill-posed problems)

$$\min_{W_i} W_i^\top G W_i + \gamma W_i^\top W_i$$

$$\text{s.t.: } \sum_j W_{ij} = 1, \forall i$$

- Solution (Lagrange methods):

$$2(G + \gamma I)W_i - \lambda I = 0$$

$$\sum_j W_{ij} = 1$$



$$W_i = \frac{(G + \gamma I)^{-1} \mathbf{1}}{\mathbf{1}^\top (G + \gamma I)^{-1} \mathbf{1}}$$



# Implementation details

## ◆ Step 3: neighborhood-preserving embedding

$$\min_Y \Phi(Y) = \sum_i \|\mathbf{y}_i - \sum_{j \in \mathcal{N}_i} W_{ij} \mathbf{y}_j\|_2^2$$

$$\text{s.t. : } \sum_i \mathbf{y}_i = \mathbf{0} \quad \text{centered around the origin}$$

$$\frac{1}{N} \sum_i \mathbf{y}_i \mathbf{y}_i^\top = I \quad \text{unit covariance}$$

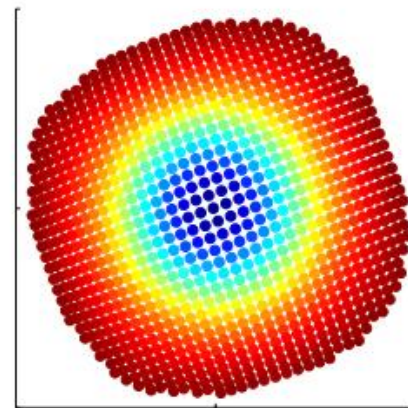
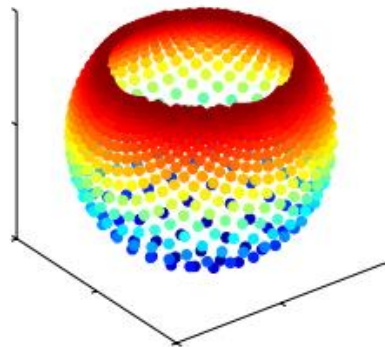
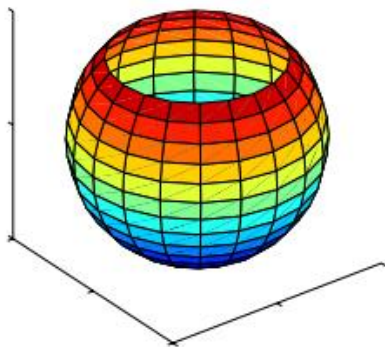
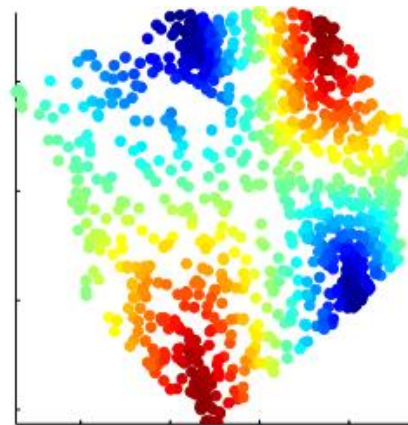
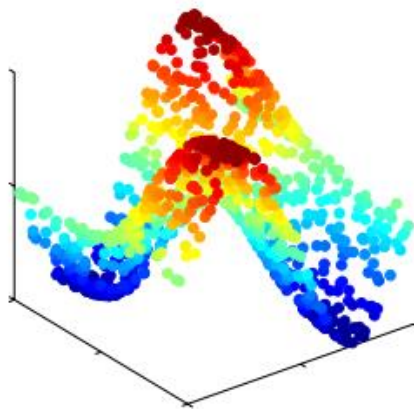
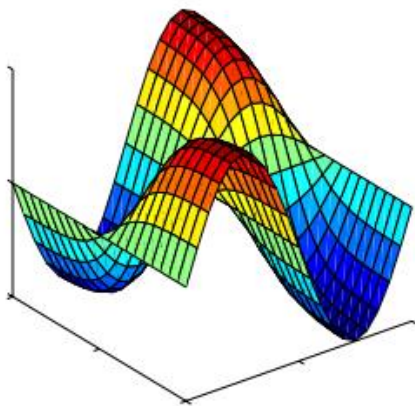
- all data points are coupled together – **global coordinates**
- Solution (Lagrange methods) – eigenvalue problem:

$$F = \frac{1}{2} \sum_i \|\mathbf{y}_i - \sum_j W_{ij} \mathbf{y}_j\|_2^2 - \frac{1}{2} \sum_{\alpha\beta} \lambda_{\alpha\beta} \left( \frac{1}{N} \sum_i y_{i\alpha} y_{i\beta} - \delta_{\alpha\beta} \right)$$

$$(\mathbf{1} - W)^\top (\mathbf{1} - W) Y = \frac{1}{N} Y \Lambda, \text{ where } \Lambda_{\alpha\beta} = \lambda_{\alpha\beta}$$

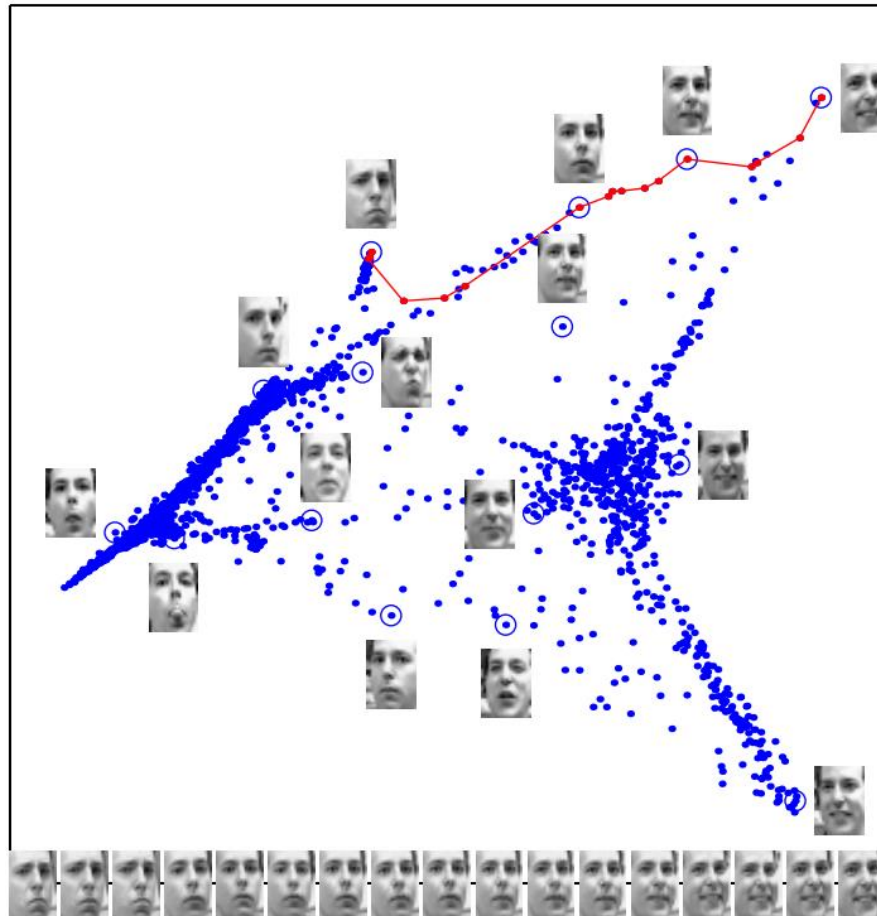
Find the  $d$  eigenvectors with the lowest eigenvalues

# More examples



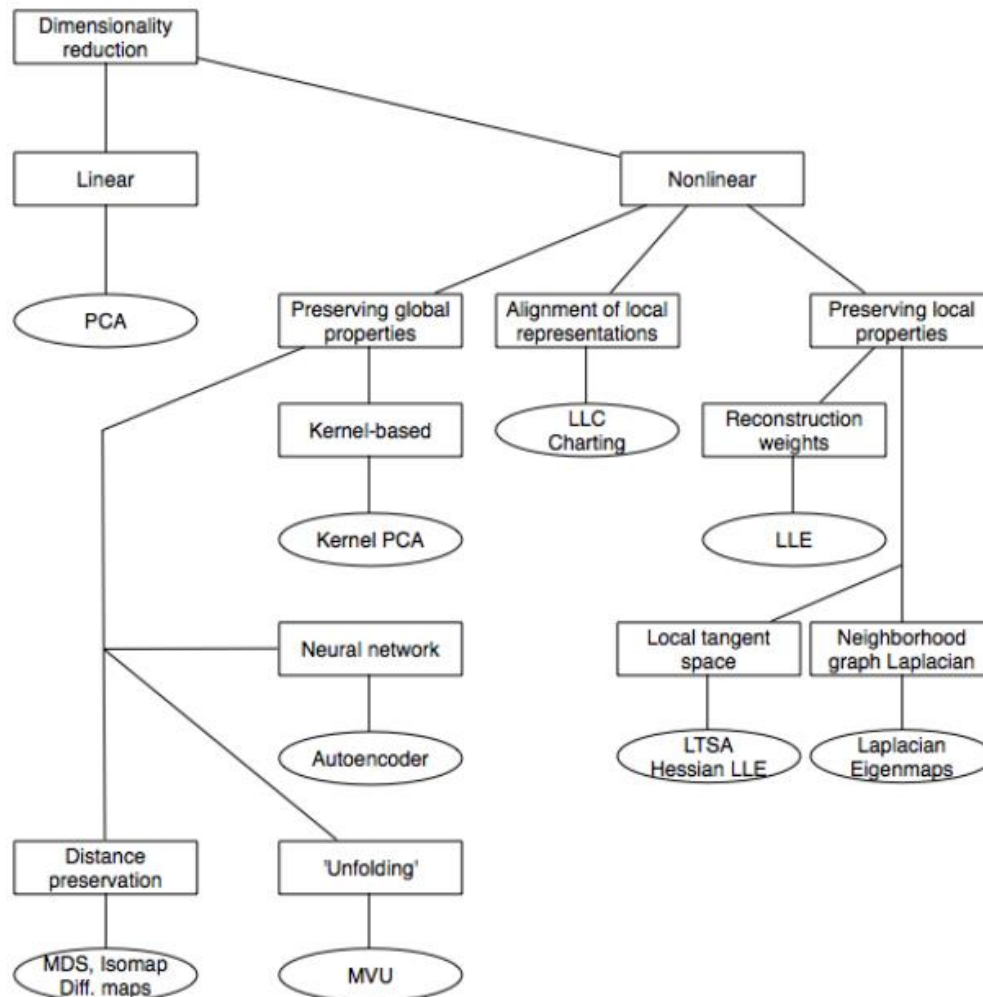
## More examples

◆ 1965 grayscale 20 x 28 images ( $D=560$ );  $K = 12$



[Roweis & Saul, Science, Vol 290, 2000; Saul & Roweis, JMLR 2003]

# Many other algorithms



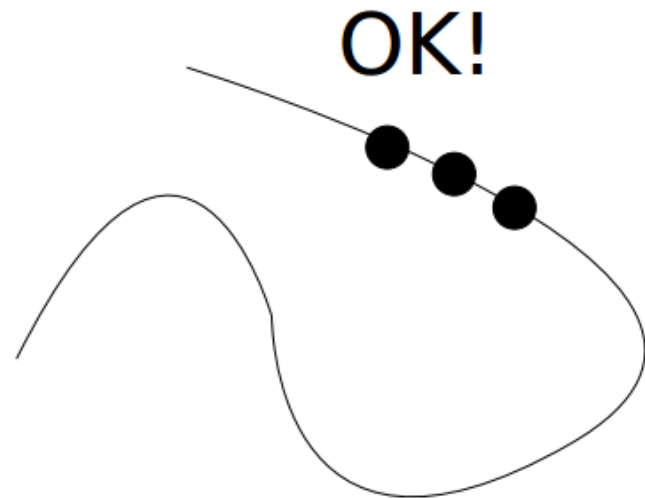
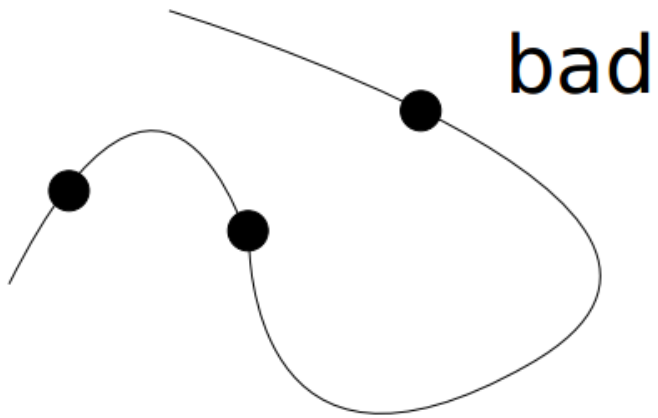
[van der Maaten et al., Dimension Reduction: A Comparative Review, 2008]

<i>Technique</i>	<i>Convex</i>	<i>Parameters</i>	<i>Computational</i>	<i>Memory</i>
PCA	yes	none	$O(D^3)$	$O(D^2)$
MDS	yes	none	$O(n^3)$	$O(n^2)$
Isomap	yes	$k$	$O(n^3)$	$O(n^2)$
MVU	yes	$k$	$O((nk)^3)$	$O((nk)^3)$
Kernel PCA	yes	$\kappa(\cdot, \cdot)$	$O(n^3)$	$O(n^2)$
Diffusion maps	yes	$\sigma, t$	$O(n^3)$	$O(n^2)$
Autoencoders	no	net size	$O(inw)$	$O(w)$
LLE	yes	$k$	$O(pn^2)$	$O(pn^2)$
Laplacian Eigenmaps	yes	$k, \sigma$	$O(pn^2)$	$O(pn^2)$
Hessian LLE	yes	$k$	$O(pn^2)$	$O(pn^2)$
LTSA	yes	$k$	$O(pn^2)$	$O(pn^2)$
LLC	no	$m, k$	$O(imd^3)$	$O(nmd)$
Manifold charting	no	$m$	$O(imd^3)$	$O(nmd)$

**Note:**  $n$  is  $N$ ;  $D$  is  $p$  in our slides

# No Free Lunch

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



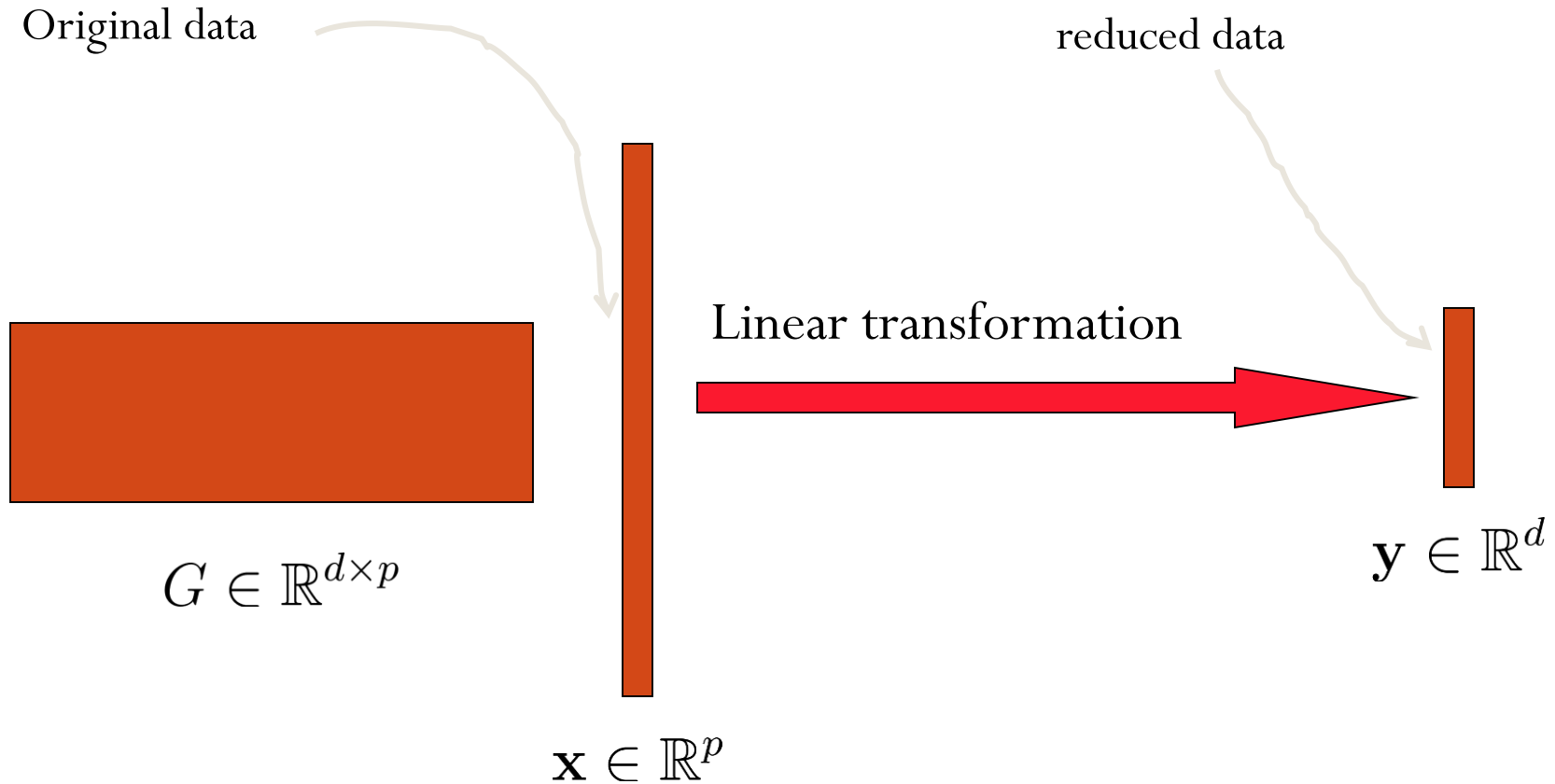
# Matlab Toolbox



**Laurens van der Maaten**

- ◆ <http://lvdmaaten.github.io/drtoolbox/> (34 techniques for dimensionality reduction and metric learning)

# What is dimension reduction? – linear case

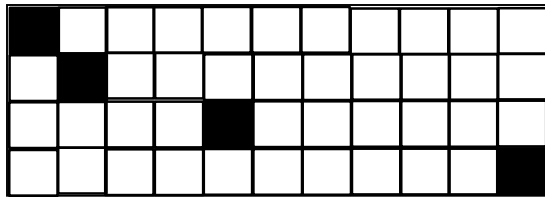


$$G \in \mathbb{R}^{d \times p} : \mathbf{x} \rightarrow \mathbf{y} = G\mathbf{x} \in \mathbb{R}^d$$



# What is feature selection?

Original data



$$G \in \{0, 1\}^{d \times p}$$

$$\sum_j G_{ij} = 1$$

$$\mathbf{x} \in \mathbb{R}^p$$

reduced data

Linear transformation

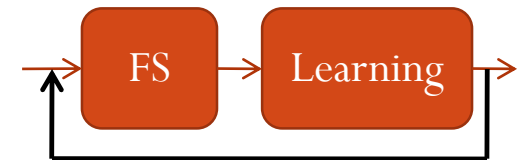
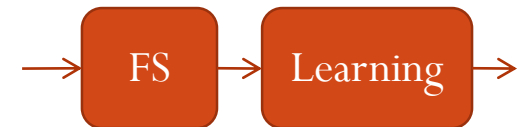


$$\mathbf{y} \in \mathbb{R}^d$$

$$G \in \{0, 1\}^{d \times p} : \mathbf{x} \rightarrow \mathbf{y} = G\mathbf{x} \in \mathbb{R}^d$$

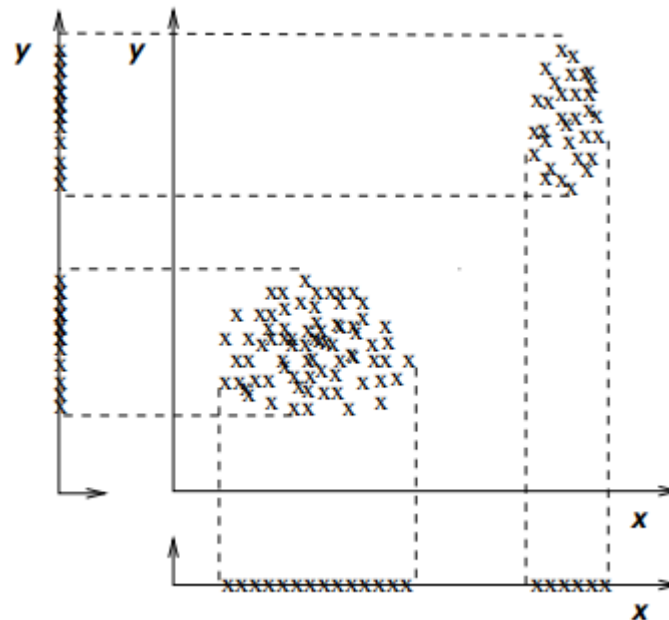
# Feature selection methods

- ◆ FS is popular in supervised learning by maximizing some function of predictive accuracy
- ◆ Selecting an optimal set of features is NP-hard (Weston et al., 2003)
- ◆ Approximate methods:
  - Filter methods [Kira & Rendell, 1992] (**Separate**)
    - Based on feature ranking (**individual predictive power**);
    - A pre-processing step and independent of prediction models (**optimal under very strict assumptions!**) [Guyon & Elisseeff, 2003]
  - Wrapper methods [Kohavi & John, 1997] (**Half-integrated**)
    - Use learning machine as a **black box** to score subsets of variables according to their predictive power
    - Can waste of resources to do many re-training!
  - Embedded methods (**Integrated**)
    - Perform FS during the process of training; Usually specific to given learning machines
    - Data efficient and Can avoid many re-training!



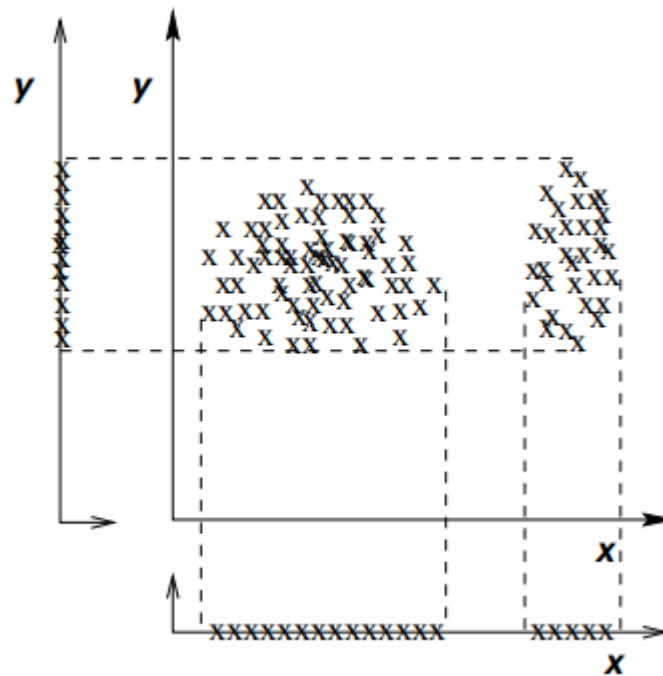
# Unsupervised feature selection

- ◆ x and y are redundant in discriminating the two clusters (i.e., each one decides the clustering results)



# Unsupervised feature selection

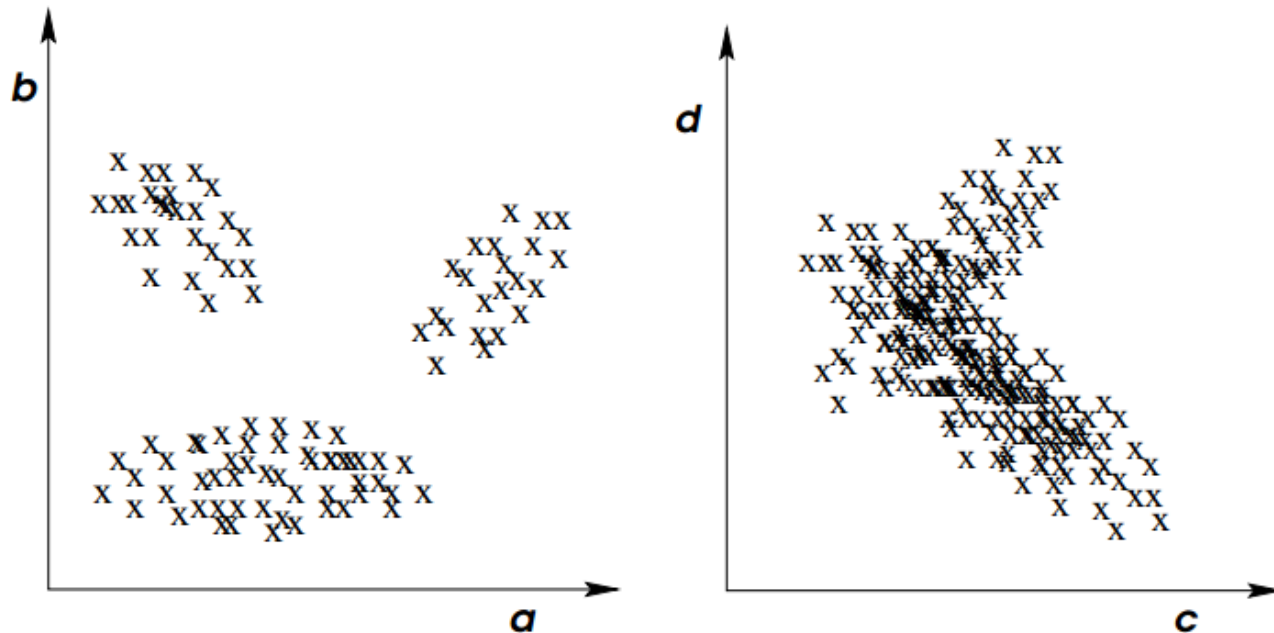
- ◆  $y$  is irredundant in discriminating the two clusters



- ◆ **Note:** irrelevant features can misguide clustering

# Unsupervised feature selection

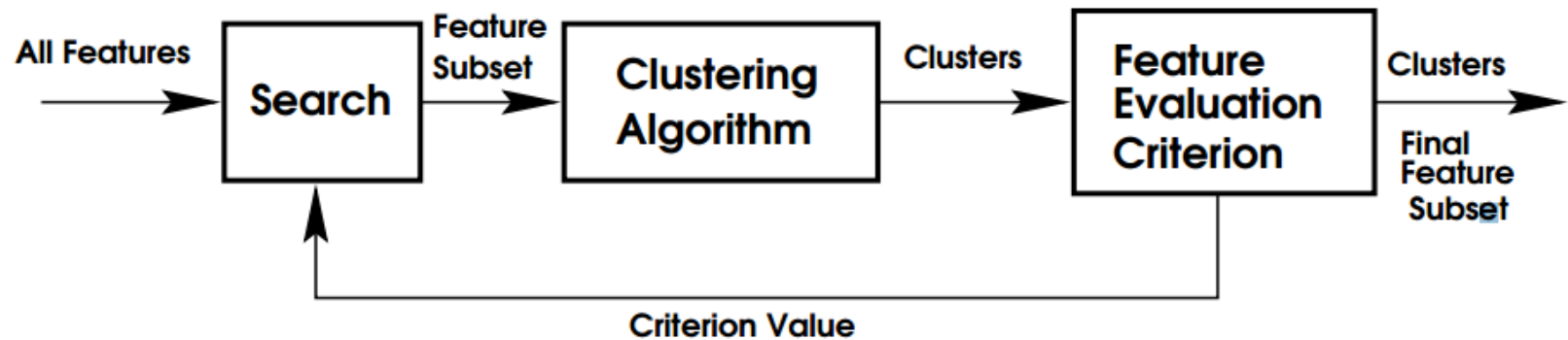
- ◆ Different feature subsets lead to different clustering



- ◆ Which one should we pick?

# Unsupervised feature selection

◆ A wrapper framework for unsupervised feature selection



◆ Some key issues:

- ❑ Different feature subsets have different numbers of clusters
- ❑ The feature selection criteria have biases w.r.t feature subset dimensionality

# Feature Search

- ◆ An exhaustive search is intractable (  $2^d$  possible feature subsets)
- ◆ Greedy search:
  - Sequential forward search
    - Starting from 0 features
    - Add one feature at a time to maximize the gain of some criterion
    - Stop when no improvement
  - Sequential backward elimination
    - Start from the full set
    - Eliminate one feature at a time to minimize the loss of some criterion
    - Stop when no change

# Clustering algorithm

- ◆ Any clustering algorithms can be used in the wrapper framework



# Feature subset selection criteria

- ◆ “different classifications [clusterings] are right for different purpose, so we cannot say any one classification is best” (Hartigan, 1985)
- ◆ Some commonly used criteria:
  - Scatter separability (applicable for any clustering methods)

$$S_w = \sum_{j=1}^K \pi_j \mathbb{E}[(X - \mu_j)(X - \mu_j)^\top | C_j] = \sum_{j=1}^K \pi_j \Sigma_j$$

$$S_b = \sum_{j=1}^K \pi_j (\mu_j - M_0)(\mu_j - M_0)^\top$$

$$M_0 = \mathbb{E}[X] = \sum_{j=1}^K \pi_j \mu_j$$


$$\text{trace}(S_w^{-1} S_b)$$

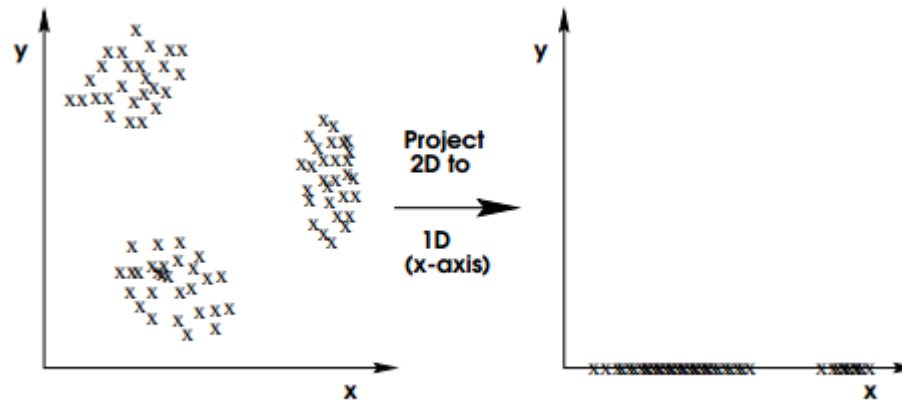
## Feature subset section criteria

- ◆ “different classifications [clusterings] are right for different purpose, so we cannot say any one classification is best” (Hartigan, 1985)
- ◆ Some commonly used criteria:
  - Maximum likelihood (applicable for probabilistic methods)

$$P(D|\text{a feature subset})$$

# The need for finding the number of clusters

- ◆ The number of clusters varies with dimension



- ◆ Some selection methods exist for K (Dy & Brodley, 2003)

# What you need to know

- ◆ Motivations for dimension reduction
- ◆ Derivations of PCA
- ◆ LLE
- ◆ Feature selection

◆ Homework 1 out, due in two weeks!

◆ Reading materials:

- Chapter 12 of Bishop's PRML
- References in slides