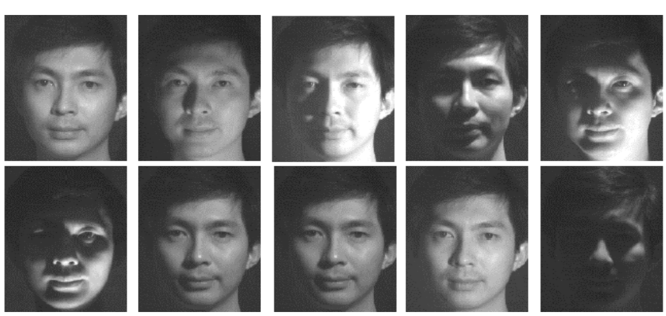
Face Recognition

Ying Wu
yingwu@ece.northwestern.edu

Electrical Engineering & Computer Science
Northwestern University, Evanston, IL
http://www.ece.northwestern.edu/~yingwu

Recognizing Faces?



Lighting

View

Outline

- ✓ Bayesian Classification
- ✓ Principal Component Analysis (PCA)
- ✓ Fisher Linear Discriminant Analysis (LDA)
- ✓ Independent Component Analysis (ICA)

Bayesian Classification

- ✓ Classifier & Discriminant Function
- ✓ Discriminant Function for Gaussian
- ✓ Bayesian Learning and Estimation

Classifier & Discriminant Function

- Discriminant function $g_i(x)$ i=1,…,C
- Classifier

$$x \to \omega_i \quad \text{if} \quad g_i(x) > g_j(x) \quad \forall j \neq i$$

- Example

$$g_i(x) = p(\omega_i \mid x)$$

$$g_i(x) = p(x \mid \omega_i \mid) p(\omega_i)$$

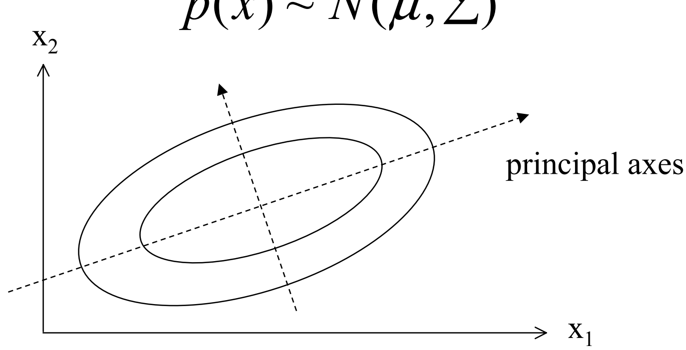
$$g_i(x) = \ln p(x \mid \omega_i) + \ln p(\omega_i)$$

The choice of D-function is not unique

- Decision boundary

Multivariate Gaussian

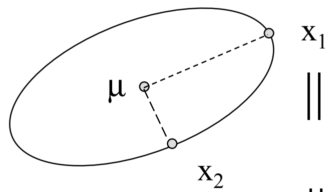
$$p(x) \sim N(\mu, \Sigma)$$



✓ The principal axes (the direction) are given by the eigenvectors of Σ;

✓ The length of the axes (the uncertainty) is given by the eigenvalues of Σ

Mahalanobis Distance



$$\| x_1 - \mu \|_2 > \| x_2 - \mu \|_2$$

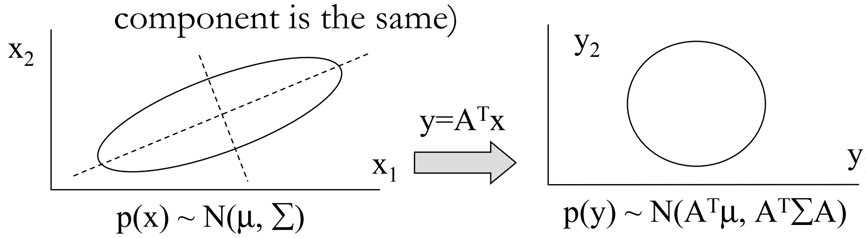
$$\| x_1 - \mu \|_M = \| x_2 - \mu \|_M$$

Mahalanobis distance is a normalized distance

$$\| x - c \|_M = \sqrt{(x-\mu)^T \Sigma^{-1} (x-\mu)}$$

Whitening

- Whitening:
  - Find a linear transformation (rotation and scaling) such that the covariance becomes an identity matrix (i.e., the uncertainty for each component is the same)



solution : $A = U^T \Lambda^{-\frac{1}{2}}$ where $\Sigma = U^T \Lambda U$

Disc. Func. for Gaussian

- Minimum-error-rate classifier

$$g_i(x) = \ln p(x \mid \omega_i) + \ln p(\omega_i)$$

$$g_i(x) = -\frac{1}{2}(x-\mu_i)^T \Sigma_i^{-1}(x-\mu_i) - \frac{d}{2}\ln 2\pi - \frac{1}{2}\ln|\Sigma_i| + \ln p(\omega_i)$$

Case I: $\Sigma_i = \sigma^2 I$

$$g_i(x) = -\frac{\|x-\mu_i\|^2}{2\sigma^2} + \ln p(\omega_i) = -\frac{1}{2\sigma^2}\left[x^T x - 2\mu_i^T x + \mu_i^T \mu\right] + \ln p(\omega_i)$$

constant

$$g_i(x) = -\left(\frac{1}{\sigma^2}\mu_i\right)^T x + \left(-\frac{1}{2\sigma^2}\mu_i^T \mu + \ln p(\omega_i)\right)$$

$$= W_i^T x + W_{i0}$$ Liner discriminant function

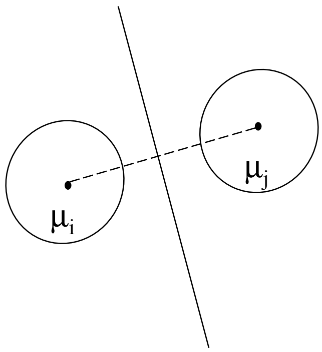
Boundary: $g_i(x) = g_j \Rightarrow W^T(x - x_0) = 0$ where

$$W = \mu_i - \mu_j$$

$$x_0 = \tfrac{1}{2}(\mu_i + \mu_j) - \tfrac{\sigma^2}{\|\mu_i - \mu_j\|^2}\ln\tfrac{p(\omega_i)}{p(\omega_j)}(\mu_i - \mu_j)$$

Example

- Assume $p(\omega_i)=p(\omega_j)$



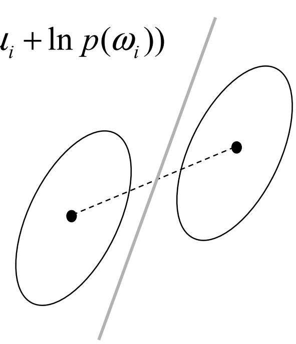
Let's derive the decision boundary:

$$(\mu_i-\mu_j)^T\left[x-\frac{\mu_i+\mu_j}{2}\right]=0$$

Case II: $\Sigma_i=\Sigma$

$$g_i(x)=-\tfrac{1}{2}(x-\mu_i)^T\Sigma^{-1}(x-\mu_i)+\ln p(\omega_i)$$

$$\Leftrightarrow g_i(x)=(\Sigma^{-1}\mu_i)^T x+(-\tfrac{1}{2}\mu_i^T\Sigma^{-1}\mu_i+\ln p(\omega_i))$$
$$=W_i^T x+W_{i0}$$



The decision boundary is still linear:

$$W^T(x-x_0)=0 \quad \text{where}$$
$$W=\Sigma^{-1}(\mu_i-\mu_j)$$
$$x_0=\frac{1}{2}(\mu_i+\mu_j)-\frac{\ln p(\omega_i)-\ln p(\omega_j)}{(\mu_i-\mu_j)^T\Sigma^{-1}\mu_i-\mu_j)}(\mu_i-\mu_j)$$

Case III: Σ_i = arbitrary

$$g_i(x) = x^T A_i x + W_i^T x + W_{i0} \qquad \text{where}$$

$$A_i = -\tfrac{1}{2}\Sigma_i^{-1}$$

$$W_i = \Sigma_i^{-1}\mu_i$$

$$W_{i0} = -\tfrac{1}{2}\mu_i^T \Sigma_i^{-1}\mu_i - \tfrac{1}{2}\ln|\Sigma_i| + \ln p(\omega_i)$$

The decision boundary is no longer linear, but hyperquadrics!

Bayesian Learning

- Learning means "training"
- i.e., estimating some unknowns from "training data"
- WHY?
  - It is very difficult to specify these unknowns
  - Hopefully, these unknowns can be recovered from examples collected.

Maximum Likelihood Estimation

- Collected examples D={x_1,x_2,…,x_n}
- Estimate unknown parameters θ in the sense that the data likelihood is maximized
- Likelihood $p(D|\theta)=\prod_{k=1}^{n} p(x_k|\theta)$
- Log Likelihood

$$L(\theta)=\ln p(D|\theta)=\sum_{k=1}^{n}\ln p(x_k|\theta)$$

- ML estimation

$$\theta^* = \arg\max_{\theta} p(D|\theta) = \arg\max_{\theta} L(D|\theta)$$

Case I: unknown μ

$$\ln p(x_k|\mu) = -\tfrac{1}{2}\ln[(2\pi)^d |\Sigma|] - \tfrac{1}{2}(x_k-\mu)^T \Sigma^{-1}(x_k-\mu)$$

$$\frac{\partial \ln p((x_k|\mu)}{\partial \mu} = \Sigma^{-1}(x_k-\mu)$$

$$\sum_{k=1}^{n}\Sigma^{-1}(x_k-\hat{\mu})=0 \quad \Leftrightarrow \quad \hat{\mu}=\frac{1}{n}\sum_{k=1}^{n}x_k$$

Case II: unknown μ and Σ

$$\ln p(x_k \mid \mu, \Delta) = -\tfrac{1}{2}\ln 2\pi\Delta - \tfrac{1}{2\Delta}(x_k - \mu)^2 \qquad \text{let } \Delta = \sigma^2$$

$$\frac{\partial \ln p(x_k \mid \mu, \Delta)}{\partial \mu} = \frac{1}{\sigma}(x_k - \mu)$$

$$\frac{\partial \ln p(x_k \mid \mu, \Delta)}{\partial \Delta} = -\frac{1}{2\Delta} + \frac{(x_k - \mu)^2}{2\Delta^2}$$

$$\Rightarrow$$

$$\sum_{k=1}^{n} \frac{1}{\hat{\Delta}}(x_k - \hat{\mu}) = 0$$

$$-\sum_{k=1}^{n} \frac{1}{\hat{\Delta}} + \sum_{k=1}^{n} \frac{(x_k - \hat{\mu})^2}{\hat{\Delta}^2} = 0$$

$$\Rightarrow$$

$$\hat{\mu} = \frac{1}{n}\sum_{k=1}^{n} x_k$$

$$\hat{\sigma}^2 = \hat{\Delta}^2 = \frac{1}{n}\sum_{k=1}^{n} (x_k - \hat{\mu})^2$$

generalize $\Rightarrow$

$$\hat{\mu} = \frac{1}{n}\sum_{k=1}^{n} x_k$$

$$\hat{\Sigma} = \frac{1}{n}\sum_{k=1}^{n} (x_k - \hat{\mu})(x_k - \hat{\mu})^T$$

Bayesian Estimation

- Collected examples $D=\{x_1, x_2, \ldots, x_n\}$, drawn independently from a fixed but unknown distribution p(x)
- Bayesian learning is to use D to determine p(x|D), i.e., to learn a p.d.f.
- p(x) is unknown, but has a parametric form with parameters $\theta \sim p(\theta)$
- Difference from ML: in Bayesian learning, θ is not a value, but a random variable and we need to recover the distribution of θ, rather than a single value.

Bayesian Estimation

$$p(x \mid D) = \int p(x, \theta \mid D) d\theta = \int p(x \mid \theta) p(\theta \mid D) d\theta$$

- This is obvious from the total probability rule, i.e., p(x|D) is a weighted average over all θ
- If p(θ |D) peaks very sharply about some value θ^*, then p(x|D) ~ p(x| θ^*)

The Univariate Case

- assume μ is the only unknown, p(x|μ)~N(μ, σ^2)
- μ is a r.v., assuming a prior p(μ) ~ N(μ_0, σ_0^2), i.e., μ_0 is the best guess of μ, and σ_0 is the uncertainty of it.

$$p(\mu \mid D) \propto p(D \mid \mu) p(\mu) = \prod_{k=1}^{n} p(x_k \mid \mu) p(\mu)$$

where $\quad p(x_k \mid \mu) \sim N(\mu, \sigma^2), \qquad p(\mu) \sim N(\mu_o, \sigma_0^2)$

$$p(\mu \mid D) \propto \exp\left[-\tfrac{1}{2} \left((\tfrac{n}{\sigma^2} + \tfrac{1}{\sigma_0^{\ 2}} \right) \mu^2 - 2(\tfrac{1}{\sigma^2} \sum_k x_k + \tfrac{\mu_0}{\sigma_0^{\ 2}}) \mu \right]$$

P(μ|D) is also a Gaussian for any # of training examples

The Univariate Case

if we let $p(\mu \mid D) \sim N(\mu_n, \sigma_n^2)$

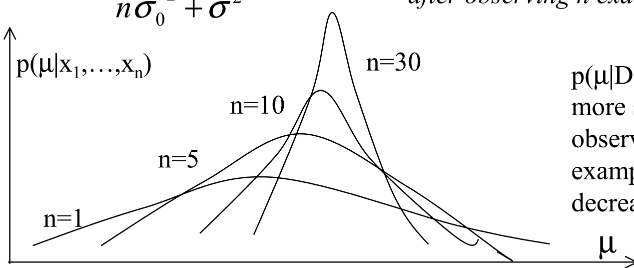
we have

$$\mu_n = \left(\frac{n\sigma_0^{\,2}}{n\sigma_0^{\,2} + \sigma^2} \right) \hat{\mu}_n + \left(\frac{\sigma^2}{n\sigma_0^{\,2} + \sigma^2} \right) \mu_0$$

The best guess for μ after observing n examples

$$\sigma_n^2 = \frac{\sigma_0^{\,2}\sigma^2}{n\sigma_0^{\,2} + \sigma^2}$$

σ_n measures the uncertainty of this guess after observing n examples



p(μ|D) becomes more and more sharply peaked when observing more and more examples, i.e., the uncertainty decreases.

$$\sigma_n^2 \to \frac{\sigma^2}{n}$$

The Multivariate Case

$p(x \mid \mu) \sim N(\mu, \Sigma)$ $\qquad$ $p(\mu) \sim N(\mu_0, \Sigma_0)$

let $p(\mu \mid D) \sim N(\mu_n, \Sigma_n)$, $\quad$ we have

$$\mu_n = \Sigma_0 (\Sigma_0 + \tfrac{1}{n}\Sigma)^{-1} \hat{\mu}_n + \tfrac{1}{n}\Sigma(\Sigma_0 + \tfrac{1}{n}\Sigma)^{-1} \hat{\mu}_0$$

$$\Sigma_n = \Sigma_0 (\Sigma_0 + \tfrac{1}{n}\Sigma)^{-1} \tfrac{1}{n}\Sigma$$

where $\hat{\mu}_n = \dfrac{1}{n}\sum_{k=1}^{n} x_k$

and $\quad p(x \mid D) = \int p(x \mid \mu) p(\mu \mid D) d\mu \sim N(\mu_n, \Sigma + \Sigma_n)$

PCA and Eigenface

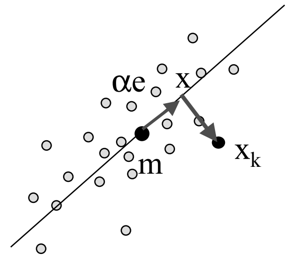
- ✓ Principal Component Analysis (PCA)
- ✓ Eigenface for Face Recognition

PCA: motivation

- Pattern vectors are generally confined within some low-dimensional subspaces
- Recall the basic idea of the Fourier transform
  - A signal is (de)composed of a linear combination of a set of basis signal with different frequencies.

PCA: idea

$$\vec{x} = \vec{m} + \alpha \vec{e}$$



$$J(\alpha_1, \ldots, \alpha_n, e) = \sum_{k=1}^{n} \| (m + \alpha_k e) - x_k \|^2$$
$$= \sum \alpha_k^2 \| e \|^2 - 2 \sum \alpha_k e^T (x_k - m) + \sum \| x_k - m \|^2$$

$$\frac{\partial J}{\partial \alpha_k} = 2\alpha_k - 2e^T (x_k - m) = 0 \quad \Rightarrow \quad \alpha_k = e^T (x_k - m)$$

PCA

$$J(e) = \sum \alpha_k^2 - 2 \sum \alpha_k^2 + \sum \| x_k - m \|^2$$
$$= -e^T \sum (x_k - m)(x_k - m)^T e + \sum \| x_k - m \|^2$$
$$= -eSe + \sum \| x_k - m \|^2$$

$$\arg\min_e J(e) = \arg\max_e e^T Se \qquad \text{s.t.} \quad \| e \| = 1$$

$$e^* = \arg\max_e e^T Se + \lambda (e^T e - 1)$$
$$Se - \lambda e = 0, \qquad \text{i.e.,} \qquad e^T Se = 1$$

To maximize $e^T Se$, we need to select λ_{max}

Algorithm

- Learning the principal components from $\{x_1, x_2, \ldots, x_n\}$

(1) $m = \frac{1}{n}\sum_{k=1}^{n} x_k, \quad A = [x_1 - m, \ldots, x_n - m]$

(2) $S = \sum_{k=1}^{n} (x_k - m)(x_k - m)^T = AA^T$

(3) eigenvalue decomposition $S = U^T \Sigma U$

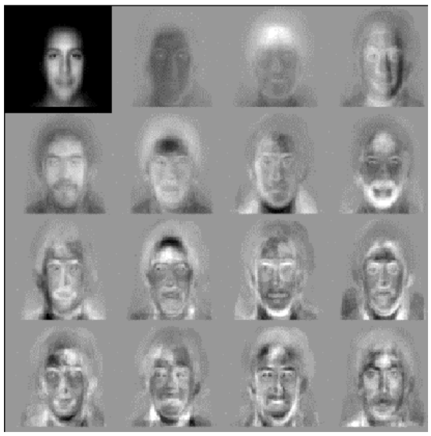
(4) sorting λ_i and u_i

(5) $P^T = [u_1^T, u_2^T, \ldots, u_m^T]$

PCA for Face Recognition

- Training data $D=\{x_1, \ldots, x_M\}$
  - Dimension (stacking the pixels together to make a vector of dimension N)
  - Preprocessing
    - ✓ cropping
    - ✓ normalization
- These faces should lie in a "face" subspace
- Questions:
  - What is the dimension of this subspace?
  - How to identify this subspace?
  - How to use it for recognition?

Eigenface



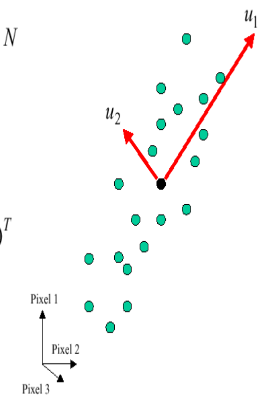
$$\{x_i\}_{i=1}^{M} \quad x \in R^N \qquad M < N$$

$$\mu = \frac{1}{M}\sum_{i=1}^{M} x_i$$

$$S = \sum_{i=1}^{M}(x_i - \mu)(x_i - \mu)^T$$

$$S = ULU^T$$

$$y = U^T(x - \mu)$$



The EigenFace approach: M. Turk and A. Pentland, 1992

An Issue

- In general, N >> M
- However, S, the covariance matrix, is NxN!
- Difficulties:
  - S is ill-conditioned. Rank(S)<<N
  - The computation of the eigenvalue decomposition of S is expensive when N is large
- Solution?

Solution I:

- Let's do eigenvalue decomposition on A^TA, which is a MxM matrix
- $A^TAv=\lambda v$
- $\rightarrow AA^TAv= \lambda Av$
- To see is clearly! $(AA^T)(Av)= \lambda(Av)$
- i.e., if v is an eigenvector of A^TA, then Av is the eigenvector of AA^T corresponding to the same eigenvalue!
- Note: of course, you need to normalize Av to make it a unit vector

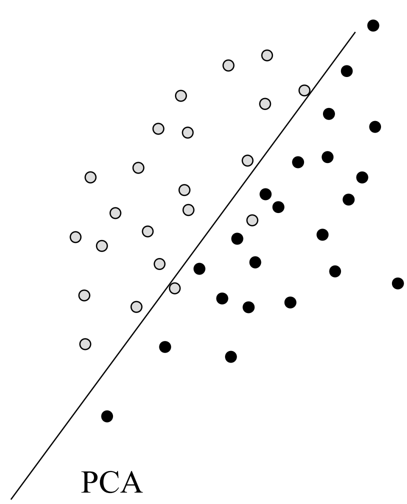
Solution II:

- You can simply use SVD (singular value decomposition)
- $A = [x_1-m, \ldots, x_M-m]$
- $A = U\Sigma V^T$
  - A: NxM
  - U: NxM $U^TU=I$
  - Σ: MxM diagonal
  - V: MxM $V^TV=VV^T=I$

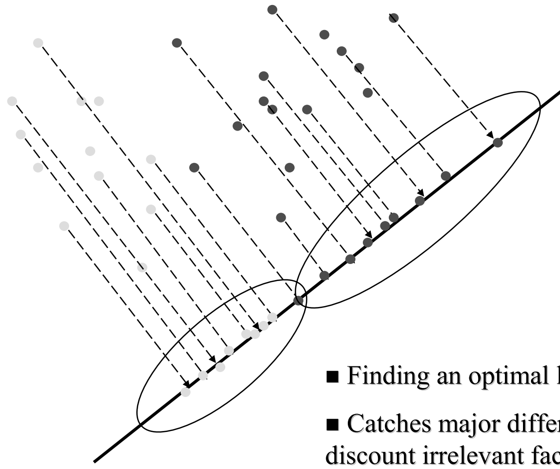
Fisher Linear Discrimination

- ✓ LDA
- ✓ PCA+LDA for Face Recognition

When does PCA fail?



Linear Discriminant Analysis



- Finding an optimal linear mapping W
- Catches major difference between classes and discount irrelevant factors
- In the mapped space, data are clustered

Within/between class scatters

$$m_1 = \frac{1}{n_1}\sum_{x\in D_1} x, \qquad m_2 = \frac{1}{n_2}\sum_{x\in D_2} x$$

the linear transform: $y = W^T x$

$$\tilde{m}_1 = \frac{1}{n_1}\sum_{x\in D_1} W^T x = W^T m_1, \qquad \tilde{m}_2 = W^T m_2$$

$$S_1 = \sum_{x\in D_1}(x-m_1)(x-m_1)^T, \qquad S_2 = \sum_{x\in D_2}(x-m_2)(x-m_2)^T$$

$$\tilde{S}_1 = \sum_{y\in Y_1}(y-\tilde{m}_1)^2 = W^T S_1 W, \qquad \tilde{S}_2 = \sum_{y\in Y_2}(y-\tilde{m}_2)^2 = W^T S_2 W$$

within class scatter: $S_W = S_1 + S_2$

between class scatter: $S_B = (m_1 - m_2)(m_1 - m_2)^T$

Fisher LDA

$$J(W)=\frac{|\tilde{m}_1-\tilde{m}_2|^2}{\tilde{S}_1+\tilde{S}_2}=\frac{W^T(m_1-m_2)(m_1-m_2)^T W}{W^T(S_1+S_2)W}=\frac{W^T S_B W}{W^T S_W W}$$

$$W^*=\arg\max_W J(W)$$

$$\max J(W) \quad \Leftrightarrow \quad S_B w=\lambda S_W w \leftarrow \text{this is a generalized eigenvalue problem}$$

Solution I

- If S_w is not singular

$$S_W^{-1}S_B w=\lambda w$$

- You can simply do eigenvalue decomposition on $S_W^{-1}S_B$

Solution II

- Noticing:
  - $S_B W$ is on the direction of m_1-m_2 (WHY?)
  - We are only concern about the direction of the projection, rather than the scale
- We have

$$w = S_W^{-1}(m_1 - m_2)$$

Multiple Discriminant Analysis

$$m_i = \frac{1}{n_i}\sum_{x \in D_i} x,$$

$$S_i = \sum_{x \in D_i}(x - m_i)(x - m_i)^T \qquad y = W^T x$$

$$S_W = \sum_{k=1}^{C} S_k$$

$$\tilde{m}_i = \frac{1}{n_i}\sum_{y \in Y_i} y = W^T m_i,$$

$$\tilde{m} = \frac{1}{n}\sum_{k=1}^{C} n_k \tilde{m}_k,$$

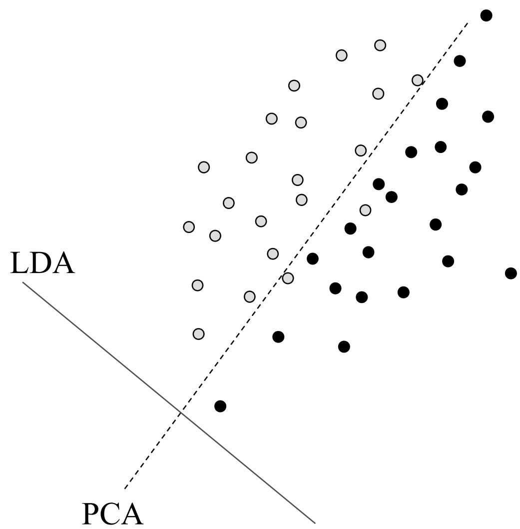
$$\tilde{S}_W = W^T S_W W,$$

$$\tilde{S}_B = \sum_{k=1}^{C} n_i(\tilde{m}_i - \tilde{m})(\tilde{m}_i - \tilde{m})^T = W^T S_B W$$

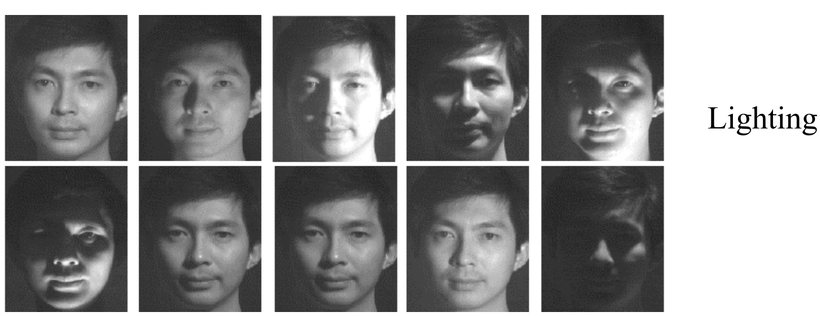
$$W^* = \arg\max_W \frac{|\tilde{S}_B|}{|\tilde{S}_W|} = \frac{|W^T S_B W|}{|W^T S_W W|}$$

$$\Longleftrightarrow \quad S_B w_i = \lambda S_W w_i$$

Comparing PCA and LDA



MDA for Face Recognition

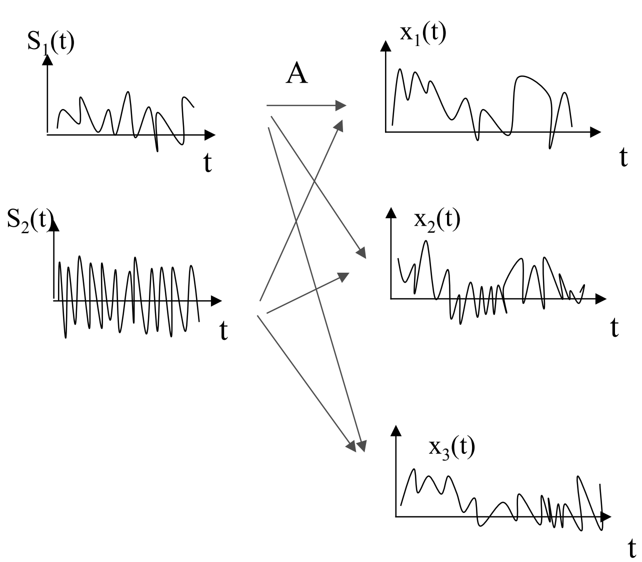


- PCA does not work well! (why?)
- solution: PCA+MDA

Independent Component Analysis

- ✓ The cross-talk problem
- ✓ ICA

Cocktail party



Can you recover $s_1(t)$ and $s_2(t)$ from $x_1(t)$, $x_2(t)$ and $x_3(t)$?

Formulation

$$x_j = a_{j1}s_1 + \ldots + a_{jn}s_n \quad \forall j$$

$$X = \sum_{i=1}^{n} a_i s_i \quad \text{or} \quad X = AS$$

Both A and S are unknowns!

Can you recover both A and S from X?

The Idea

$$Y = W^T X = W^T AS = Z^T S$$

- y is a linear combination of $\{s_i\}$
- Recall the central limit theory!
- A sum of even two independent r.v. is more Gaussian than the original r.v.
- So, $Z^T S$ should be more Gaussian than any of $\{s_i\}$
- In other words, $Z^T S$ become least Gaussian when in fact equal to $\{s_i\}$
- Amazed!

Face Recognition: Challenges

View