Linear Models: PCA, FA

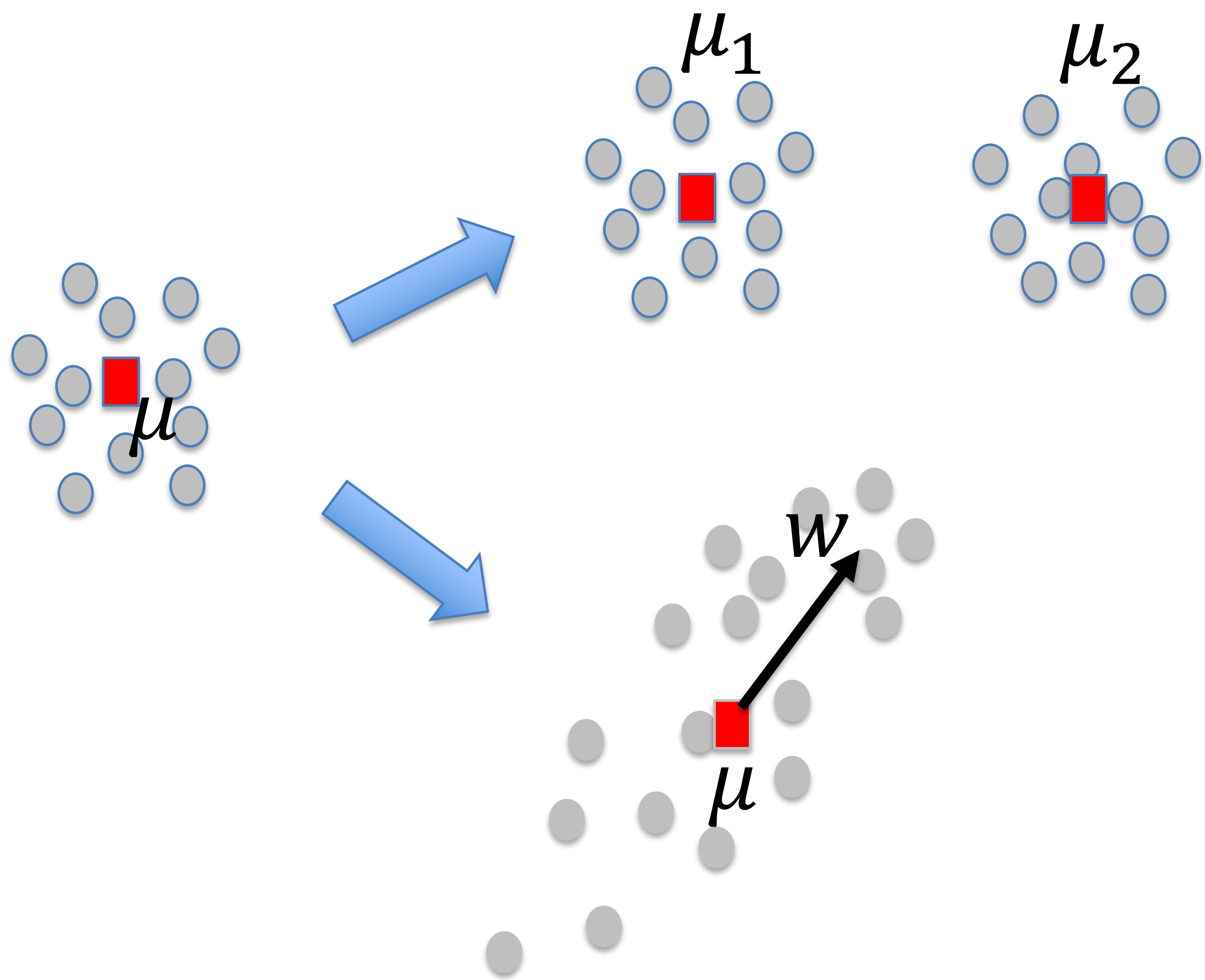
Shikui Tu

2019-03-19

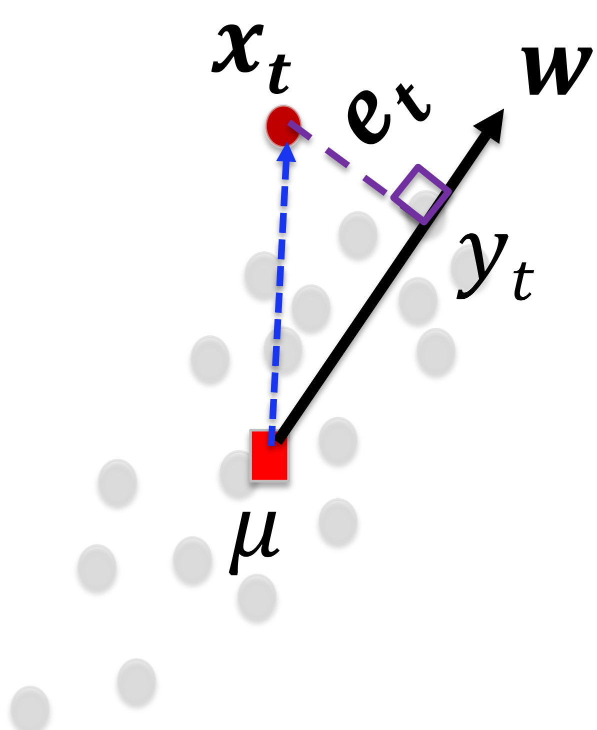
Outline

- 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



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

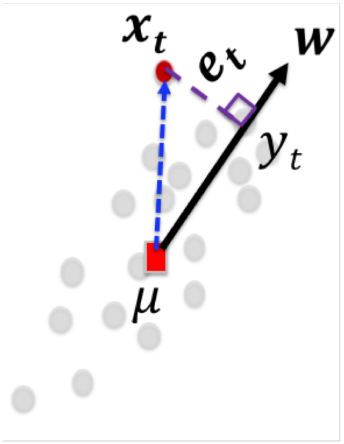
$$y_t = \boldsymbol{x_t^T w}$$

$$\boldsymbol{e_t} = ||\boldsymbol{x_t} - y_t \boldsymbol{w}||^2$$

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

Mean Square Error (MSE)

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



$$||w|| = 1$$

$\boldsymbol{x}_t^T \boldsymbol{x}_t - (\boldsymbol{x}_t^T \boldsymbol{w})\boldsymbol{w}^T \boldsymbol{x}_t - \boldsymbol{x}_t^T (\boldsymbol{x}_t^T \boldsymbol{w})\boldsymbol{w} + (\boldsymbol{x}_t^T \boldsymbol{w})\boldsymbol{w}^T (\boldsymbol{x}_t^T \boldsymbol{w})\boldsymbol{w}$
$= \boldsymbol{x}_t^T \boldsymbol{x}_t - \boldsymbol{w}^T (\boldsymbol{x}_t \boldsymbol{x}_t^T)\boldsymbol{w}$

Introduce a Lagrange multiplier λ

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

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

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

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

Eigenvalues and Eigenvectors

Lagrange multiplier

From wiki

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

Lagrange multiplier λ

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

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

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

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

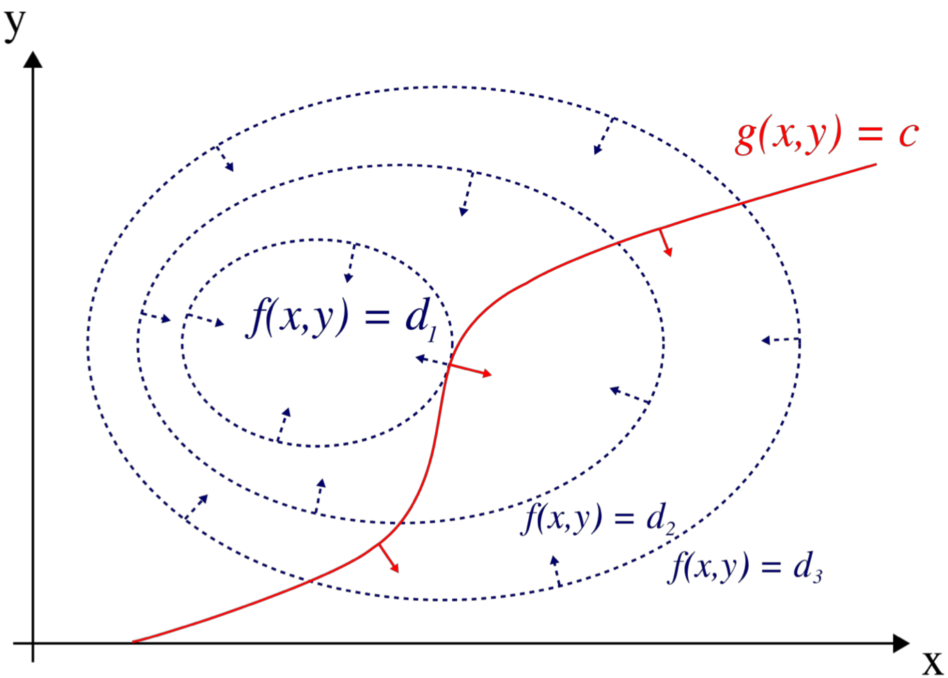
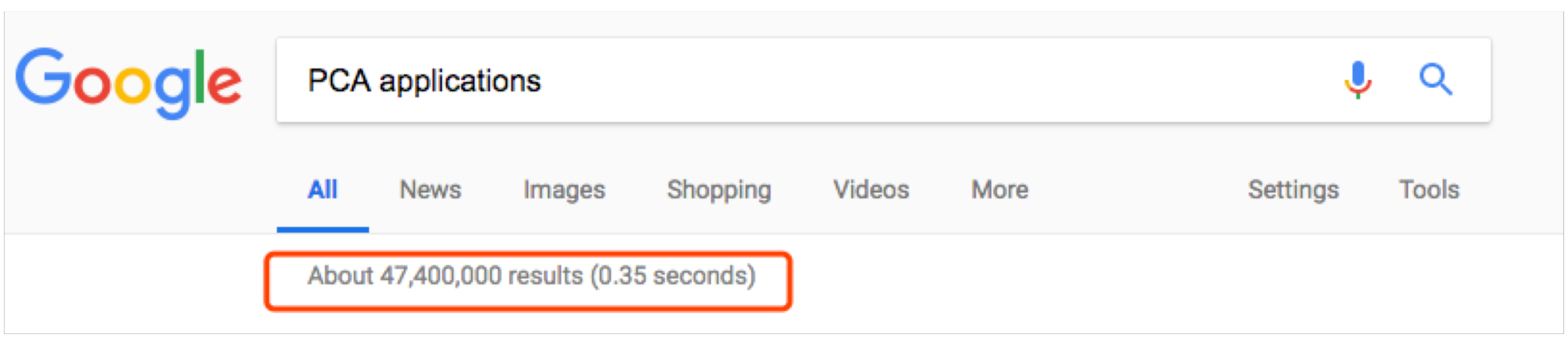


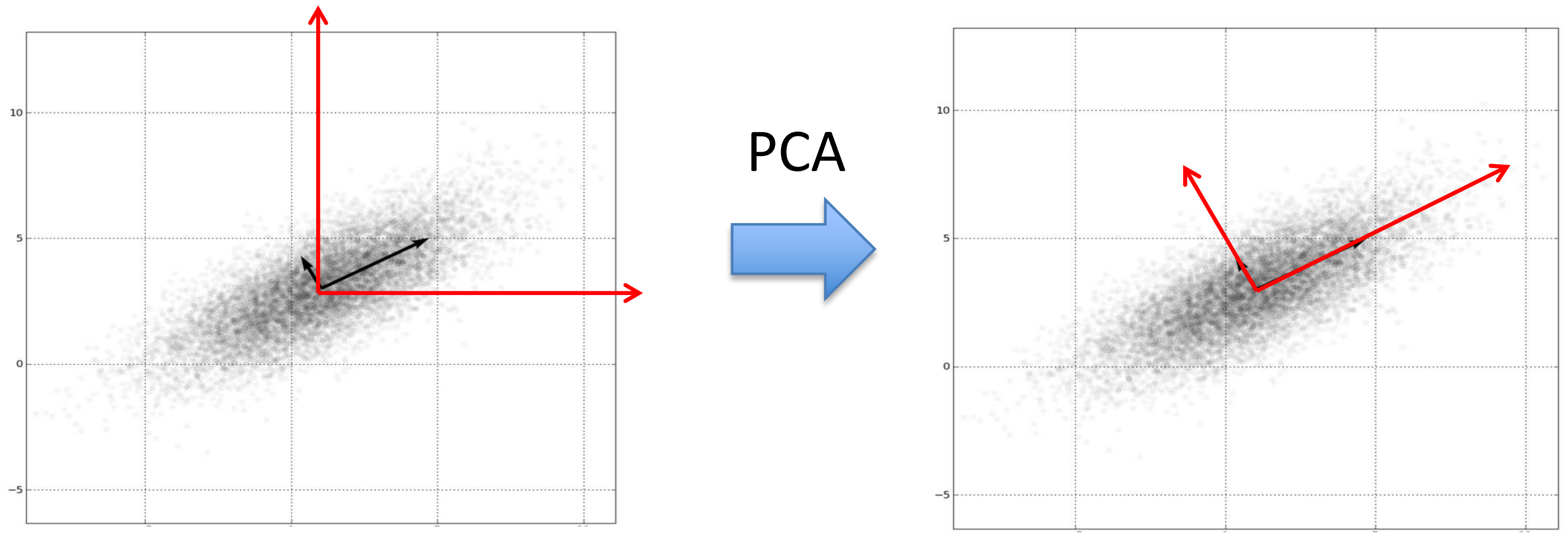
Figure 1: The red line shows the constraint $g(x, y) = c$. The blue lines are contours of $f(x, y)$. The point where the red line tangentially touches a blue contour is the maximum of $f(x, y)$, since $d_1 > d_2$.

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)

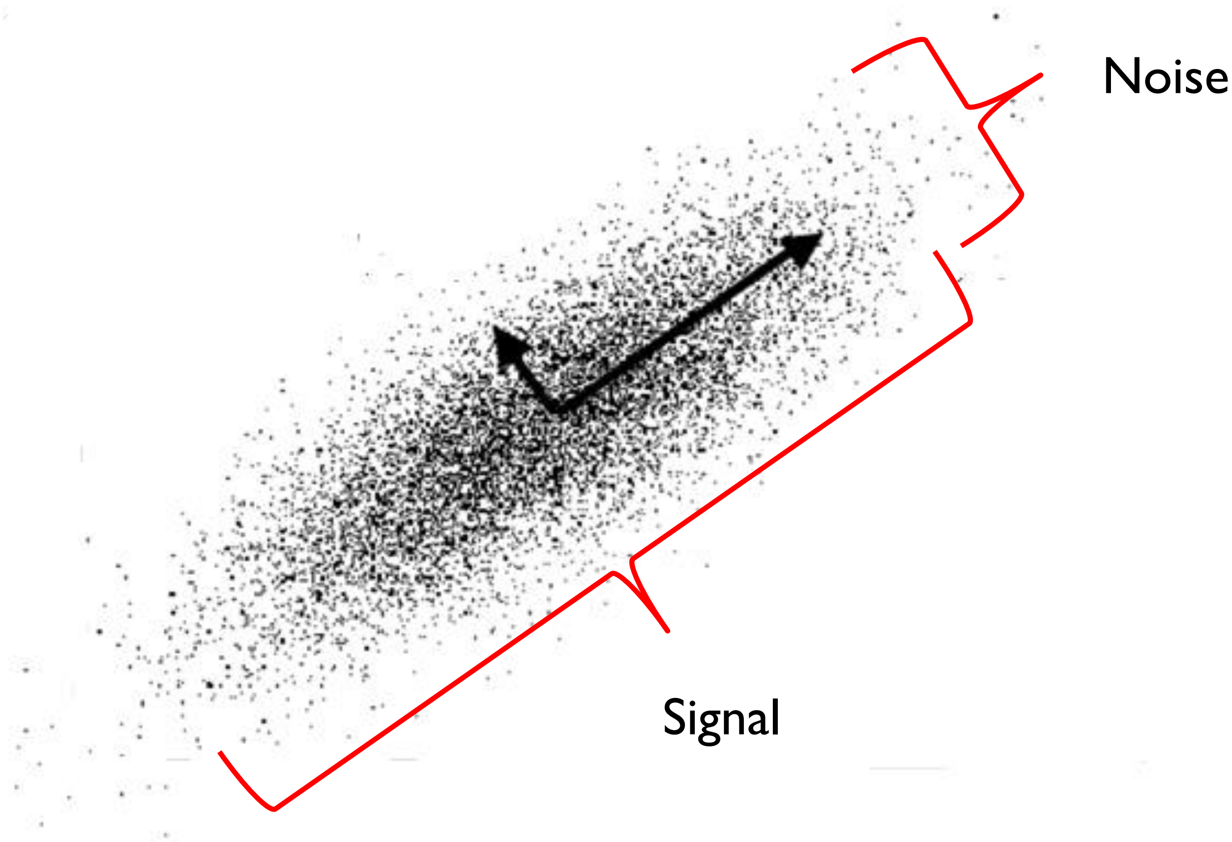


Data correlation



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

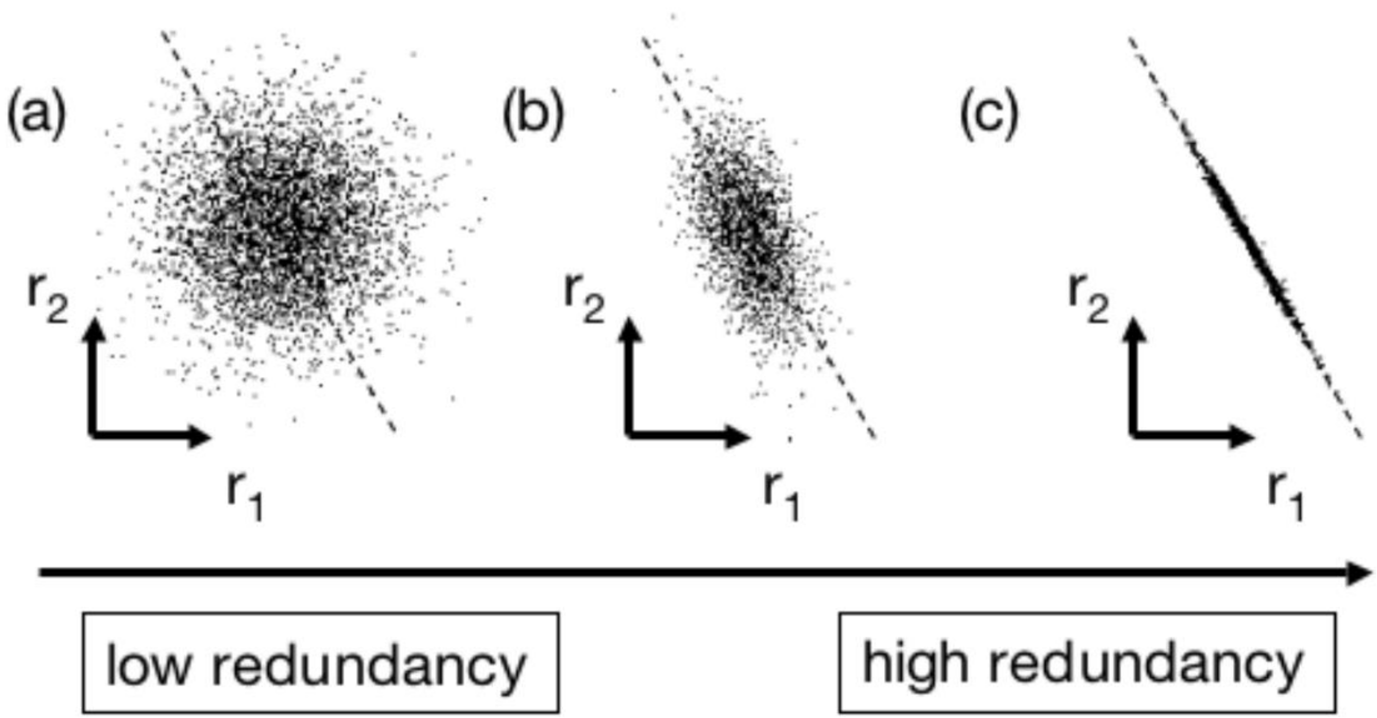


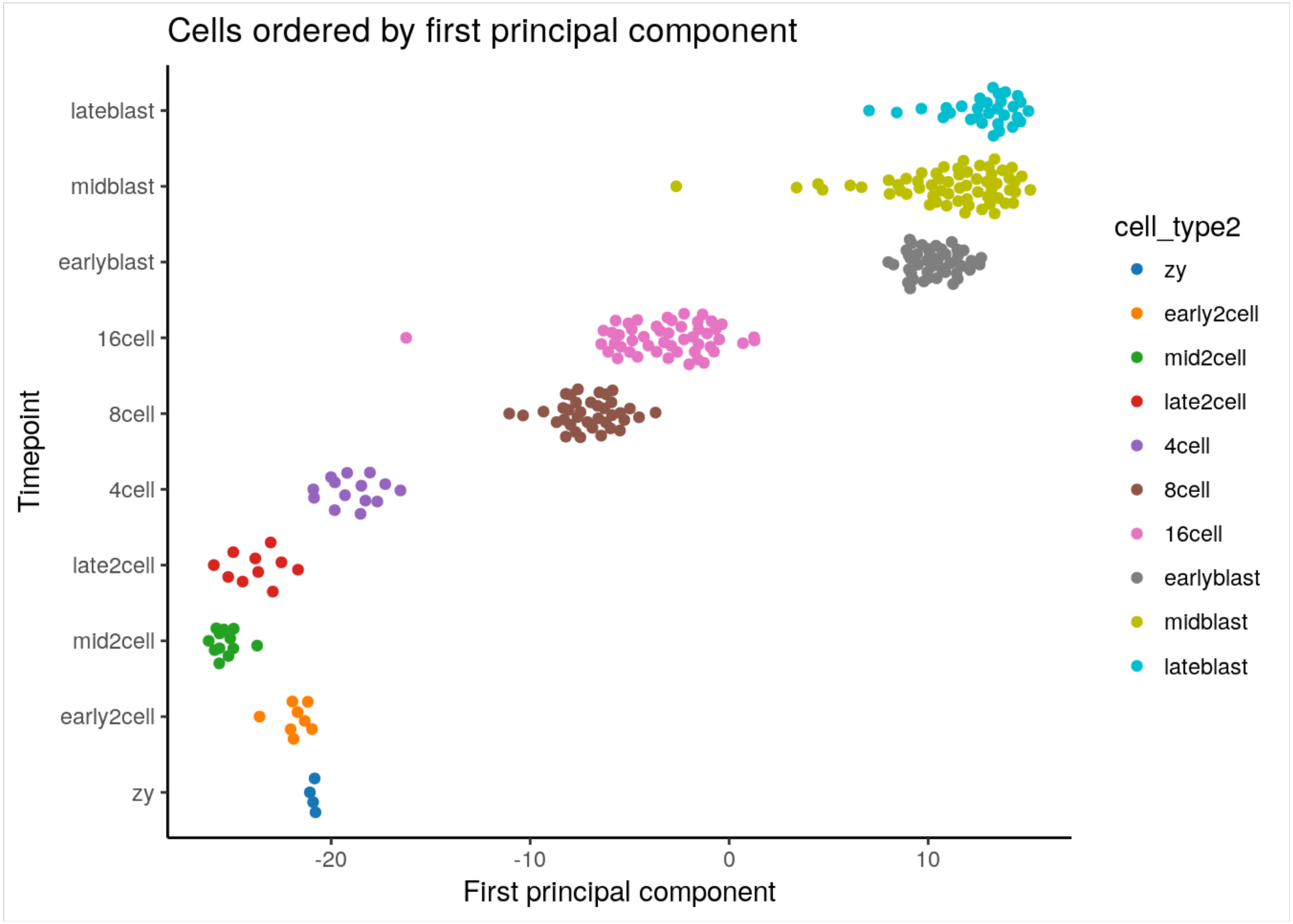
Image denoising by PCA

(a) Noisy image (b) PGPCA (PSNR=33.6) (c) PLPCA (PSNR=34.8)

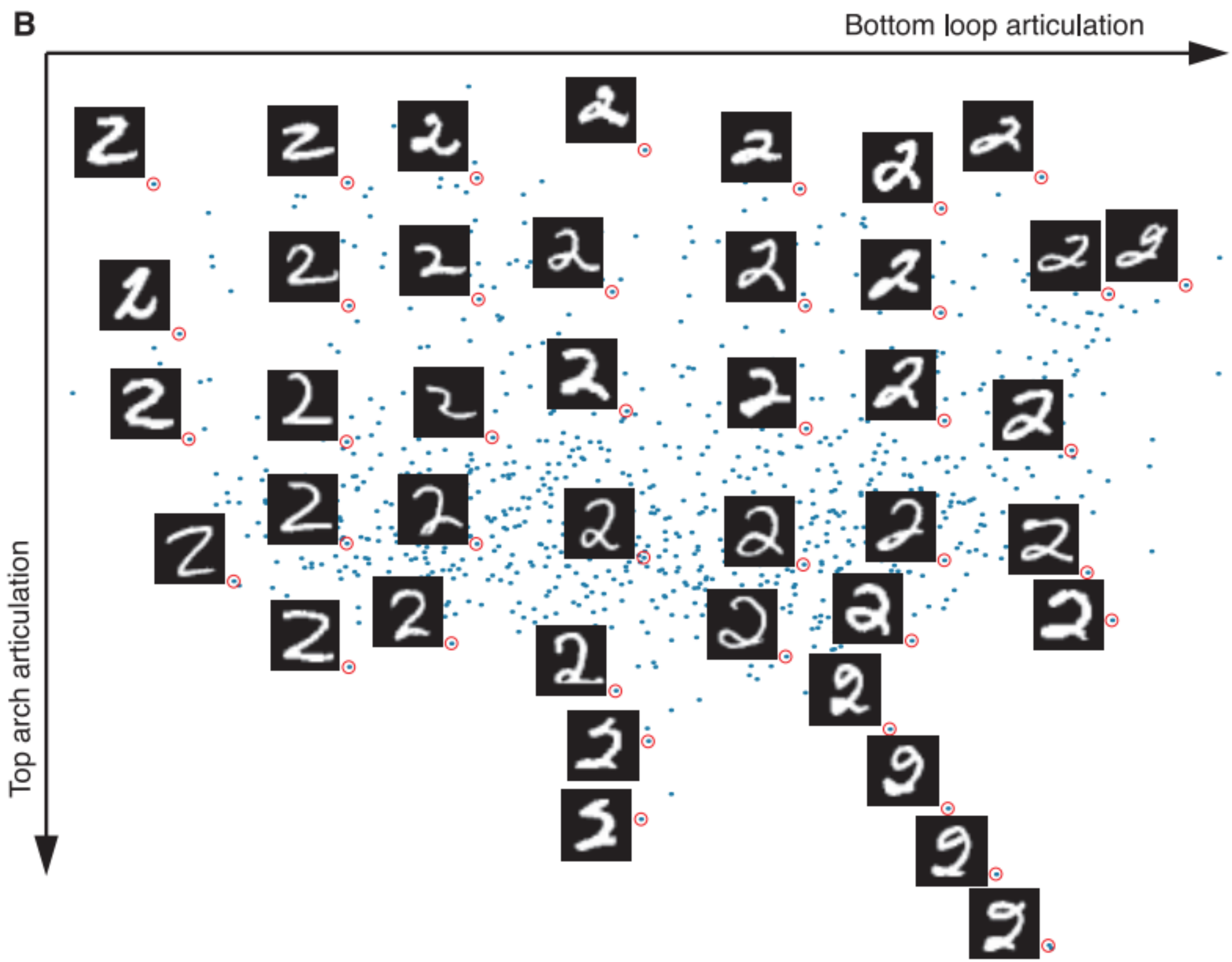
Figure 6: Visual evaluation of the denoising performance of PGPCA and PLPCA on an image (Barbara) damaged by an AWGN with noise level $\sigma = 10$, with their PSNR.

Charles-Alban Deledalle, Joseph Salmon, Arnak Dalalyan; BMVC 2011

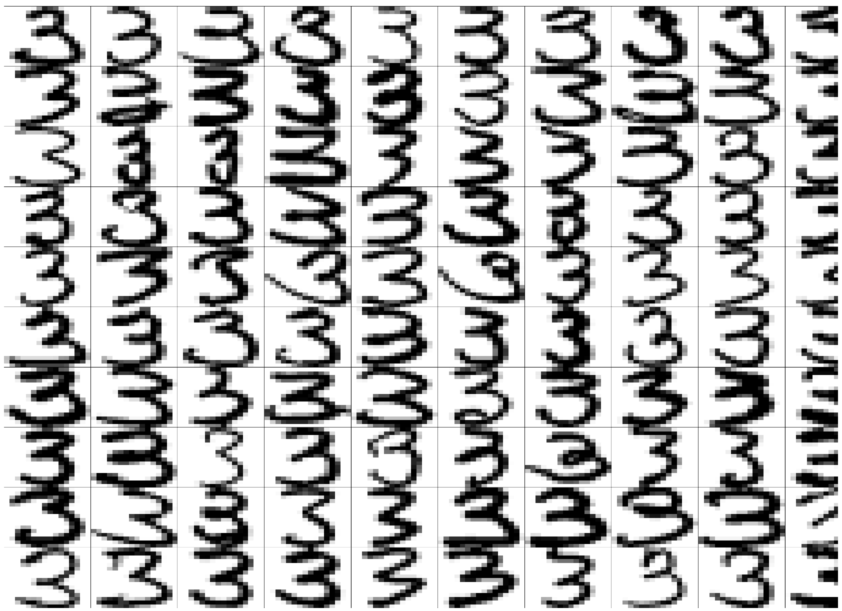
Order the cells by PCA



Visualize the hand-written digits



PCA of hand-written digits



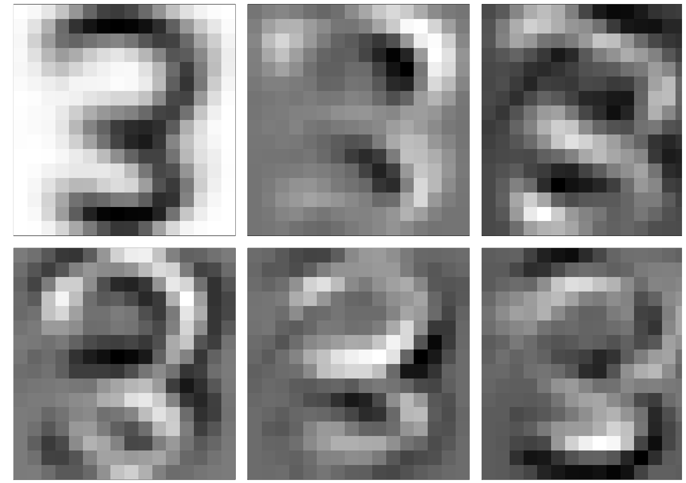


Figure 4: Digits: mean and eigenvectors

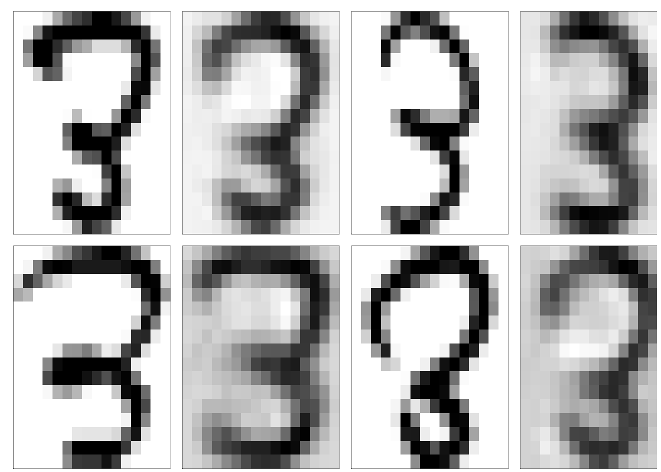


Figure 5: Digits data: Top: digits. Bottom: their reconstructions.

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 kth eigenface as:

  $$w_k = V_k^T(U - M)$$

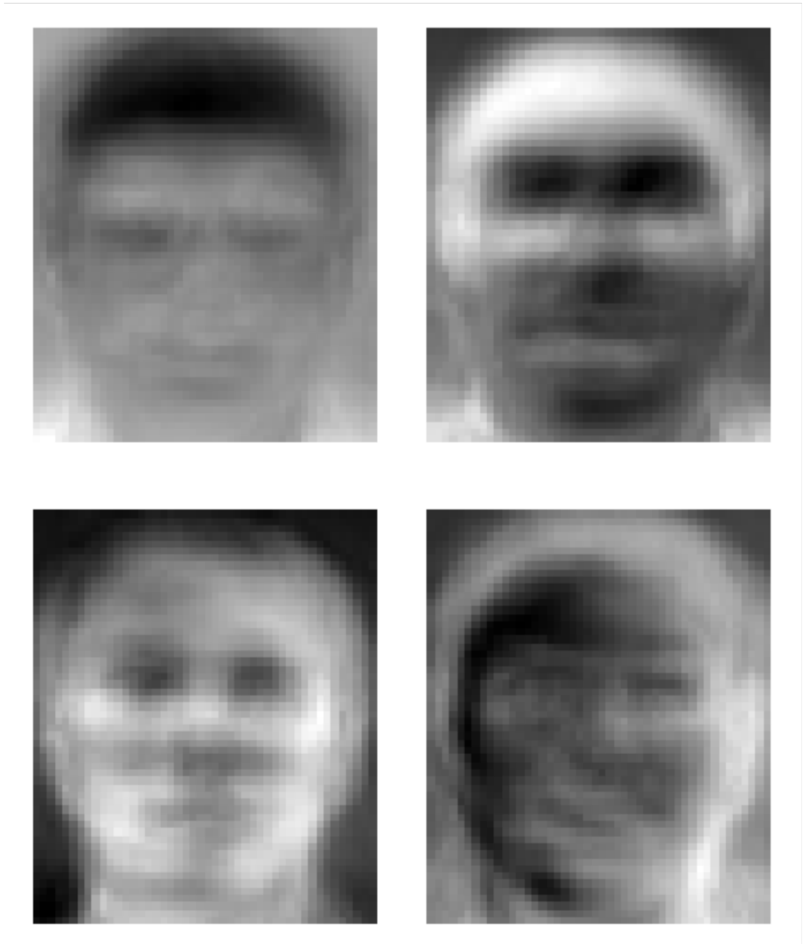
  Then form a weight vector $W = [w_1, w_2, \ldots, w_k, \ldots, w_n]$
- Compare W with weight vectors W_m of images in the database. Find the Euclidean distance.

  $$d = ||W - W_m||^2$$
- If $d < \epsilon_1$, then the mth 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.

https://en.wikipedia.org/wiki/Eigenface

Eigen-face examples

Eigen-face



Some eigenfaces from AT&T Laboratories Cambridge

Reconstruction by top-k eigenvectors

k=2 k=15 k=40

Thank you!