

Linear Models: PCA, FA

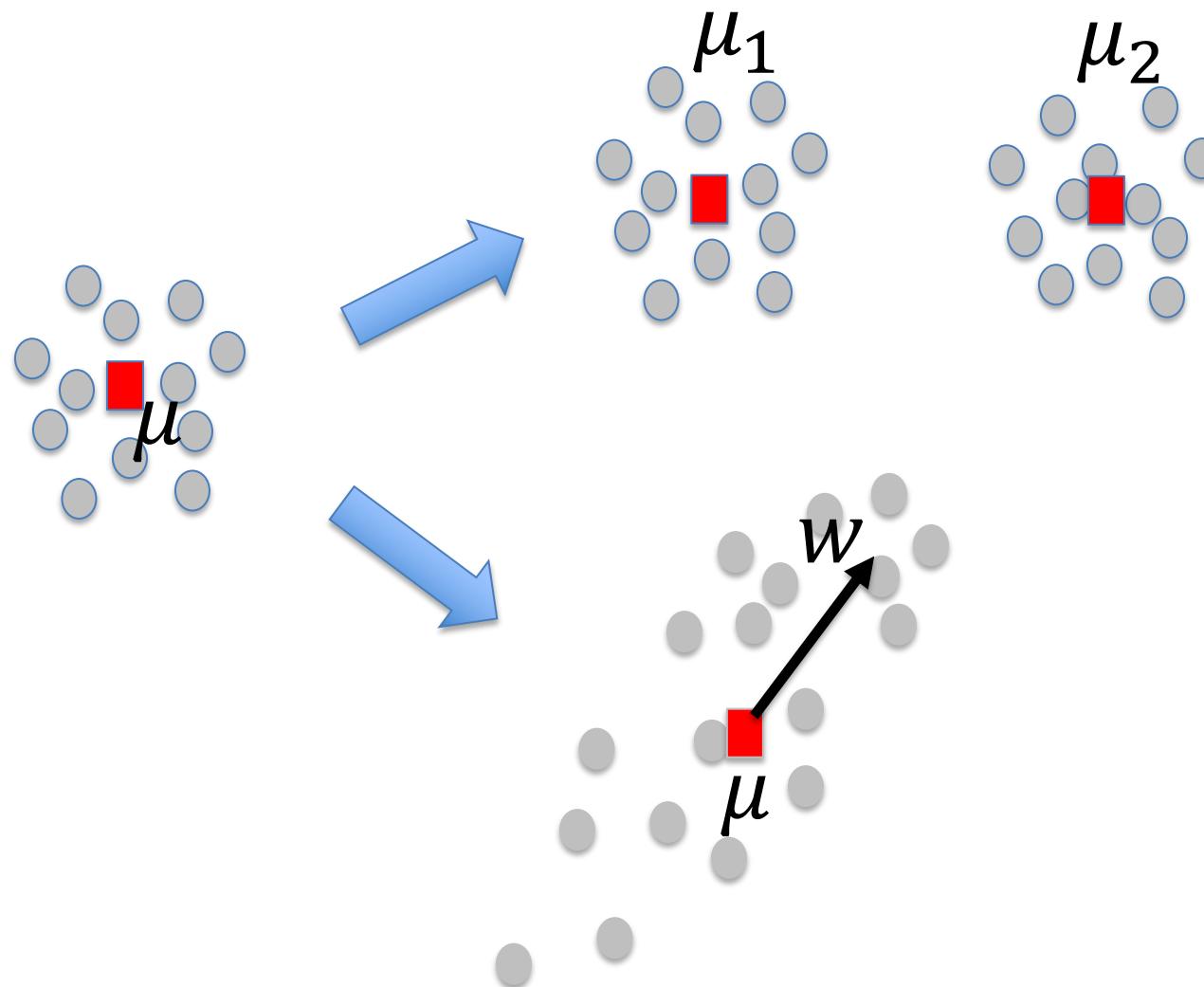
Shikui Tu

2018-03-22

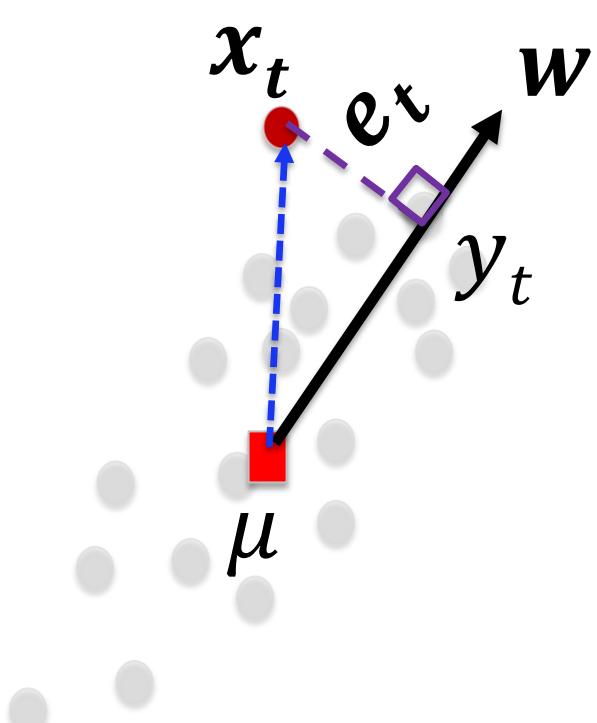
Outline

- Principal Component Analysis (PCA)
- From matrix factorization perspectives
- Hebbian learning, LMSER and PCA
- Probabilistic PCA, Factor Analysis (FA)

Model from “one point” to “one line”



Define the error



$$||w|| = 1$$

$$y_t = x_t^T w$$

$$e_t = ||x_t - y_t w||^2$$

$$J(w) = \frac{1}{N} \sum_{t=1}^N ||x_t - (x_t^T w)w||^2$$

Mean Square Error (MSE)

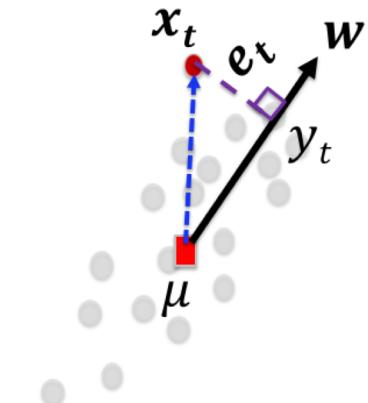
$$J(\mathbf{w}) = \frac{1}{N} \sum_{t=1}^N \|\mathbf{x}_t - (\mathbf{x}_t^T \mathbf{w}) \mathbf{w}\|^2$$

$$\begin{aligned} & \mathbf{x}_t^T \mathbf{x}_t - (\mathbf{x}_t^T \mathbf{w}) \mathbf{w}^T \mathbf{x}_t - \mathbf{x}_t^T (\mathbf{x}_t^T \mathbf{w}) \mathbf{w} + (\mathbf{x}_t^T \mathbf{w}) \mathbf{w}^T (\mathbf{x}_t^T \mathbf{w}) \mathbf{w} \\ &= \mathbf{x}_t^T \mathbf{x}_t - \mathbf{w}^T (\mathbf{x}_t \mathbf{x}_t^T) \mathbf{w} \end{aligned}$$

Introduce a Lagrange multiplier λ

$$L(\{\mathbf{x}_t\}, \mathbf{w}) = J(\{\mathbf{x}_t\}, \mathbf{w}) - \lambda \cdot (\mathbf{w}^T \mathbf{w} - 1)$$

$$\frac{\partial J(\mathbf{w})}{\partial \mathbf{w}} - \lambda \cdot \frac{\partial (\mathbf{w}^T \mathbf{w} - 1)}{\partial \mathbf{w}} = -2(\Sigma_x \mathbf{w}) - \lambda \cdot 2\mathbf{w} = \mathbf{0}$$



$$\|\mathbf{w}\| = 1$$

$$\Sigma_x = \frac{1}{N} \sum_{t=1}^N \mathbf{x}_t \mathbf{x}_t^T$$

$$\Sigma_x \mathbf{w} = (-\lambda) \cdot \mathbf{w}$$

Eigenvalues and Eigenvectors

Lagrange multiplier

From wiki

maximize $f(x, y)$
subject to $g(x, y) = 0$

Lagrange multiplier λ

$$\mathcal{L}(x, y, \lambda) = f(x, y) - \lambda \cdot g(x, y)$$

$$\nabla_{x,y} f = \lambda \nabla_{x,y} g$$

$$\nabla_{x,y} f = \left(\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y} \right)$$

$$\nabla_{x,y} g = \left(\frac{\partial g}{\partial x}, \frac{\partial g}{\partial y} \right)$$

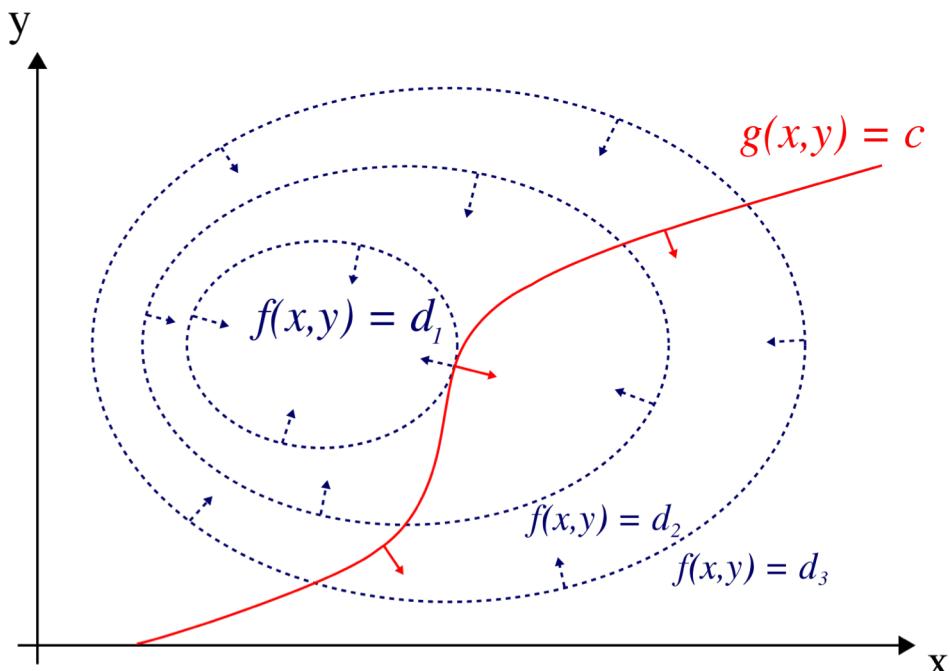
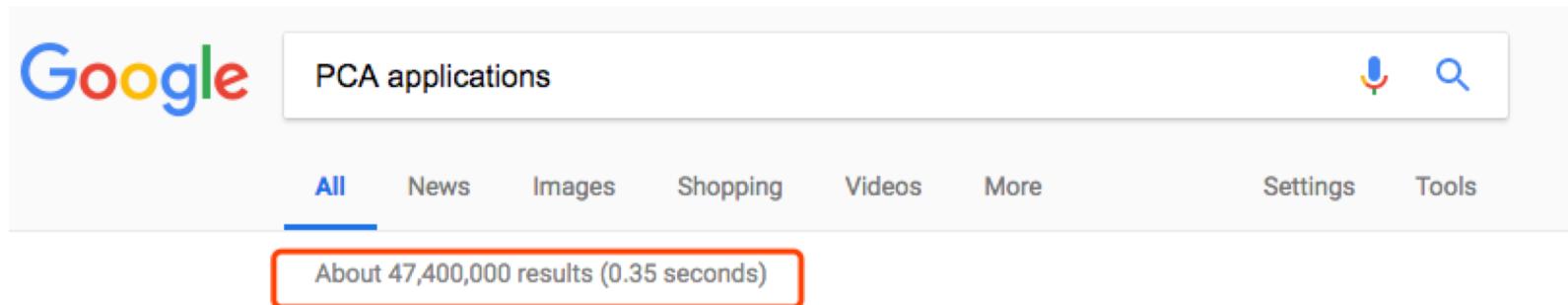


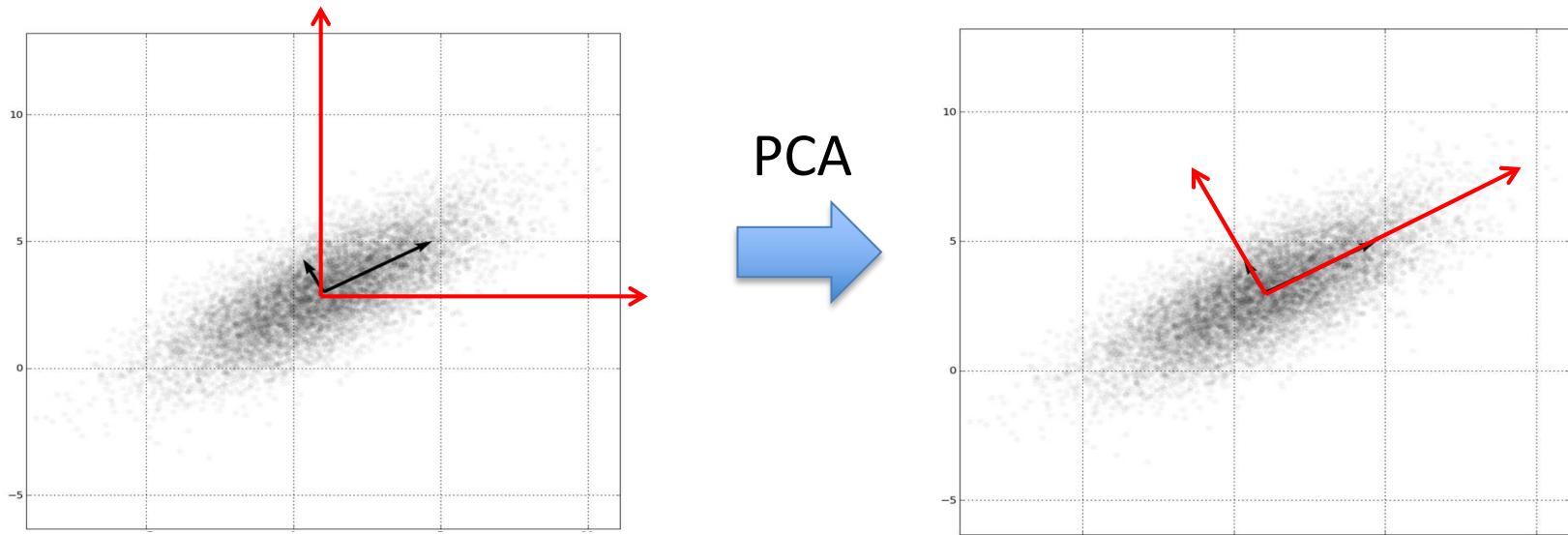
Figure 1: The red line shows the constraint $g(x, y) = c$. The blue lines are contours of $f(x, y)$. The point where the red line tangentially touches a blue contour is the maximum of $f(x, y)$, since $d_1 > d_2$.

Applications of PCA

- Popular in multivariate statistics, signal processing
- Reduce data noise, redundancy, correlation, ...
- Dimensionality reduction
- Feature selection and feature extraction
- Data visualization (to 2D or 3D)

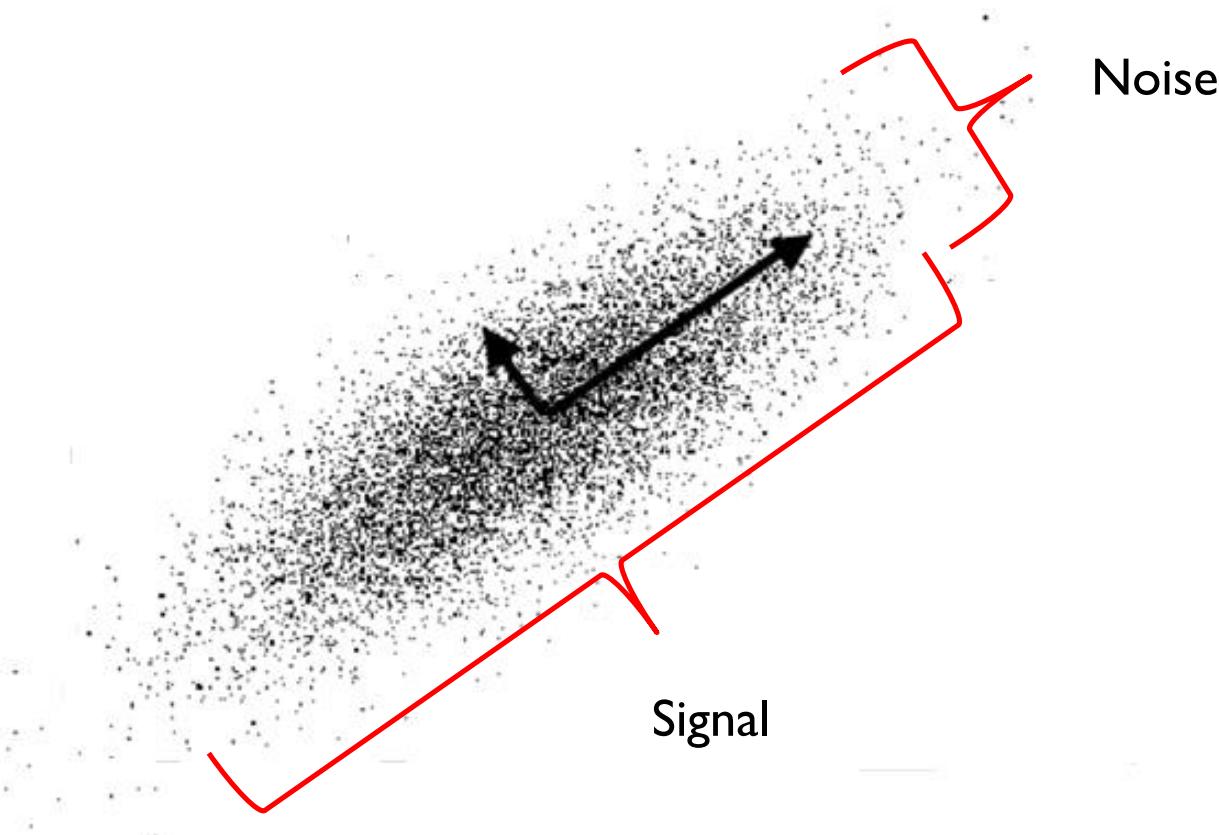


Data correlation



Correlation can be removed by rotating the coordinates to principal components.

Signal-noise ratio maximization



Keep one signal dimension, discard one noisy dimension.

Information redundancy

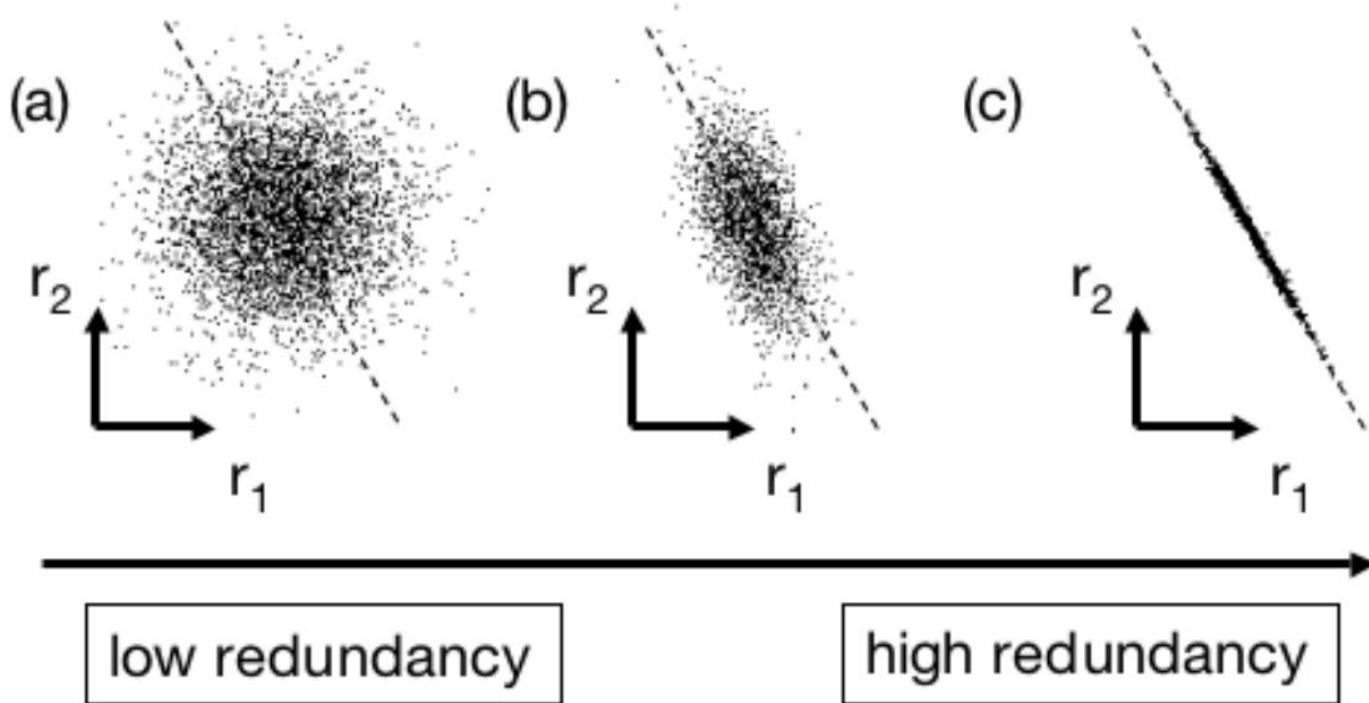


Image denoising by PCA



(a) Noisy image



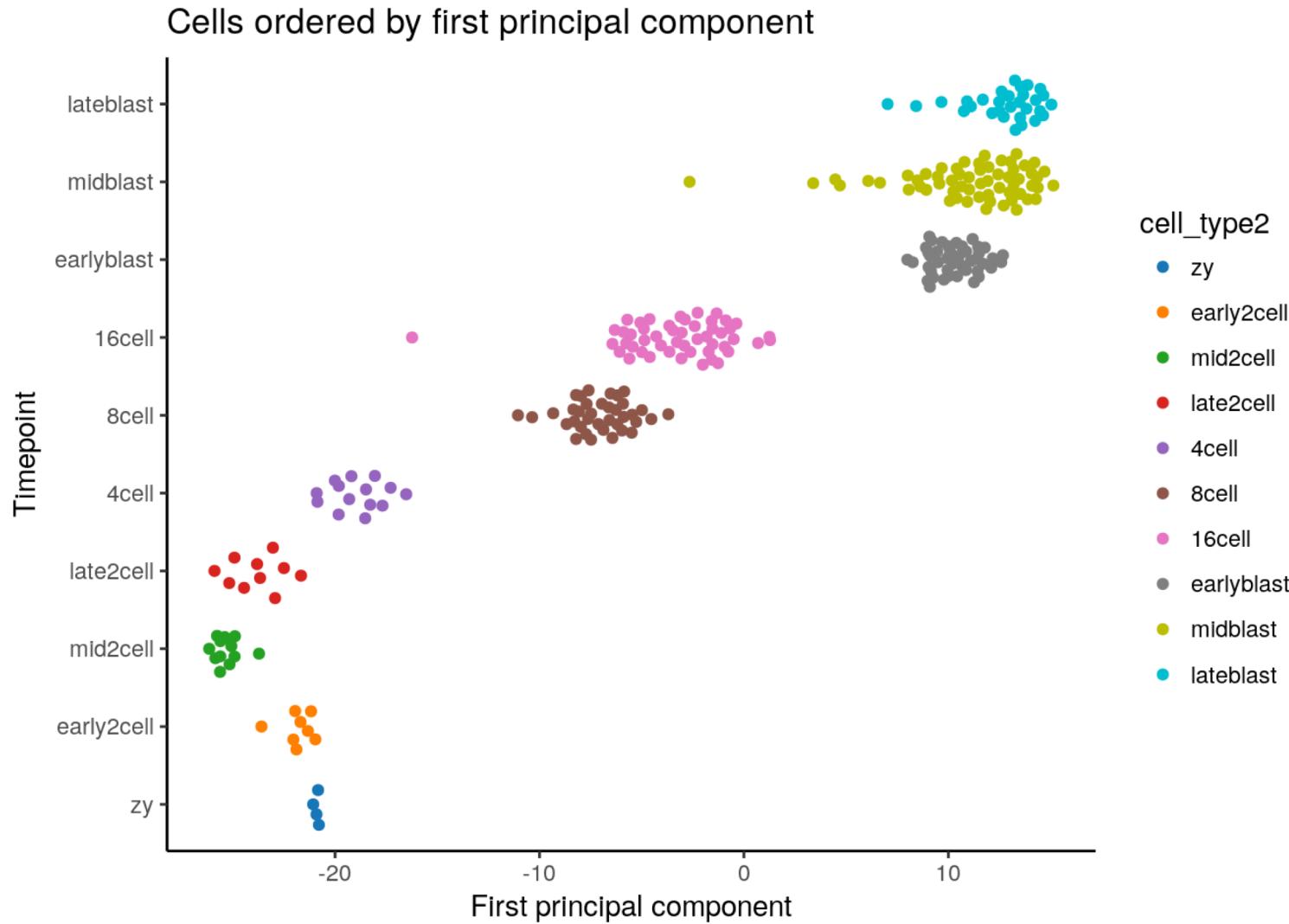
(b) PGPCA (PSNR=33.6)



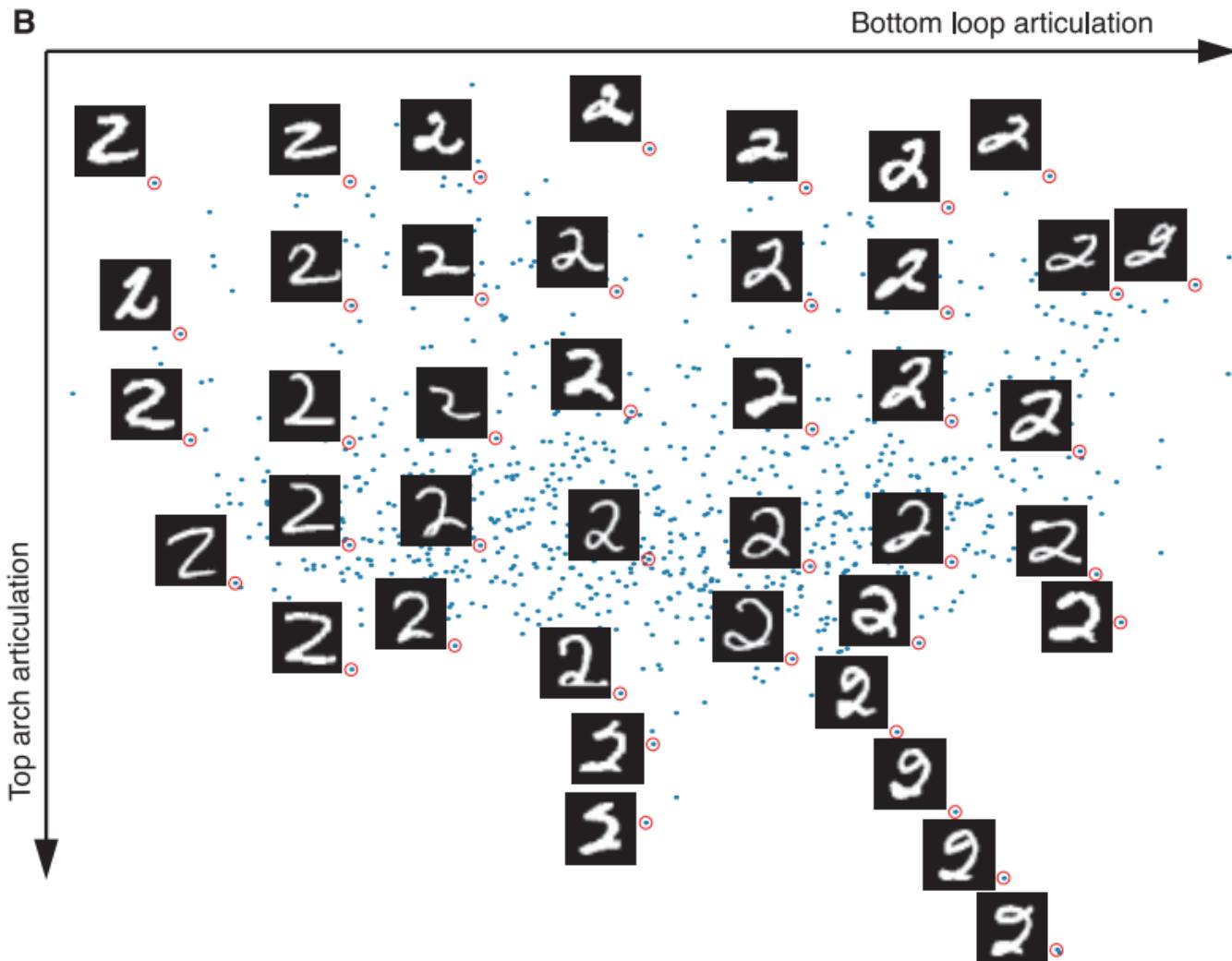
(c) PLPCA (PSNR=34.8)

Figure 6: Visual evaluation of the denoising performance of PGPCA and PLPCA on an image (Barbara) damaged by an AWGN with noise level $\sigma = 10$, with their PSNR.

Order the cells by PCA



Visualize the hand-written digits



PCA of hand-written digits

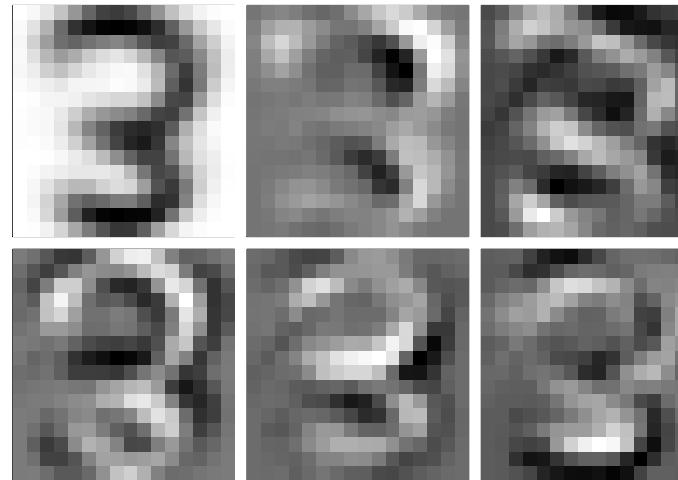
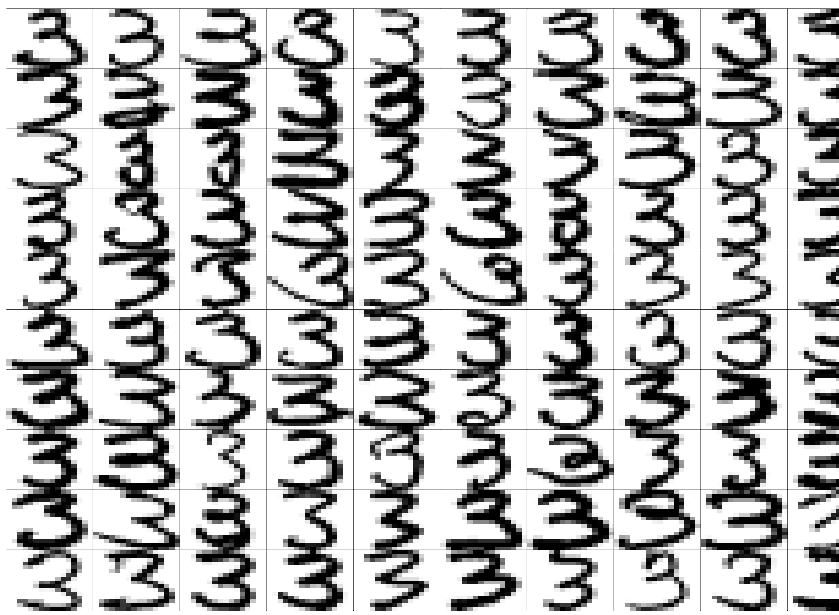


Figure 4: Digits: mean and eigenvectors



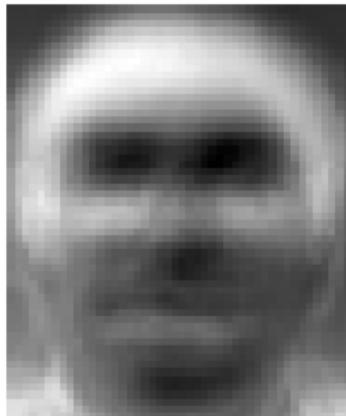
Figure 5: Digits data: Top: digits. Bottom: their reconstructions.

Eigen-face method

- Sirovich and Kirby (1987) showed that PCA could be used on a collection of face images to form a set of basis features.
 - Given input image vector $U \in \Re^n$, the mean image vector from the database M , calculate the weight of the k th eigenface as:
$$w_k = V_k^T(U - M)$$
Then form a weight vector $W = [w_1, w_2, \dots, w_k, \dots, w_n]$
 - Compare W with weight vectors W_m of images in the database. Find the Euclidean distance.
$$d = ||W - W_m||^2$$
 - If $d < \epsilon_1$, then the m th entry in the database is a candidate of recognition.
 - If $\epsilon_1 < d < \epsilon_2$, then U may be an unknown face and can be added to the database.
 - If $d > \epsilon_2$, U is not a face image.

Eigen-face examples

Eigen-face



Reconstruction by top-k eigenvectors

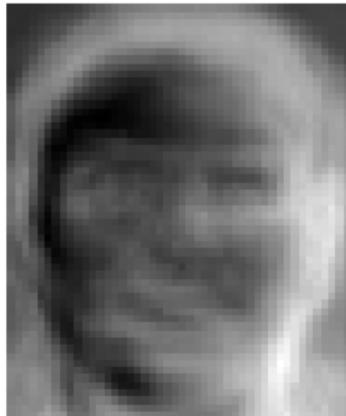
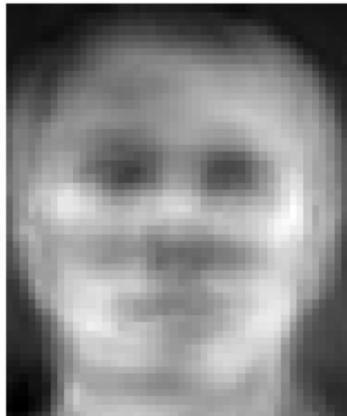
k=2



k=15



k=40



Some eigenfaces from AT&T
Laboratories Cambridge

Outline

- Principal Component Analysis (PCA)
- From matrix factorization perspectives
- Hebbian learning, LMSER and PCA
- Probabilistic PCA, Factor Analysis (FA)

From matrix factorization perspectives

- Eigen-decomposition

$$\Sigma_x \approx U\Lambda U^T$$

$$U = [\mathbf{u}_1, \dots, \mathbf{u}_k]_{d \times N}^T$$

$$\Lambda = \text{diag}[\lambda_1, \dots, \lambda_k]$$

Covariance matrix (assume x_t zero mean):

$$\Sigma_x = \frac{1}{N} \sum_{t=1}^N \mathbf{x}_t \mathbf{x}_t^T = \frac{1}{N} X X^T$$

$$X = [\mathbf{x}_1, \dots, \mathbf{x}_N]_{d \times N}$$

- Singular Value Decomposition (SVD)

$$X = UDV^T$$

$$X X^T = UDV^T \cdot VDU^T = U D^2 U^T$$

PCA and SVD

Data: n points in p -dim: $X = (x_1, x_2, \dots, x_n)$

Covariance $C = XX^T = \sum_{k=1}^p \lambda_k u_k u_k^T$

Gram (kernel) matrix $X^T X = \sum_{k=1}^r \lambda_k v_k v_k^T$

Principal directions: u_k
(Principal axis, subspace)

Principal components: v_k
(projection on the subspace)

Underlying basis: SVD $X = \sum_{k=1}^p \sigma_k u_k v_k^T = U \Sigma V^T$

Methods of PCA utilization

Principal components
(uncorrelated random variables):

$$X = (x_1, x_2, \dots, x_n)$$

$$u_k = u_k(1) \cdot X_1 + \dots + u_k(d) \cdot X_d$$

Dimension reduction: $X = \sum_{k=1}^p \sigma_k u_k v_k^T = U \Sigma V^T$

Projection to low-dim
subspace

$$\tilde{X} = U^T X \quad U = (u_1, \dots, u_k)$$

Sphereing the data
Transform data to $N(0,1)$

$$\tilde{X} = C^{-1/2} X = U \Sigma^{-1} U^T X$$

From PCA to spectral clustering

Consider the kernel matrix: $W_{ij} = \langle \phi(x_i), \phi(x_j) \rangle$

In **Kernel PCA** we compute eigenvector: $Wv = \lambda v$

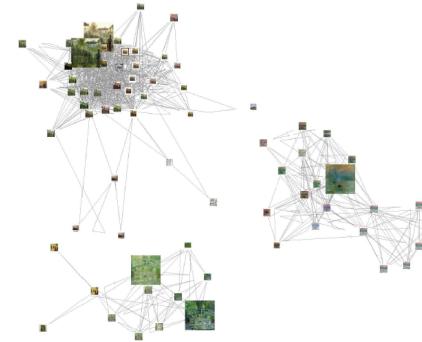
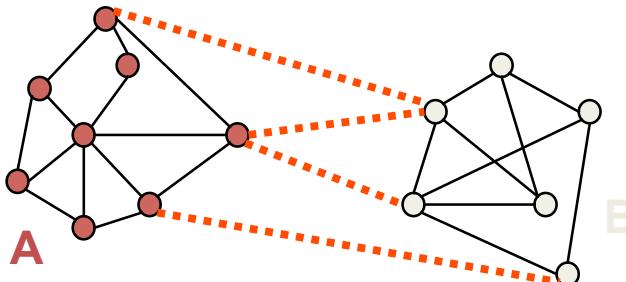
Generalized Eigenvector: $Wq = \lambda Dq$

$$D = \text{diag}(d_1, \dots, d_n) \quad d_i = \sum_j w_{ij}$$

This leads to **Spectral Clustering** !

Spectral clustering

Group data points based on links in a graph:



Graph Laplacian matrix L :

$$L = D - W.$$

$$L_{\text{sym}} := D^{-1/2} L D^{-1/2} = I - D^{-1/2} W D^{-1/2}$$

$$L_{\text{rw}} := D^{-1} L = I - D^{-1} W.$$

Unnormalized spectral clustering

Input: Similarity matrix $S \in \mathbb{R}^{n \times n}$, number k of clusters to construct.

- Construct a similarity graph by one of the ways described in Section 2. Let W be its weighted adjacency matrix.
- Compute the unnormalized Laplacian L .
- **Compute the first k eigenvectors u_1, \dots, u_k of L .**
- Let $U \in \mathbb{R}^{n \times k}$ be the matrix containing the vectors u_1, \dots, u_k as columns.
- For $i = 1, \dots, n$, let $y_i \in \mathbb{R}^k$ be the vector corresponding to the i -th row of U .
- Cluster the points $(y_i)_{i=1, \dots, n}$ in \mathbb{R}^k with the k -means algorithm into clusters C_1, \dots, C_k .

Output: Clusters A_1, \dots, A_k with $A_i = \{j \mid y_j \in C_i\}$.

Scaled PCA for spectral clustering

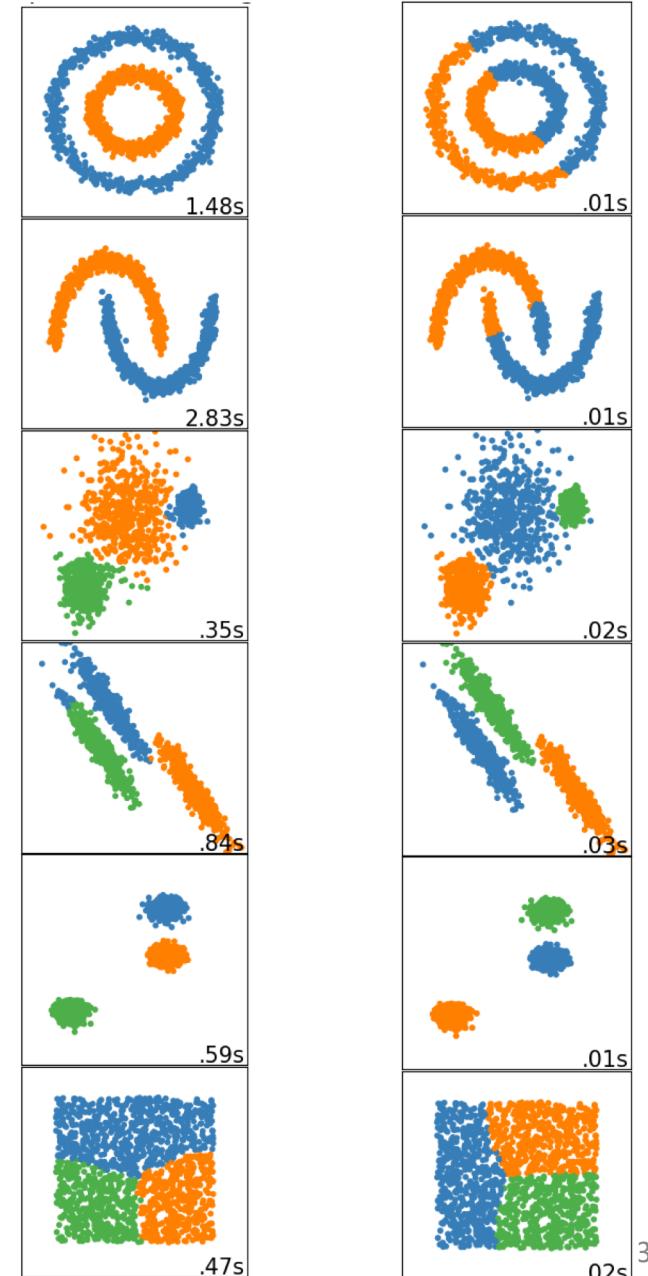
PCA: $W = \sum_k v_k \lambda_k v_k^T$

Scaled PCA: $W = D^{\frac{1}{2}} \tilde{W} D^{\frac{1}{2}} = D \sum_{k=1} q_k \lambda_k q_k^T D$

$$\tilde{W} = D^{\frac{1}{2}} W D^{\frac{1}{2}}, \quad \tilde{w}_{ij} = w_{ij} / (d_i d_j)^{1/2}$$

$q_k = D^{-\frac{1}{2}} v_k$ scaled principal component

Spectral clustering GMM clustering



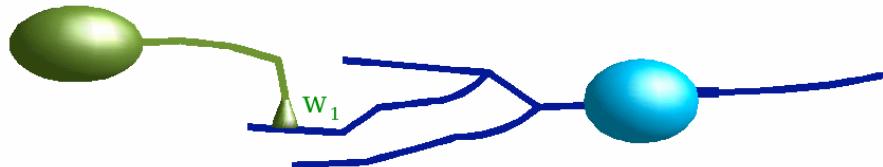
Outline

- Principal Component Analysis (PCA)
- From matrix factorization perspectives
- Hebbian learning, LMSER and PCA
- Probabilistic PCA, Factor Analysis (FA)

Hebbian learning



- Donald Hebb wrote in 1949:
 - When an axon in cell A is near enough to excite cell B and repeatedly and persistently takes part in firing it, some growth process or metabolic change takes place in one or both cells such that A's efficiency in firing B is increased.
- The Hebbian synapse



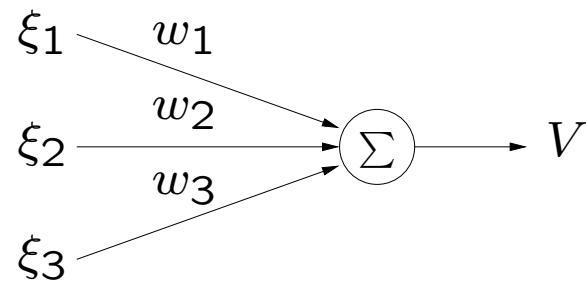
$$\Delta w_1(t) \propto x(t) y(t)$$

The Hebbian Neuron

A computational system which implements Hebbian learning.

Let's assume a linear unit; experiment shows this is largely sufficient:

$$V = \sum_j w_j \xi_j = \bar{w}^T \bar{\xi} \quad (2)$$



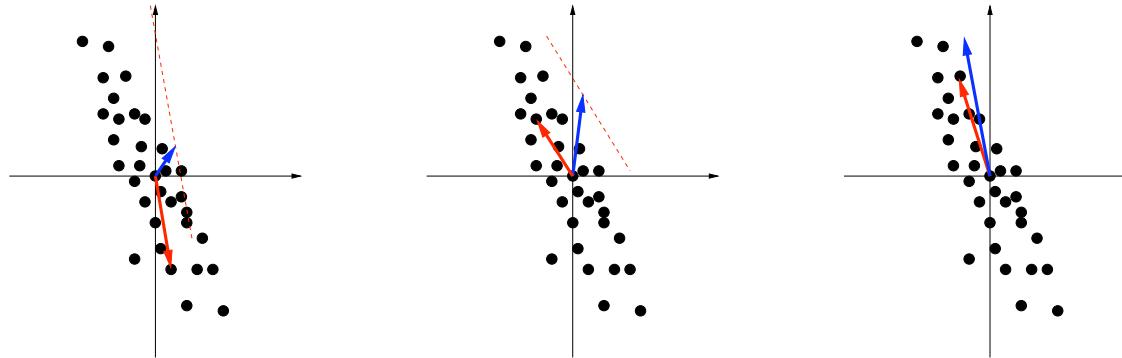
Plain Hebbian learning:

$$\Delta \bar{w} = \eta V \bar{\xi} \quad (3)$$

A adaptive learning rule

Hebbian learning implements PCA

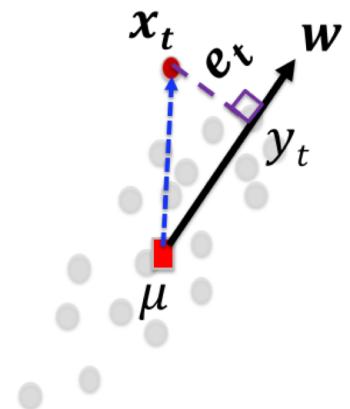
Recall that $\Delta \bar{w} = \eta V \bar{\xi}$ and that $V = \bar{w}^T \bar{\xi}$.



In this case, the weight vector will ultimately align itself with the *direction of greatest variance* in the data.

$$J(w) = \frac{1}{N} \sum_{t=1}^N \|x_t - (x_t^T w)w\|^2$$

$$\begin{aligned} & x_t^T x_t - (x_t^T w) w^T x_t - x_t^T (x_t^T w) w + (x_t^T w) w^T (x_t^T w) w \\ &= x_t^T x_t - w^T (x_t x_t^T) w \end{aligned}$$



LMSER for PCA

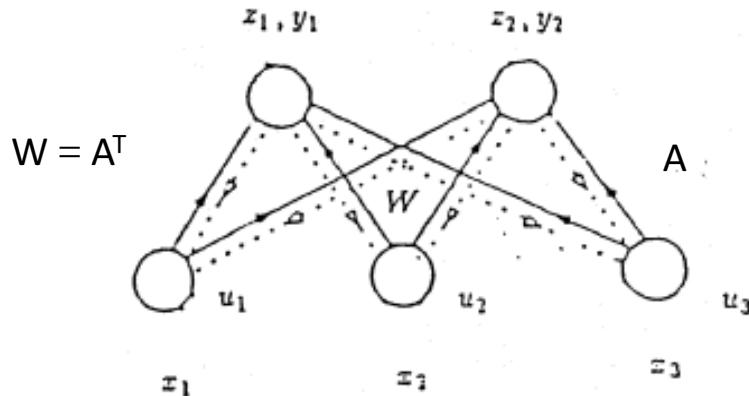
Least Mean Square Error Reconstruction (LMSER)

Xu, 1991, 93 Proc. IJCNN 91, Singapore, pp. 2362-2373
Neural Networks, vol. 6, pp. 627-648, 1993

$$s(\bar{y}) = [s(\bar{y}^{(1)}), \dots, s(\bar{y}^{(m)})]^T$$

$$s(r) = \frac{1}{1+e^{-r}}, \quad \bar{y} = W(x - \mu)$$

$$J(W) = \frac{1}{N} \sum_{t=1}^N ||x_t - W^T W x_t||^2$$



$$\vec{y} = W\vec{x}, \vec{u} = W^T \vec{y}, \vec{y}^r = W\vec{u}$$

$$\bar{z} = S(\bar{y}) = A_m \bar{y}, A_m = \text{diag}[a_1, \dots, a_{n_1}]$$

$a_1 > \dots > a_{n_1}$ are all positive.

$$\tau^W \frac{dW}{dt} = \bar{z}\bar{x}^t - \bar{y}\bar{u}^t$$

$$\tau^W \frac{dW}{dt} = \bar{z}\bar{x}^t - \bar{y}\bar{u}^t + \bar{z}\bar{x}^t - \bar{y}^t\bar{x}^t$$

LMSER implements PCA/PSA

THEOREM 3. Assume $\lambda_1 \geq \lambda_2 \dots \lambda_{n_1} > \lambda_{n_1+1} \geq \lambda_{n_0} > 0$ and $\Phi = [\vec{\phi}_1, \dots, \vec{\phi}_{n_0}]$. Then all the critical points of eqn (10b) given by Theorem 2 are saddle points of the energy landscape of J , except those with $W = R'\Phi'^t$,

$R'^t R' = I$, $\Phi'^t \Phi' = I$, where $\Phi' = D\Phi'$, and $D = [D_1 | \vec{0}]$ with D_1 being diagonal matrix and its diagonal elements being either +1 or -1. Furthermore, these $W = R'\Phi'^t$ let $J = \frac{1}{2}E(\|\vec{x} - \vec{u}\|^2) = \frac{1}{2}E(\|\vec{x} - W^t W \vec{x}\|^2)$ reach its only local (also global) minimum $J_{min} = \frac{1}{2} \sum_{i=n_1+1}^{n_0} \lambda_i$.

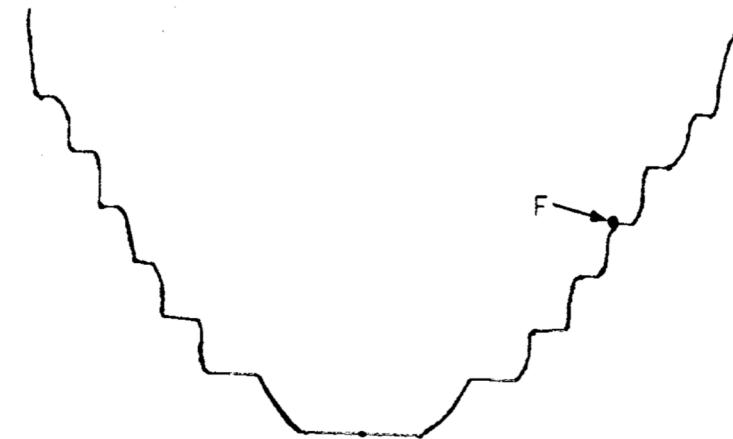


FIGURE 3. The landscape of J . There is only one unique minimum which is the flat bottom, and there are in total $\sum_{i=1}^{n_0} C'_{n_0} - 1$ plateaux which are the saddle points.

Outline

- Principal Component Analysis (PCA)
- From matrix factorization perspectives
- Hebbian learning, LMSER and PCA
- Probabilistic PCA, Factor Analysis (FA)

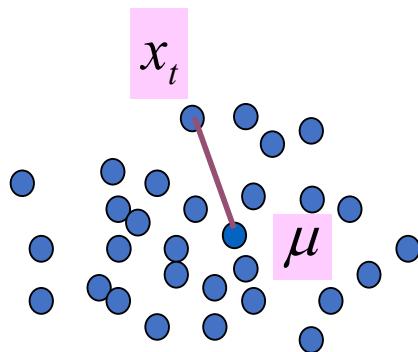
平方误差与高斯分布

$$e_t = x_t - \mu, \quad (x_t - \mu)^2 \Leftrightarrow p(x_t, \theta) \propto e^{-c(x_t - \mu)^2}, c > 0 \text{ Gaussian}$$

$$G(x | \mu, \sigma^2) = \frac{1}{\sqrt{2\pi}\sigma} \exp\left\{-\frac{(x - \mu)^2}{2\sigma^2}\right\}$$

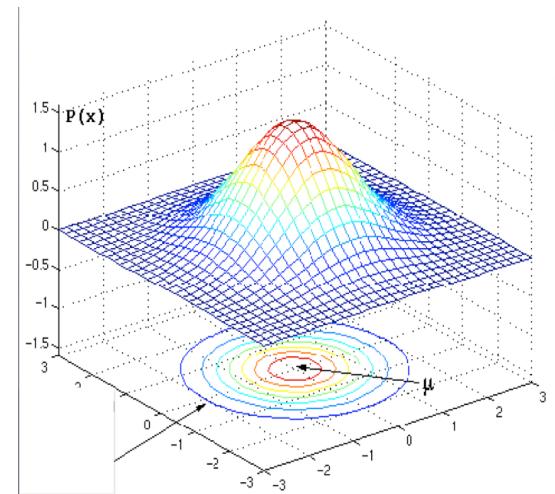
where σ^2 is the variance, and μ is the mean value.

$$e_t = x_t - \mu, \quad \|x_t - \mu\|^2 \Leftrightarrow p(x_t, \theta) \propto e^{-c\|x_t - \mu\|^2}, c > 0 \text{ Gaussian}$$



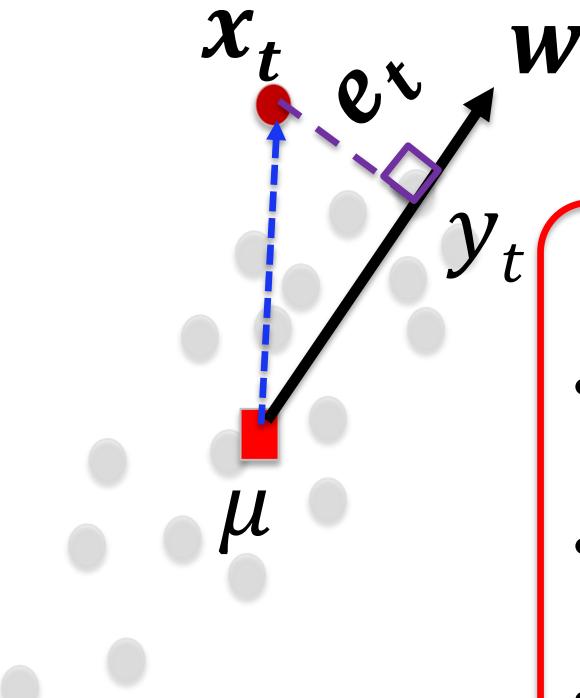
$$G(\mathbf{x} | \boldsymbol{\mu}, \sigma^2) = \frac{1}{(2\pi)^{d/2} \sigma^d} \exp\left\{-\frac{1}{2\sigma^2} (\mathbf{x} - \boldsymbol{\mu})^T (\mathbf{x} - \boldsymbol{\mu})\right\}$$

where σ^2 is the variance, and $\boldsymbol{\mu}$ is the mean vector.
 d is the dimension.



Generative model perspective

Continuous latent variable y



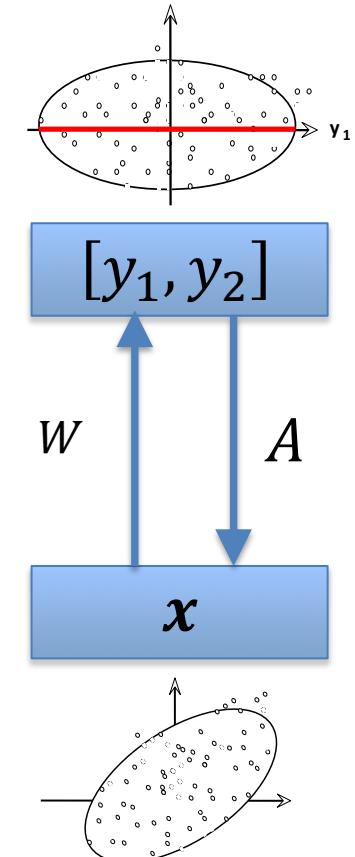
$$||w|| = 1$$

$$y_t = x_t^T w$$

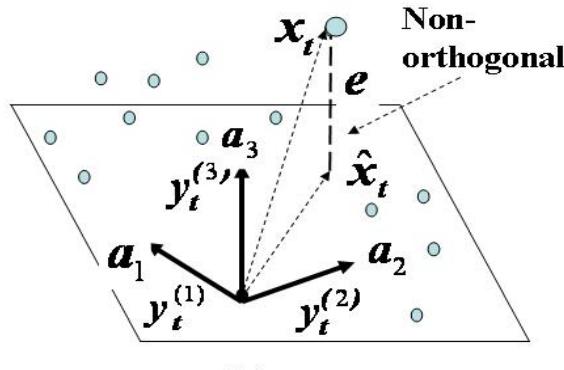
$$e_t = ||x_t - y_t w||^2$$

For the t -th data point:

- Randomly sample a y_t :
 $y_t \sim G(y|\mathbf{0}, \Sigma_y);$
- Randomly generate a noise e_t
 $e_t \sim G(e|0, \sigma^2 I)$
- Generate x_t by:
$$x_t = Ay_t + \mu + e_t$$



Factor Analysis (FA) Model



$A^T A = I$ has been removed because it impedes $\sum_t \|e_t\|^2$ to reach its minimum

Indeterminacy

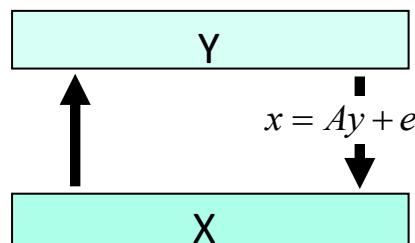
- e) a rotation matrix since $A' = \Phi A$ spans the same subspace;
- f) a diagonal D with $A' = AD$,

$$y' = D^{-1}y.$$

Two Choices

$$q(y) = \begin{cases} G(y|0, I), & \text{choice (a)} \\ G(y|0, \Lambda), & \text{choice (b)} \end{cases}$$

$$q(y) = \prod_{j=1}^k G(y_j | 0, 1) = G(y | 0, I)$$



- g) a unknown allocation between the two additive terms $Exx^T = A^T \Lambda A + Eee^T$.

- 样本方差之分割不变性 (样本方差在子空间内外可任意分割)
- 超阈维数不变性 (高于某维的子空间以零误差描述有限样本集)

EM algorithm for FA

E-Step: $p^{old}(\mathbf{y}|\mathbf{x}) = \frac{G(\mathbf{y}|0, I)G(\mathbf{x}|A\mathbf{y} + \boldsymbol{\mu}, \sigma^2 I)}{G(\mathbf{x}|\boldsymbol{\mu}, AA^T + \sigma^2 I)}$

$$E[\mathbf{y}|\mathbf{x}] = W\mathbf{x} \quad W = A^T(AA^T + \sigma^2 I)^{-1}$$

$$E[\mathbf{y}\mathbf{y}^T|\mathbf{x}] = I - WA + W\mathbf{x}\mathbf{x}^TW^T$$

M-Step: $\max Q(p^{old}(\mathbf{y}|\mathbf{x}), \Theta)$

$$Q = \int p^{old}(\mathbf{y}|\mathbf{x}) \cdot \ln[G(\mathbf{y}|0, I)G(\mathbf{x}|A\mathbf{y} + \boldsymbol{\mu}, \sigma^2 I)] d\mathbf{y}$$

$$A^{new} = \left(\sum_{t=1}^N \mathbf{x}_t (E[\mathbf{y}|\mathbf{x}_t])^T \right) \left(\sum_{t=1}^N E[\mathbf{y}\mathbf{y}^T|\mathbf{x}_t] \right)^{-1}$$

$$\sigma^2^{new} = \frac{1}{Nd} Tr \left\{ \sum_{t=1}^N \{ \mathbf{x}_t \mathbf{x}_t^T - A^{new} E[\mathbf{y}|\mathbf{x}_t] \mathbf{x}_t^T \} \right\}$$

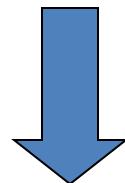
Maximum likelihood FA implements PCA

$$p(\mathbf{y}) = G(\mathbf{y}|\mathbf{0}, I), \quad p(\mathbf{x}|\mathbf{y}) = G(\mathbf{x}|\mathbf{A}\mathbf{y} + \boldsymbol{\mu}, \Sigma_e),$$

$$p(\mathbf{x}|\Theta) = \int p(\mathbf{y})p(\mathbf{x}|\mathbf{y})d\mathbf{y} = G(\mathbf{x}|\boldsymbol{\mu}, \mathbf{A}\mathbf{A}^T + \Sigma_e),$$

$$\max_{\Theta} \log \left\{ \prod_{t=1}^{N=1} p(\mathbf{x}_t | \Theta) \right\}$$

Maximum Likelihood



$$\Sigma_e = \sigma_e^2 \mathbf{I}_n$$

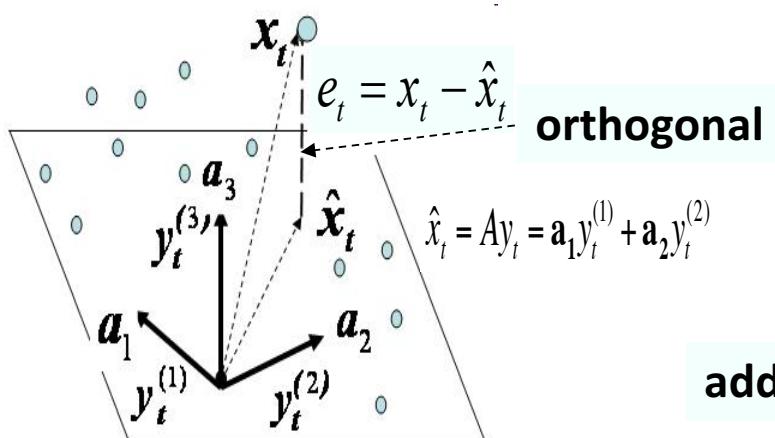
assume $\boldsymbol{\mu} = \mathbf{0}$

PCA

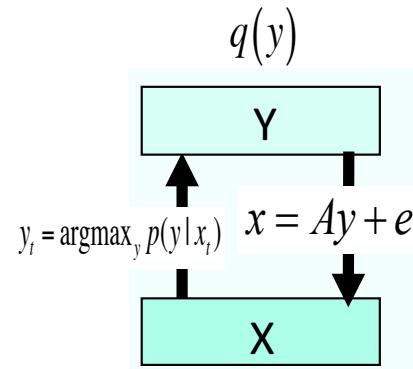
$$\begin{cases} \hat{\mathbf{A}}_{n \times m}^{ML} = \mathbf{U}_{n \times m} (\mathbf{D}_m - \hat{\sigma}_e^2)^{\frac{1}{2}} \mathbf{R}^T, & \mathbf{D}_m = \text{diag}[s_1, \dots, s_m] \\ \hat{\sigma}_e^{2,ML} = \frac{1}{n-m} \sum_{i=m+1}^n s_i, \end{cases}$$

\mathbf{U} is eigenvectors of sample cov.

Reparameterization of FA



$$y_t = [y_t^{(1)}, y_t^{(2)}]^T$$



$$\max_{\theta} \sum_t \ln q(x_t | \theta)$$

- a) e and y are not correlated
- b) $\sum_t \|e_t\|^2$ reaches minimum
- c) e is a Gaussian noise $G(e|0, \sigma^2 I)$
- d) a unique plane but A not

add extra constraint : I

$$q(y) = G(y|\nu, \Lambda)$$

$\Lambda = \text{diag}[\lambda_1, \dots, \lambda_m]$ denotes a diagonal matrix with diagonal elements $\lambda_1, \dots, \lambda_m$.

$p(y|x) = q(y|x)$
 $q(y|x) = \frac{G(x|Ay + \mu, \Sigma)G(y|\nu, \Lambda)}{G(x|\mu, A\Lambda A^T + \Sigma)}$

$x = Ay + e$
 $Eye^T = 0$
 $y = W(x - \mu) + \varepsilon$

$q(x|y) = G(e|\mu, \Sigma)$
 $G(x|\mu, A\Lambda A^T + \Sigma) = \int G(x|Ay + \mu, \Sigma)G(y|\nu, \Lambda)dy$

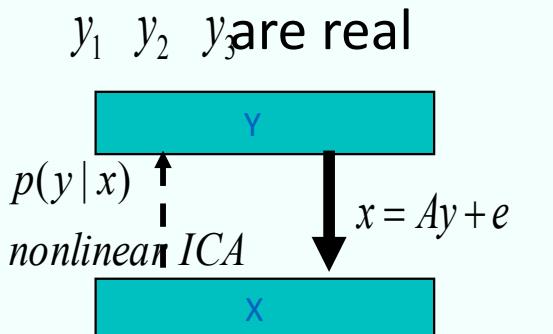
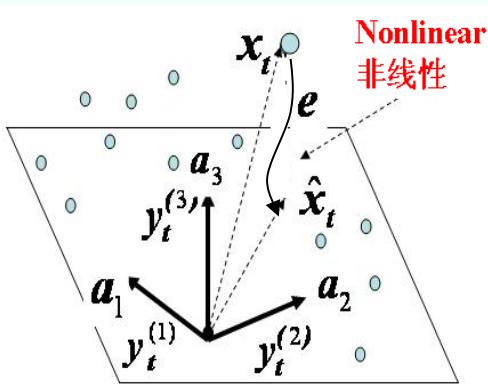
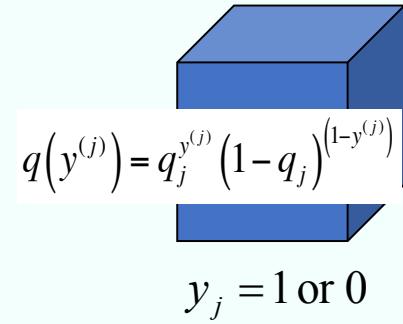
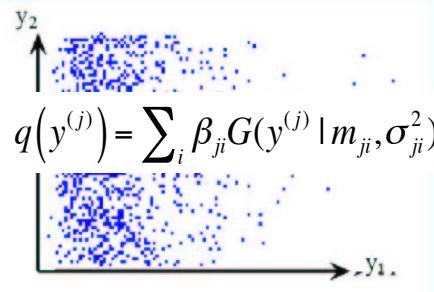
Two parameterizations of FA

	TYPE-A FA-A: $\Theta_m^a = \{A, \mu, \Sigma_e\}$	TYPE-B FA-B: $\Theta_m^b = \{U, \mu, \Lambda, \Sigma_e\}$
$E[ye^T]$	0 (y and e uncorrelated)	0 (y and e uncorrelated)
$q(y \Theta)$	$G(y 0, I_m)$	$G(y 0, \Lambda)$, $\Lambda = diag[\lambda_1, \dots, \lambda_m]$
A	any full column rank matrix	$A = U$, $U^T U = I_m$
$q(x y, \Theta)$	$G(x Ay + \mu, \Sigma_e)$	$G(x Uy + \mu, \Sigma_e)$
$q(x \Theta)$	$G(x \mu, AA^T + \Sigma_e)$	$G(x \mu, U\Lambda U^T + \Sigma_e)$

Which one is better?

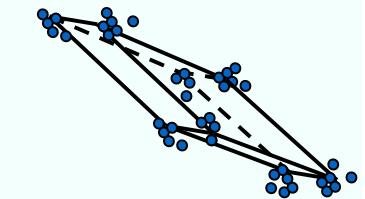
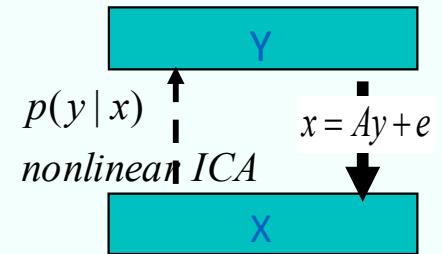
Linear non-Gaussian

$$q(y) = \prod_{j=1}^k q(y^{(j)})$$



$x = y_2 a_2 + y_1 a_1 + e$

NFA



BFA

Thank you!