CS 4602

Introduction to Machine Learning

Dimensionality reduction

Instructor: Po-Chih Kuo

Roadmap

- Introduction and Basic Concepts
- Regression
- Bayesian Classifiers
- Decision Trees
- Linear Classifier
- Neural Networks
- Deep learning
- Convolutional Neural Networks
- The others
- KNN
- Clustering
- Data Exploration & Dimensionality reduction
- Model Selection and Evaluation

Outline

- Curse of dimensionality
- Linear Dimensionality reduction
  - PCA
  - ICA
- Nonlinear Dimensionality Reduction
  - MDS
  - ISOMAP
  - LLE
  - t-SNE

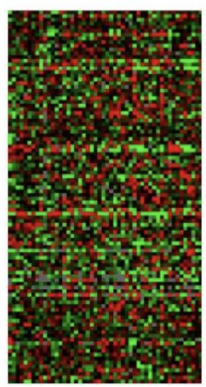
Lots of high-dimensional data...

face images

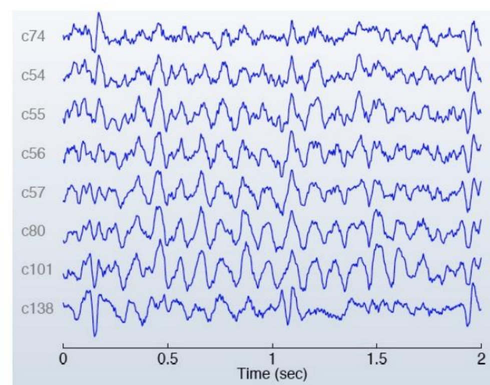
Zambian President Levy Mwanawasa has won a second term in office in an election his challenger Michael Sata accused him of rigging, official results showed on Monday.

According to media reports, a pair of hackers said on Saturday that the Firefox Web browser, commonly perceived as the safer and more customizable alternative to market leader Internet Explorer, is critically flawed. A presentation on the flaw was shown during the ToorCon hacker conference in San Diego.

documents



gene expression data



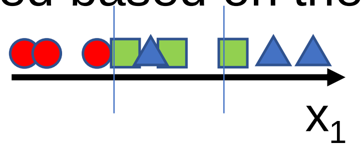
MEG readings

Why do dimensionality reduction?

- Computational: compress data ⇒ time/space efficiency
- Statistical: fewer dimensions ⇒ better generalization
- Visualization: understand structure of data
- Anomaly detection: describe normal data, detect outliers

The “curse of dimensionality”

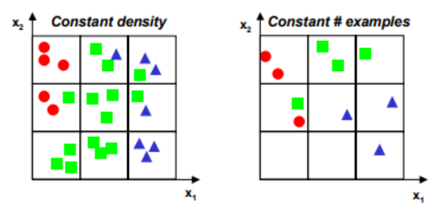
- Refers to the problems associated with multivariate data analysis as the dimensionality increases
- Consider a 3-class classification problem
  - Three types of objects have to be classified based on the value of a single feature:

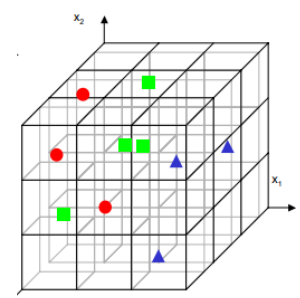


  - A simple procedure:
    - Divide the feature space into uniform bins
    - Compute the ratio of examples for each class at each bin
    - For a new example, find its bin and choose the predominant class in that bin
    - Notice that there exists a lot of overlap between classes
    - To improve discrimination, we decide to incorporate a second feature

Curse of dimensionality

- Moving to two dimensions
  - The number of bins increases from 3 to 3^2=**9**
  - QUESTION: Which should we maintain constant?
    - The density of examples per bin? This increases the number of examples from 9 to 27
    - The total number of examples? This results in a 2D scatter plot that is very sparse

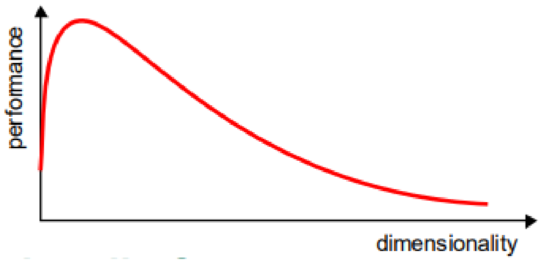




- Moving to three features …
  - The number of bins grows to 3^3=**27**
  - To maintain the initial density of examples, the number of required examples grows to 81
  - For the same number of examples, the 3D scatter plot is almost empty

Curse of dimensionality

- Implications of the curse of dimensionality
  - Exponential growth with dimensionality in the number of examples required to accurately estimate a function
- For a given sample size, there is a maximum number of features above which the performance of our classifier will degrade rather than improve
  - The information that was lost by discarding some features is compensated by a more accurate mapping in lower-dimensional space
  - Hence, more dimensional the training set is, the greater the risk of overfitting.
- How do we beat the curse of dimensionality?
  - By incorporating prior knowledge
  - By increasing the size of dataset
  - By reducing the dimensionality



Dimensionality Reduction

- In the presence of many of features, select the most relevant subset of (weighted) combinations of features.

Feature Selection: $X_1, \ldots, X_m \rightarrow X_{k1}, \ldots, X_{kp}$

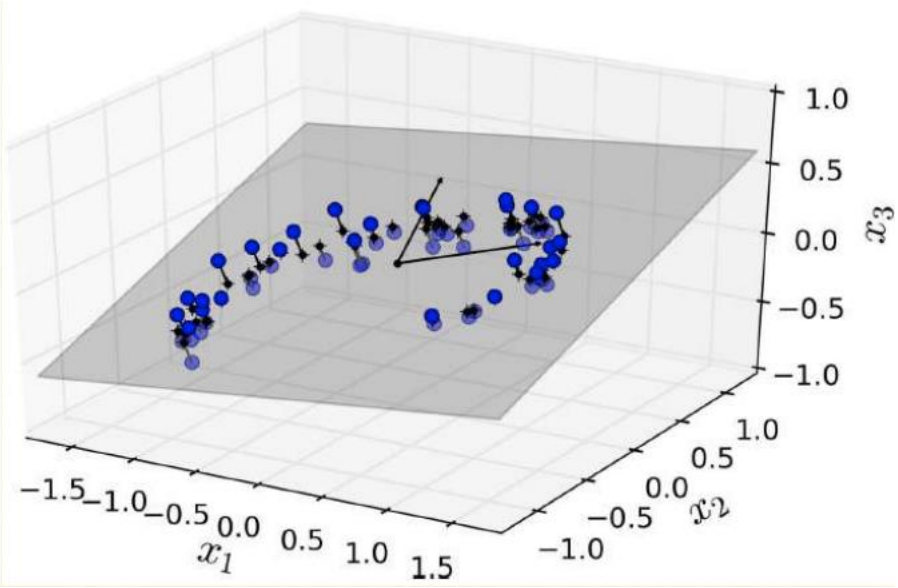
Dimensionality Reduction: $X_1, \ldots, X_m \rightarrow f_1(X_1, \ldots, X_m), \ldots, f_p(X_1, \ldots, X_m)$

- Linear feature extraction

$$\begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_N \end{bmatrix} \xrightarrow{\text{linear feature extraction}} \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_M \end{bmatrix} = \begin{bmatrix} w_{11} & w_{12} & \cdots & w_{1N} \\ w_{21} & w_{22} & \cdots & w_{2N} \\ \vdots & \vdots & \ddots & \vdots \\ w_{M1} & w_{M2} & & w_{MN} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_N \end{bmatrix}$$

Projection

- Most real-world problems do not have training instances spread out across all dimensions
- Many features are almost constant and correlated

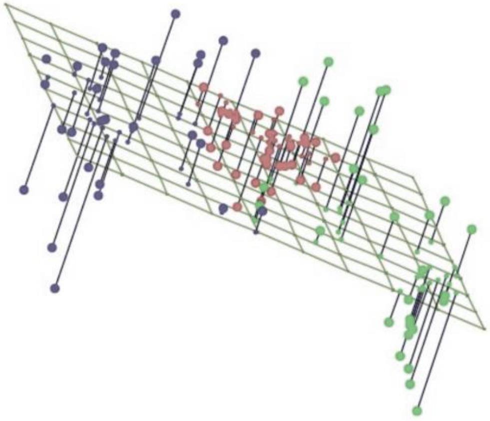


How many features are there?

Which of the feature is almost constant for almost all instances?

Basic idea of linear dimensionality reduction

Represent each face as a high-dimensional vec$\mathbf{x} \in \mathbb{R}^{361}$



$$\mathbf{x} \in \mathbb{R}^{361}$$

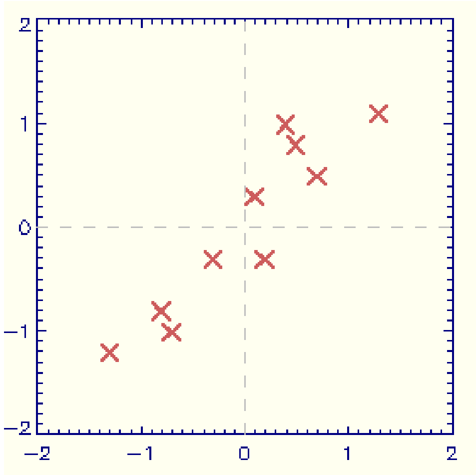
$$\mathbf{z} = \mathbf{U}^\top \mathbf{x}$$

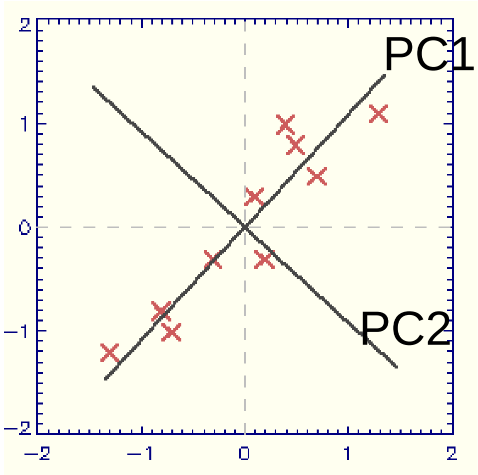
$$\mathbf{z} \in \mathbb{R}^{10}$$

How do we choose U ?

Principal Components Analysis (PCA)

- PCA finds a linear mapping of dataset x to a dataset z of lower dimensionality.





PCA objective 1: reconstruction error

Given n data points in d dimensions: $\mathbf{x}_1, \ldots, \mathbf{x}_n \in \mathbb{R}^d$

$$\mathbf{X} = \begin{pmatrix} | & & | \\ \mathbf{x}_1 & \cdots\cdots & \mathbf{x}_n \\ | & & | \end{pmatrix} \in \mathbb{R}^{d \times n}$$

Want to reduce dimensionality from d to k

Choose k directions $\mathbf{u}_1, \ldots, \mathbf{u}_k$

$$\mathbf{U} = \begin{pmatrix} | & & | \\ \mathbf{u}_1 & \cdot\cdot & \mathbf{u}_k \\ | & & | \end{pmatrix} \in \mathbb{R}^{d \times k}$$

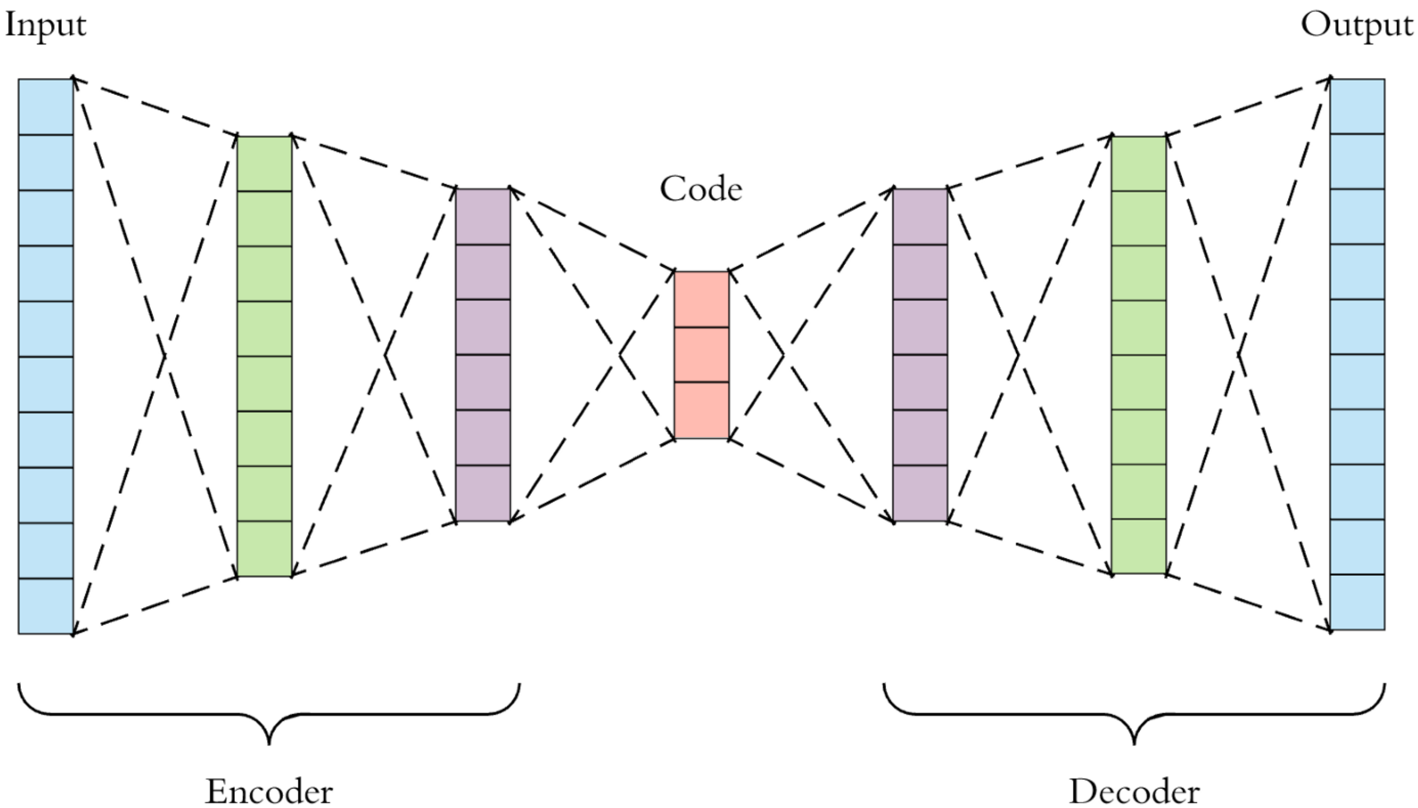
$\mathbf{U}$ serves two functions:

- Encode: $\mathbf{z} = \mathbf{U}^\top \mathbf{x}, \quad z_j = \mathbf{u}_j^\top \mathbf{x}$
- Decode: $\tilde{\mathbf{x}} = \mathbf{U}\mathbf{z} = \sum_{j=1}^k z_j \mathbf{u}_j$

Want reconstruction error $\|\mathbf{x} - \tilde{\mathbf{x}}\|$ to be small

Objective: minimize total squared reconstruction error $\min_{\mathbf{U} \in \mathbb{R}^{d \times k}} \sum_{i=1}^n \|\mathbf{x}_i - \mathbf{U}\mathbf{U}^\top \mathbf{x}_i\|^2$

Autoencoder!



PCA objective 2: projected variance

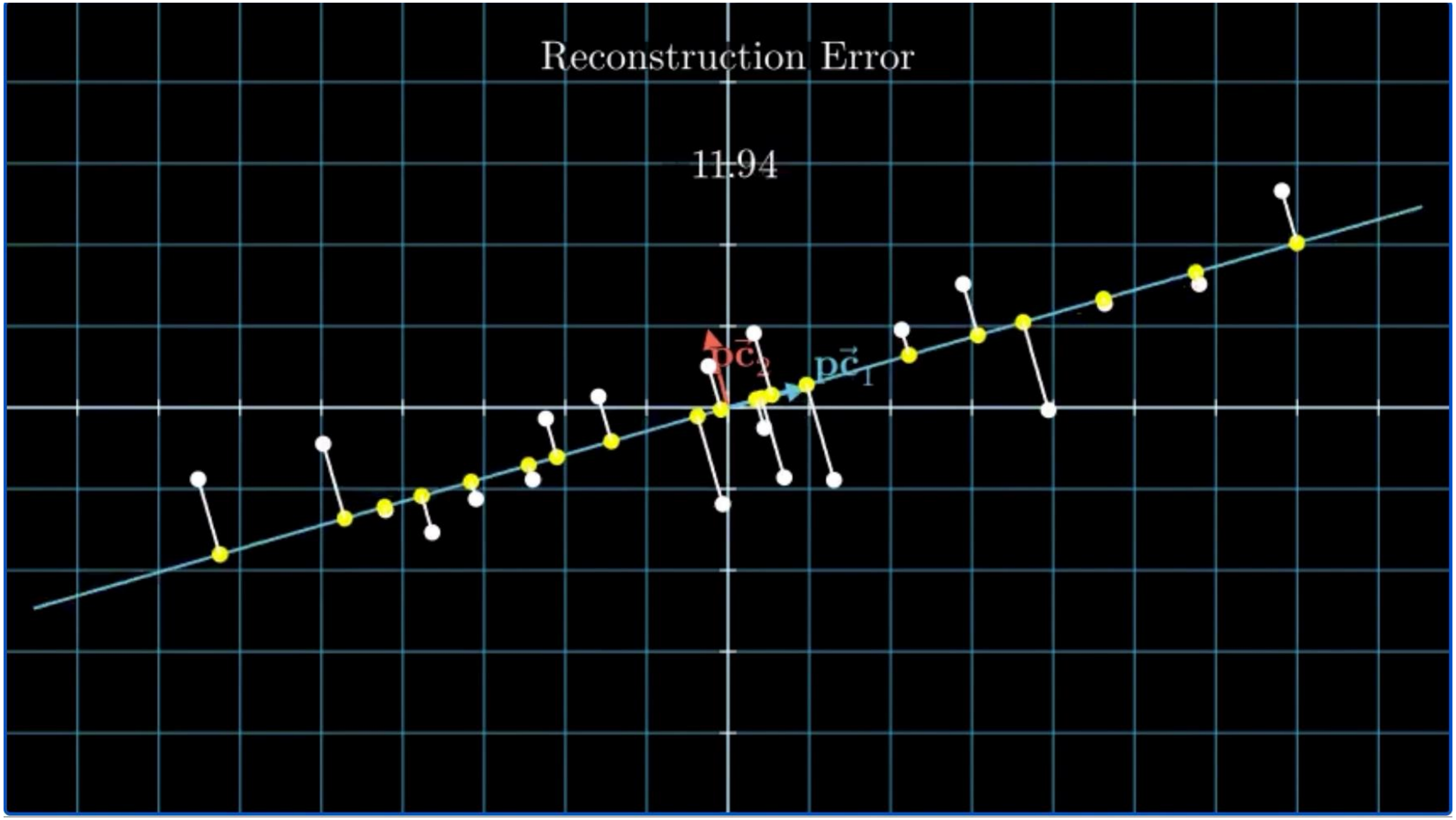
$$\max_{\mathbf{U} \in \mathbb{R}^{d \times k}, \mathbf{U}^\top \mathbf{U} = I} \hat{\mathbb{E}}[\|\mathbf{U}^\top \mathbf{x}\|^2]$$

Equivalence in objective 1 & 2

Key intuition:

$$\underbrace{\text{variance of data}}_{\text{fixed}} = \underbrace{\text{captured variance}}_{\text{want large}} + \underbrace{\text{reconstruction error}}_{\text{want small}}$$

Minimize reconstruction error $\leftrightarrow$ Maximize captured variance



From 3blue1brown

PCA

- How should we determine the "optimal" lower dimensional space basis vectors $<u_1, u_2, \ldots, u_K>$?

The optimal space of lower dimensionality can be defined by:

(1) Finding the eigenvectors u_i of the covariance matrix of the data $\Sigma_{\mathbf{x}}$

$$\Sigma_{\mathbf{x}} u_i = \lambda_i u_i$$

(2) Choosing the K "largest" eigenvectors u_i (i.e., corresponding to the K "largest" eigenvalues λ_i)

We refer to "largest" eigenvectors u_i as principal components.

Covariance

- Variance and Covariance:
  - Measure of the “spread” of a set of points around their mean
- Variance:
  - Measure of the deviation from the mean for points in one dimension
- Covariance:
  - Measure of how much each of the dimensions vary from the mean with respect to each other
  - Covariance sees if there is a relation between two dimensions
  - Covariance between one dimension is the variance

Covariance

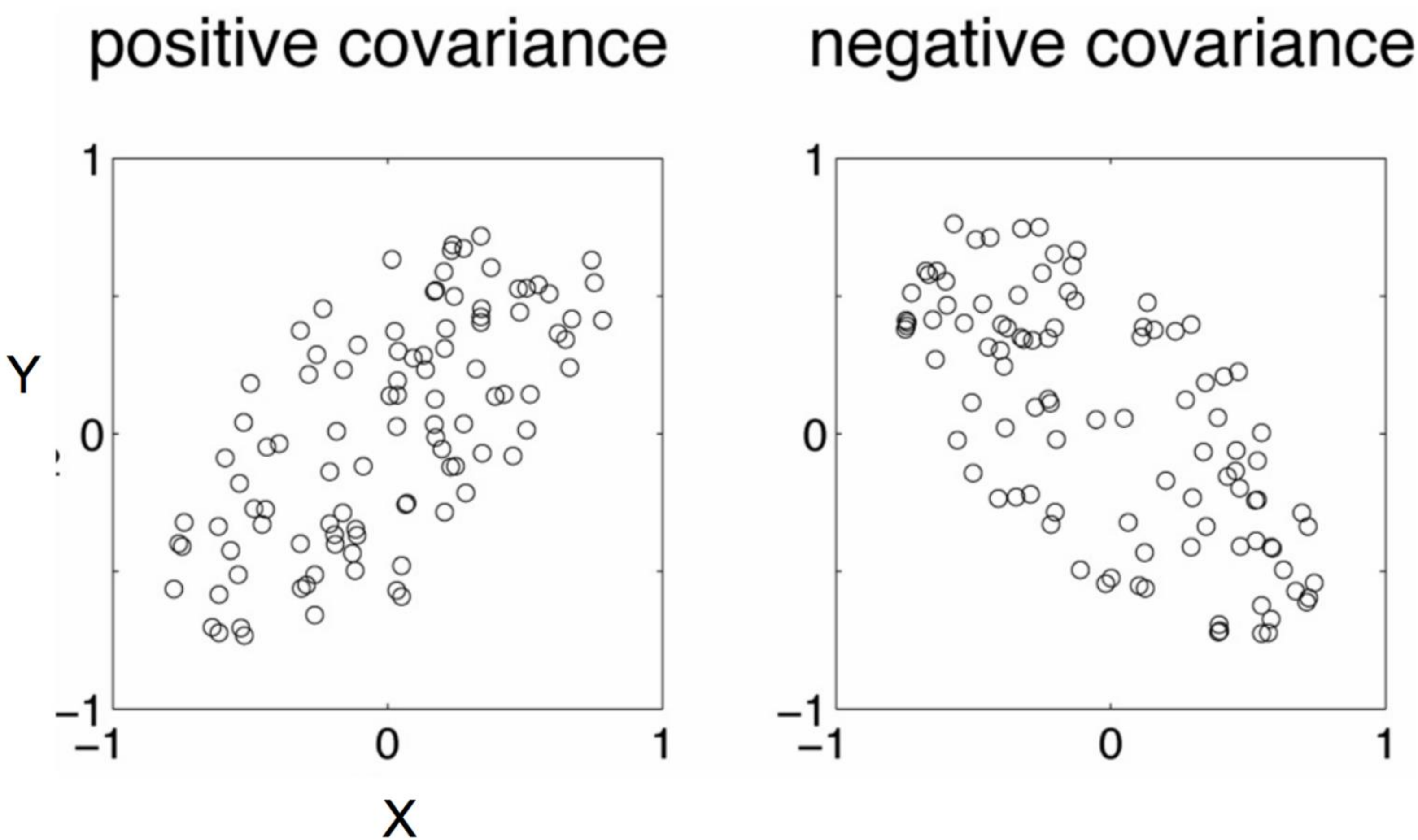
- Used to find relationships between dimensions in high dimensional data sets

$$\operatorname{cov}(X_i, X_j) = \mathrm{E}[(X_i - \mu_i)(X_j - \mu_j)]$$

- Covariance Matrix

$$\Sigma = \begin{bmatrix} \mathrm{E}[(X_1 - \mu_1)(X_1 - \mu_1)] & \mathrm{E}[(X_1 - \mu_1)(X_2 - \mu_2)] & \cdots & \mathrm{E}[(X_1 - \mu_1)(X_n - \mu_n)] \\ \mathrm{E}[(X_2 - \mu_2)(X_1 - \mu_1)] & \mathrm{E}[(X_2 - \mu_2)(X_2 - \mu_2)] & \cdots & \mathrm{E}[(X_2 - \mu_2)(X_n - \mu_n)] \\ \vdots & \vdots & \ddots & \vdots \\ \mathrm{E}[(X_n - \mu_n)(X_1 - \mu_1)] & \mathrm{E}[(X_n - \mu_n)(X_2 - \mu_2)] & \cdots & \mathrm{E}[(X_n - \mu_n)(X_n - \mu_n)] \end{bmatrix}.$$

$$\boxed{\mathbf{COR(X,\ Y)} = \frac{\mathbf{COV(X,\ Y)}}{\sqrt{\mathbf{VAR(X)VAR(Y)}}}}$$



Positive: Both dimensions increase together **Negative: While one increase the other decrease**

PCA - Steps

- Suppose we are given $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n$ d x 1 vectors

Step 1: compute sample mean

$$\bar{\mathbf{x}} = \frac{1}{n}\sum_{i=1}^{n}\mathbf{x}_i$$

Step 2: subtract sample mean (i.e., center data at zero)

$$\Phi_i = \mathbf{x}_i - \bar{\mathbf{x}}$$

Step 3: compute the sample covariance matrix Σ_x

$$\Sigma_{\mathbf{x}} = \frac{1}{n}\sum_{i=1}^{n}\Phi_i\,\Phi_i^T \qquad = \frac{1}{n}AA^T$$

, where $A=[\Phi_1\ \Phi_2\ \ldots\ \Phi_n]$ (d x n matrix)

PCA - Steps

Step 4: compute the eigenvalues/eigenvectors of Σ_x

$$\Sigma_x u_i = \lambda_i u_i$$

we assume

eigenvalues $\lambda_1 > \lambda_2 > \ldots > \lambda_d$ and $u_1, u_2, \ldots, u_d$ are the corresponding eigenvectors

Since Σ_x is symmetric, <$u_1, u_2, \ldots, u_d$> form an orthogonal basis in R^d, therefore:

$$\mathbf{x} - \bar{\mathbf{x}} = \sum_{i=1}^{d} z_i u_i = z_1 u_1 + z_2 u_2 + \ldots + z_d u_d$$

$$z_i = \frac{(\mathbf{x} - \overline{\mathbf{x}})^T u_i}{u_i^T u_i} = (\mathbf{x} - \overline{\mathbf{x}})^T u_i \text{ if } ||u_i|| = 1$$

Note : most software packages normalize u_i to unit length to simplify calculations

PCA - Steps

Step 5: dimensionality reduction step – approximate **x** using only the first K eigenvectors (K<d) (i.e., corresponding to the K largest eigenvalues where K is a parameter):

$$\mathbf{x} - \bar{\mathbf{x}} = \sum_{i=1}^{d} z_i u_i = z_1 u_1 + z_2 u_2 + \ldots + z_d u_d$$

$$\Big\Downarrow \quad \text{approximate using first K terms}$$

$$\hat{\mathbf{x}} - \bar{\mathbf{x}} = \sum_{i=1}^{K} z_i u_i = z_1 u_1 + z_2 u_2 + \ldots + z_K u_K$$

or

$$(\hat{\mathbf{x}} - \bar{\mathbf{x}}) = U \begin{bmatrix} z_1 \\ z_2 \\ \cdot \\ \cdot \\ z_K \end{bmatrix}$$

$$\text{where } \mathrm{U} = [\mathrm{u}_1 \mathrm{u}_2 \ldots \mathrm{u}_\mathrm{K}] \ \ \mathrm{dxK}$$

i.e., the columns of U are the the first K eigenvectors of $\Sigma_\mathbf{x}$

PCA – Linear Transformation

- The linear transformation $R^d \rightarrow R^K$ which performs the dimensionality reduction is:

$$\mathbf{y} = \mathbf{U}^T\mathbf{x} \in R^K \text{ where } K<d$$

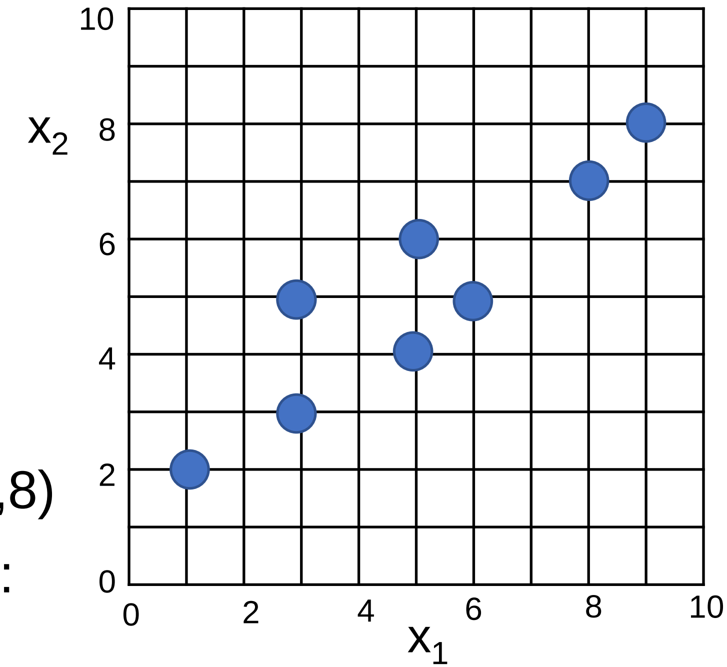
$$\begin{bmatrix} z_1 \\ z_2 \\ . \\ . \\ z_K \end{bmatrix} = U^T(\hat{\mathbf{x}} - \bar{\mathbf{x}}) = \begin{bmatrix} u_1^T \\ u_2^T \\ . \\ . \\ u_K^T \end{bmatrix} (\hat{\mathbf{x}} - \bar{\mathbf{x}})$$

i.e., the rows of U^T are the first K eigenvectors of $\Sigma_\mathbf{x}$

Example

- Compute the PCA for dataset

  (1,2),(3,3),(3,5),(5,4),(5,6),(6,5),(8,7),(9,8)



What do they look like?

- Compute the sample covariance matrix is:

$$\Sigma_x = \frac{1}{n}\sum_{i=1}^{n}(\mathbf{x}_i - \hat{\boldsymbol{\mu}})(\mathbf{x}_i - \hat{\boldsymbol{\mu}})^{\mathrm{T}}$$

$$\Sigma_x = \begin{bmatrix} 6.25 & 4.25 \\ 4.25 & 3.5 \end{bmatrix}$$

- The eigenvalues can be computed by finding the roots of the characteristic polynomial:

$$\Sigma_x v = \lambda v \Rightarrow |\Sigma_x - \lambda I| = 0$$
$$\Rightarrow \begin{vmatrix} 6.25 - \lambda & 4.25 \\ 4.25 & 3.5 - \lambda \end{vmatrix} = 0$$
$$\Rightarrow \boldsymbol{\lambda_1 = 9.34; \lambda_2 = 0.41}$$

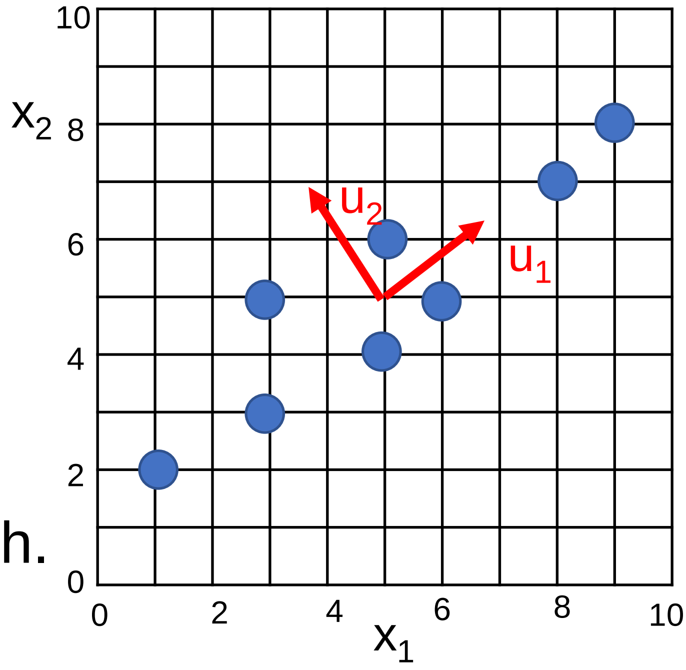
Example (cont'd)

- The eigenvectors are the solutions of the systems:

$$\Sigma_{\mathbf{x}} u_i = \lambda_i u_i$$

$$\begin{bmatrix} 6.25 & 4.25 \\ 4.25 & 3.5 \end{bmatrix} \mathrm{u}_1 = \begin{bmatrix} \lambda_1 v_{11} \\ \lambda_1 v_{12} \end{bmatrix} \Rightarrow \mathrm{u}_1 = \begin{bmatrix} 0.81 \\ 0.59 \end{bmatrix}$$

$$\begin{bmatrix} 6.25 & 4.25 \\ 4.25 & 3.5 \end{bmatrix} \mathrm{u}_2 = \begin{bmatrix} \lambda_2 v_{21} \\ \lambda_2 v_{22} \end{bmatrix} \Rightarrow \mathrm{u}_2 = \begin{bmatrix} -0.59 \\ 0.81 \end{bmatrix}$$



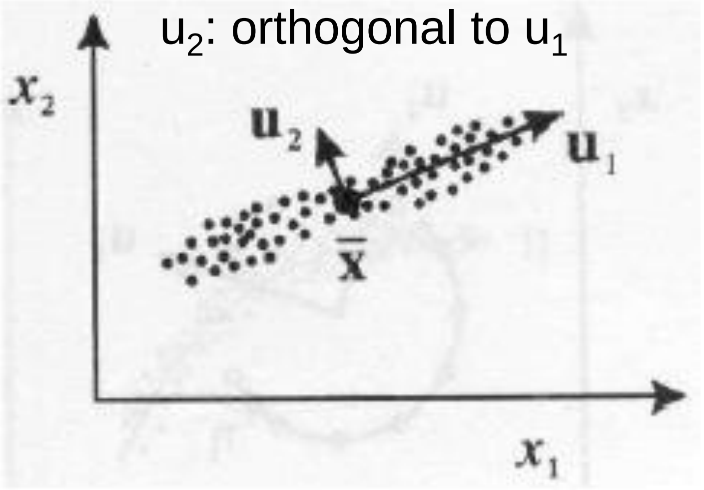
- Normalize the eigenvectors to unit-length.

Note: if u_i is a solution, then (cu_i) is also a solution where c is any constant.

Geometric interpretation

- PCA chooses the eigenvectors of the covariance matrix corresponding to the largest eigenvalues.
- The eigenvalues correspond to the variance of the data along the eigenvector directions.
- Therefore, PCA projects the data along the directions where the data varies most!

u_1: direction of max variance
u_2: orthogonal to u_1



How do we choose K?

- *Similar to question of "How many clusters?"*
- *K* is typically chosen based on how much information (variance) we want to preserve:

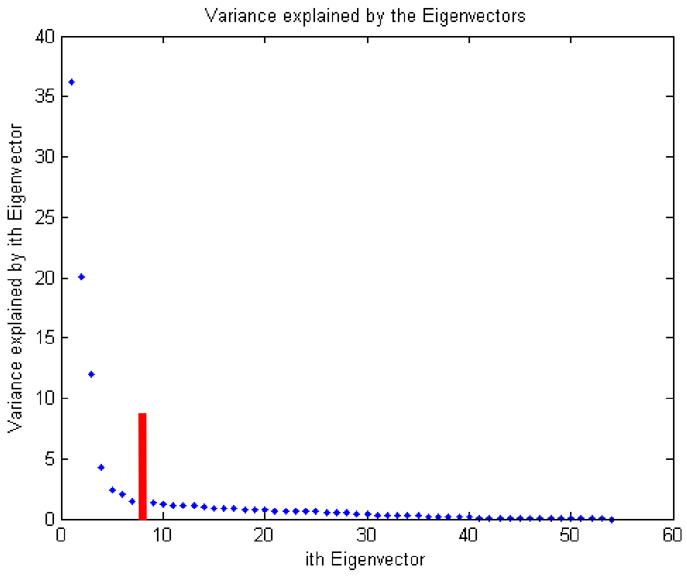
$$\frac{\sum_{i=1}^{K} \lambda_i}{\sum_{i=1}^{N} \lambda_i} > T \quad where\ T\ is\ a\ threshold\ (e.g., 0.9)$$

- If T=0.9, for example, we say that we "preserve" 90% of the information (variance) in the data.

- If K=N, then we "preserve" 100% of the information in the data (i.e., just a change of basis)

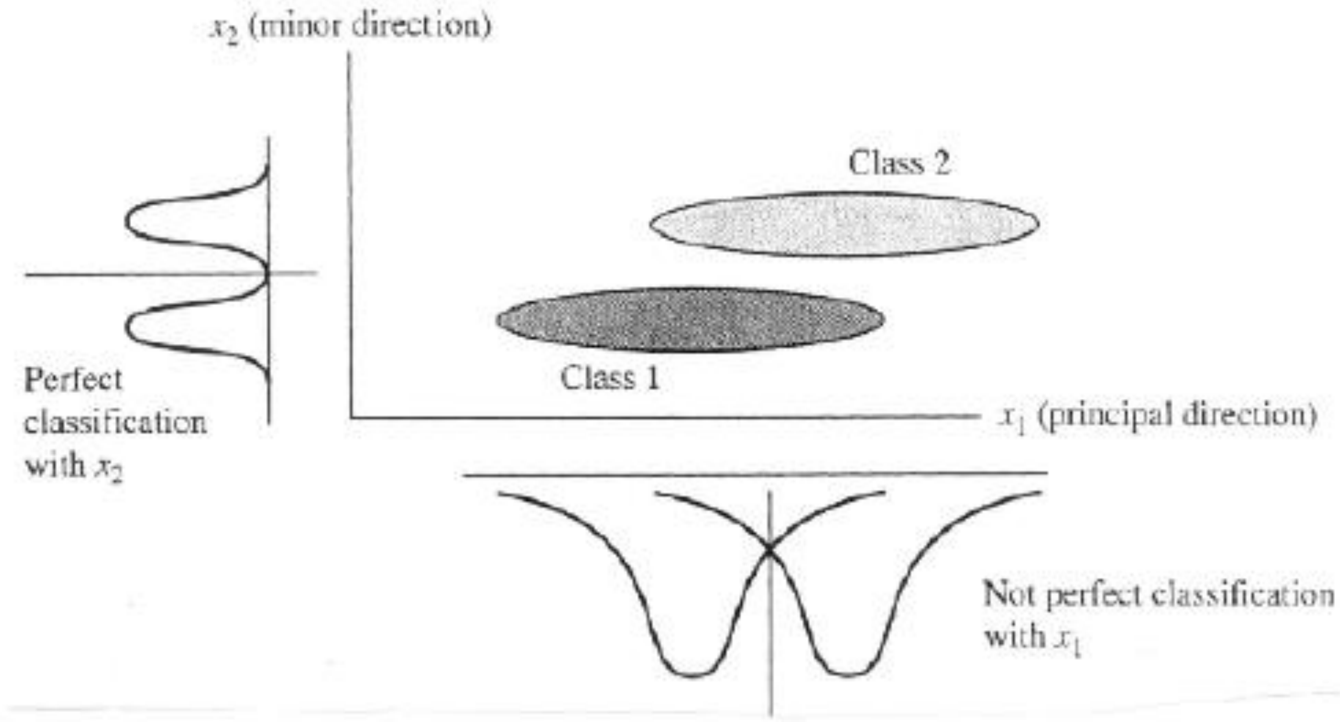
How do we choose K

- *Check the distribution of eigenvalues*
- *Take enough eigenvectors to cover 80-90% of the variance*



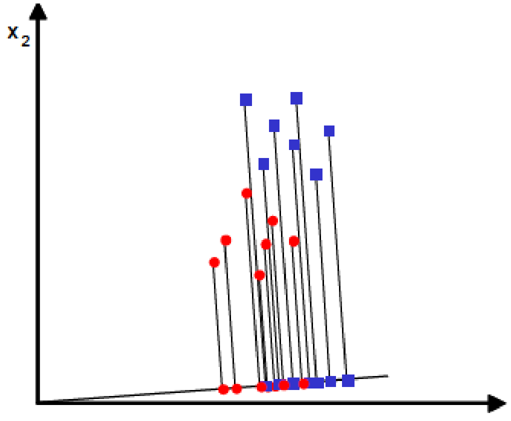
Limitations

- PCA is not always an optimal dimensionality-reduction technique for classification purposes.

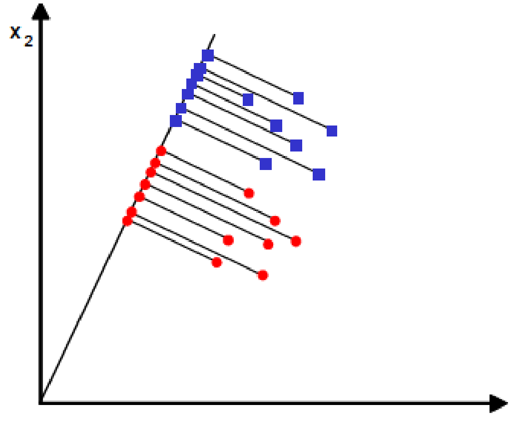


Linear Discriminant Analysis (LDA)

- What is the goal of LDA?
  - Seeks to find directions along which the classes are best separated (i.e., increase discriminatory information).
  - It takes into consideration the scatter within-classes and between-classes.
  - See previous slides for more details.

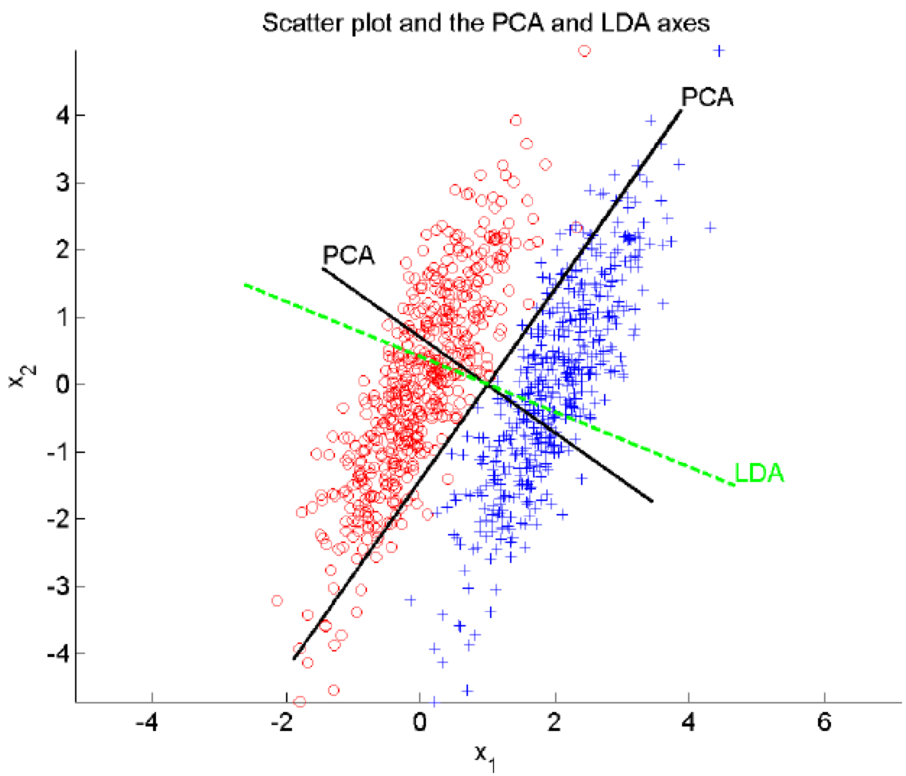


Bad separability

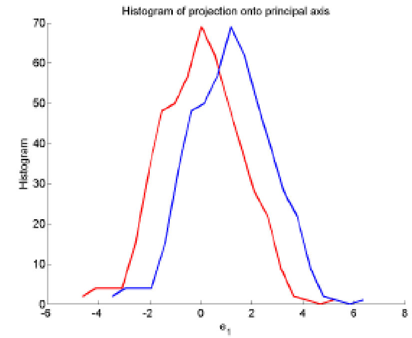


Good separability

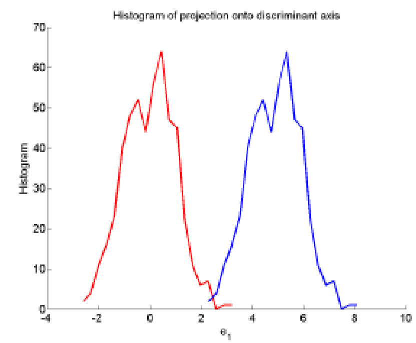
PCA vs. LDA



(a) Scatter plot.



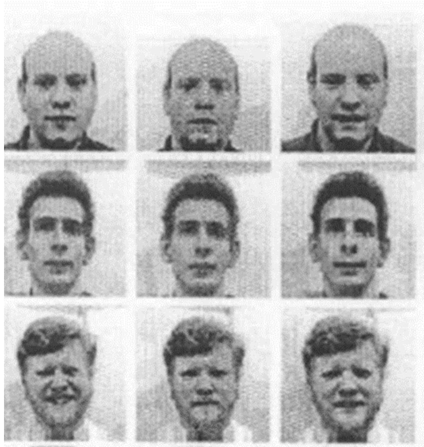
(b) Projection onto the first PCA axis.

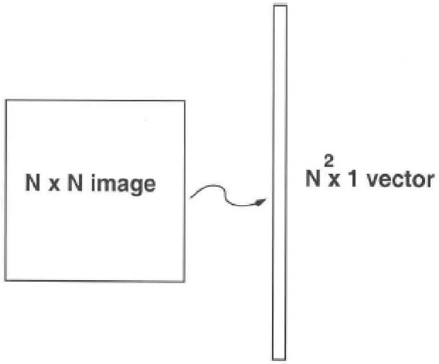


(c) Projection onto the first LDA axis.

Application to Images

- The goal is to represent images in a space of lower dimensionality using PCA.
  - Useful for various applications, e.g., face recognition, image compression, etc.
- Given M images of size N x N, first represent each image as a 1D vector (i.e., by stacking the rows together).
  - Note that for face recognition, faces must be centered and of the same size.





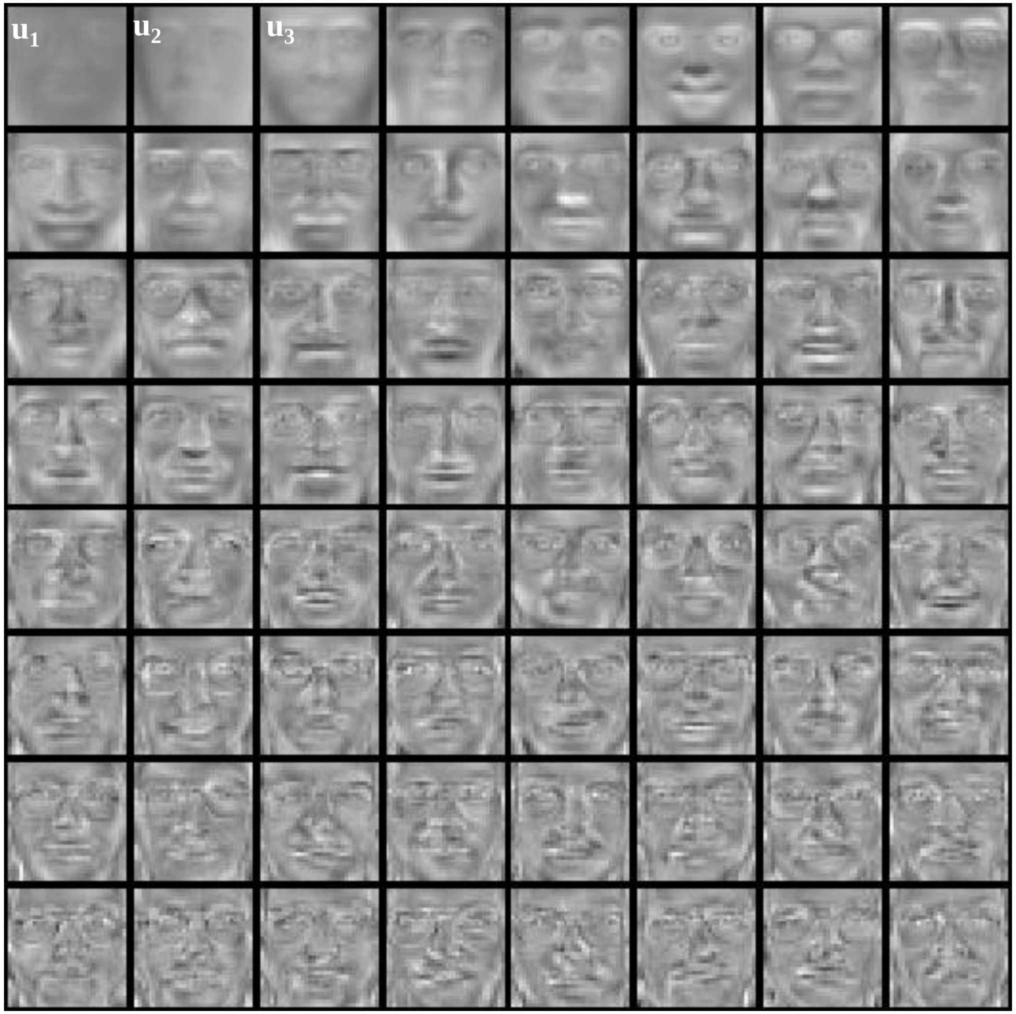
Example: face recognition

Dataset

Top eigenvectors: $\mathbf{u}_1,\ldots\mathbf{u}_k$

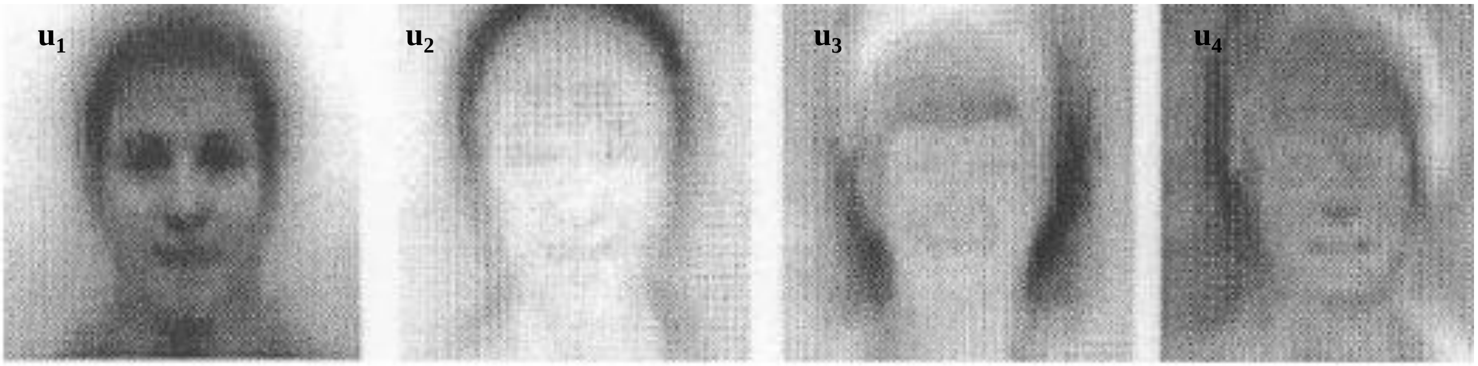
(visualized as images - eigenfaces)

Mean face: $\bar{\mathbf{x}}$



Example (cont'd)

- Interpretation: represent a face in terms of eigenfaces



$$\hat{\mathbf{x}} = \sum_{i=1}^{K} z_i u_i = z_1 u_1 + z_2 u_2 + \ldots + z_K u_K + \bar{\mathbf{x}}$$

$$Z = \begin{bmatrix} z_1 \\ z_2 \\ \cdot \\ \cdot \\ z_K \end{bmatrix}$$

$+\overline{\mathbf{x}}$

Experiments in the original Eigenface paper presented the following results: an average of 96% with light variation, 85% with orientation variation, and 64% with size variation. (Turk & Pentland 1991)

Image Compression

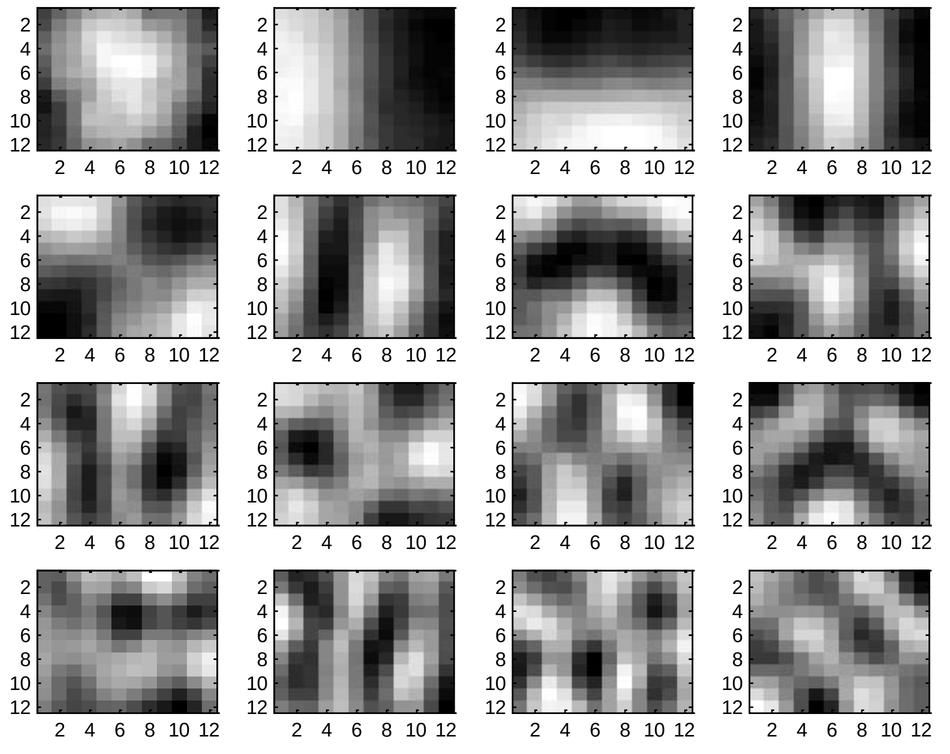
Original Image

- Divide the original 372x492 image into patches:
  - Each patch is an instance that contains 12x12 pixels on a grid
- View each as a 144-D vector

PCA compression: 144D → 60D

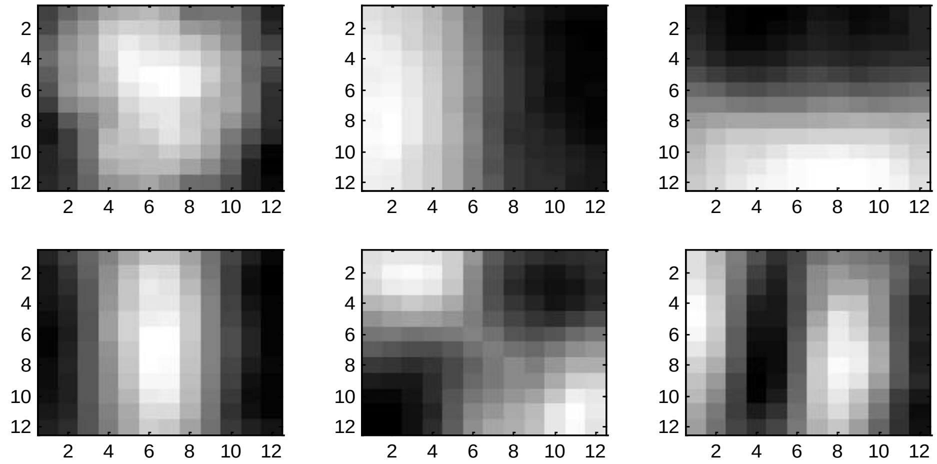
PCA compression: 144D → 16D

16 most important eigenvectors



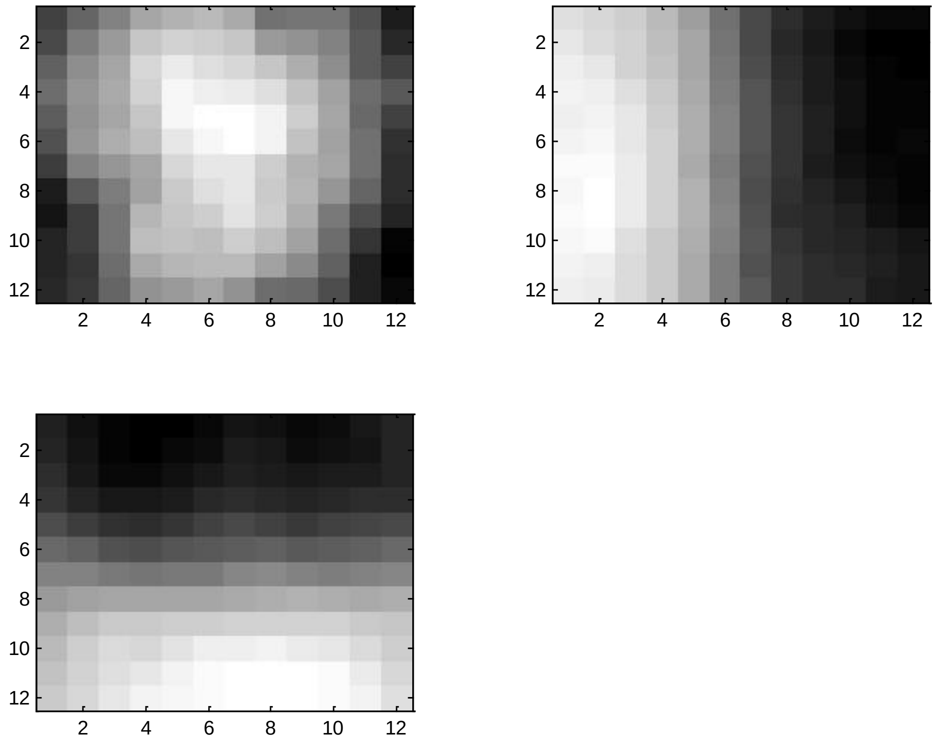
PCA compression: 144D → 6D

6 most important eigenvectors



PCA compression: 144D → 3D

3 most important eigenvectors



PCA compression: 144D → 1D?

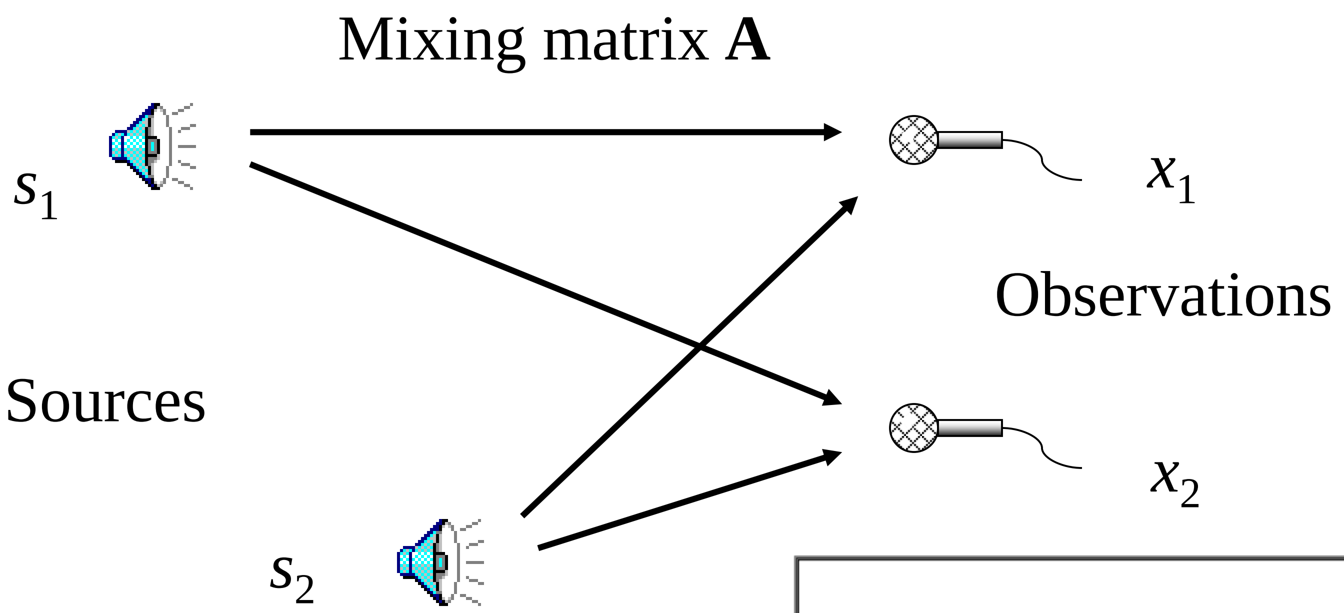
Independent Component Analysis (ICA)

Blind Signal Separation (BSS) or Independent Component Analysis (ICA) is the identification & separation of mixtures of sources

- Applications include:
  - Audio Processing
  - Medical data
  - Finance
  - Coding
- … and most applications where PCA is currently used.
- While PCA seeks directions that represents data best in a $\boldsymbol{\Sigma|x_0 - x|^2}$ sense, ICA seeks such directions that are most independent from each other.

Often used on Time Series separation of Multiple Targets

The “Cocktail Party” Problem



$$\boxed{X = AS}$$

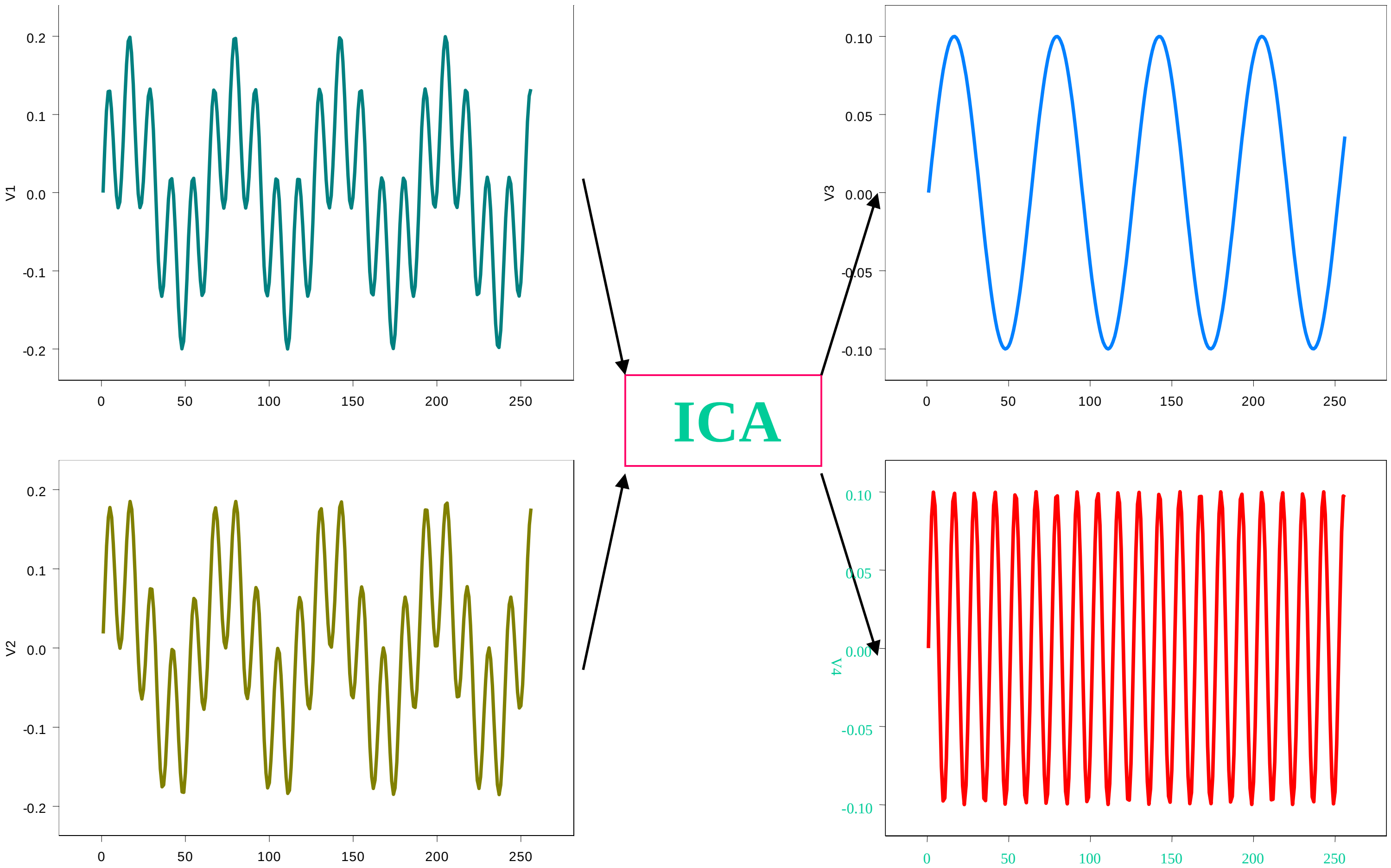
“observations” “mixing matrix” “sources”

n sources, m=*n* observations

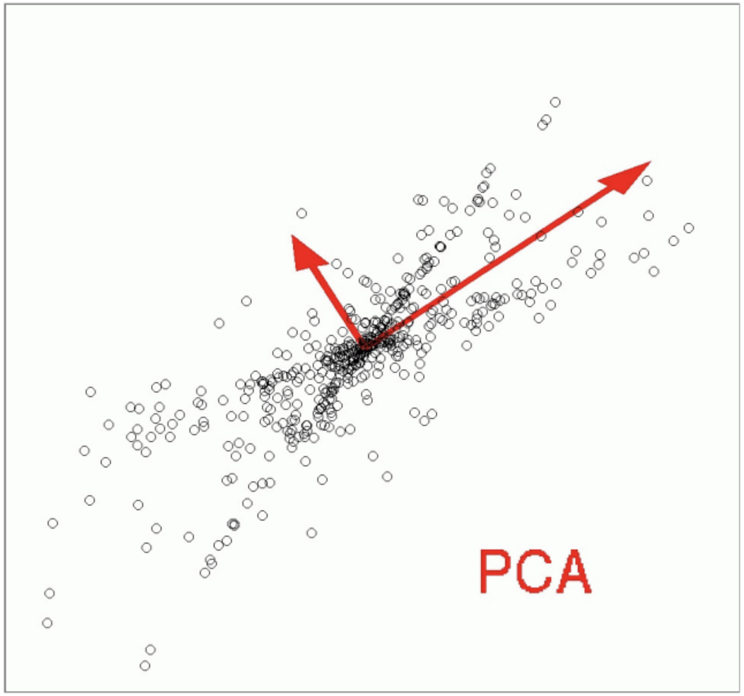
ICA estimation

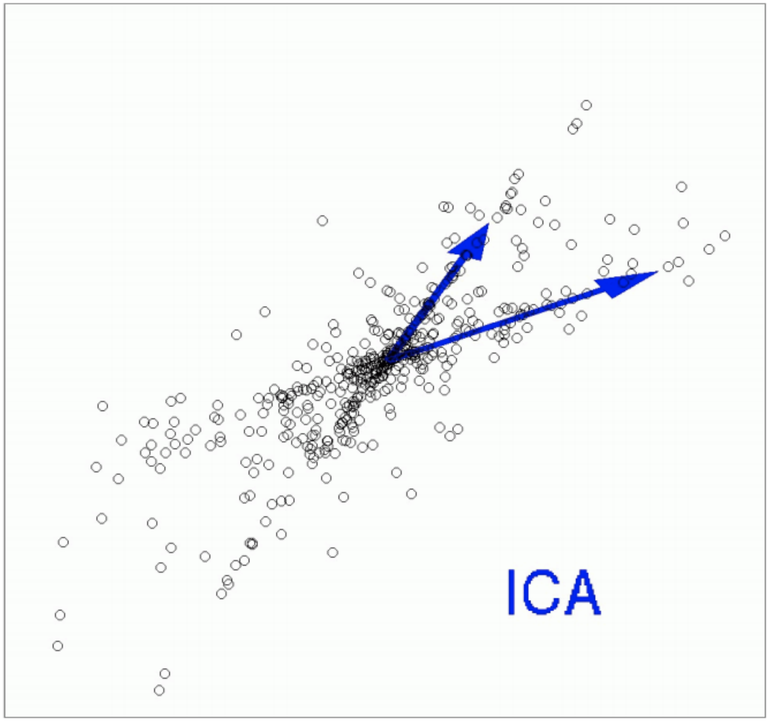
Observing signals

Original source signal



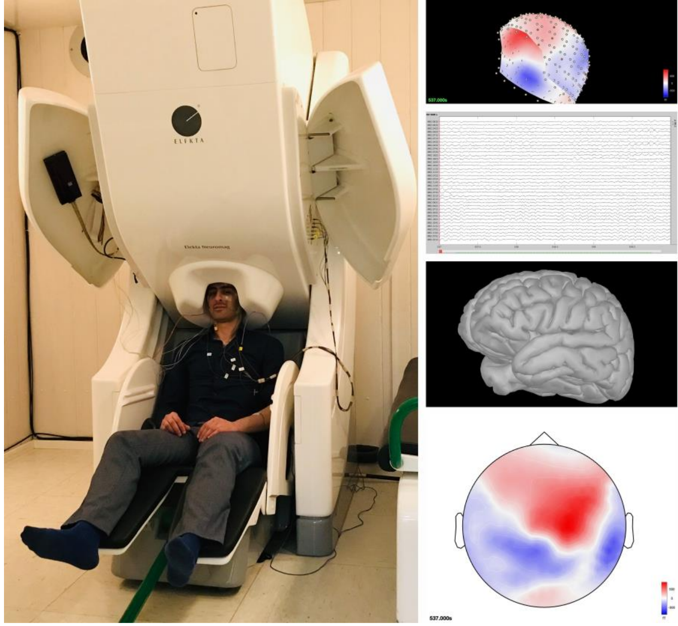
ICA vs PCA





- PCA vectors are orthogonal
- ICA vectors are not orthogonal

Application: Signal processing



- Data: MEG data
  - Eye artifacts:
    - ask person to “blink” and to make “horizontal saccades”
  - Muscle artifacts:
    - Asked to bite teeth for as long as 20 seconds.
  - Other artifact:
    - Cardiac cycle
- Subset: 12 subset of MEG signals $x_i(t)$

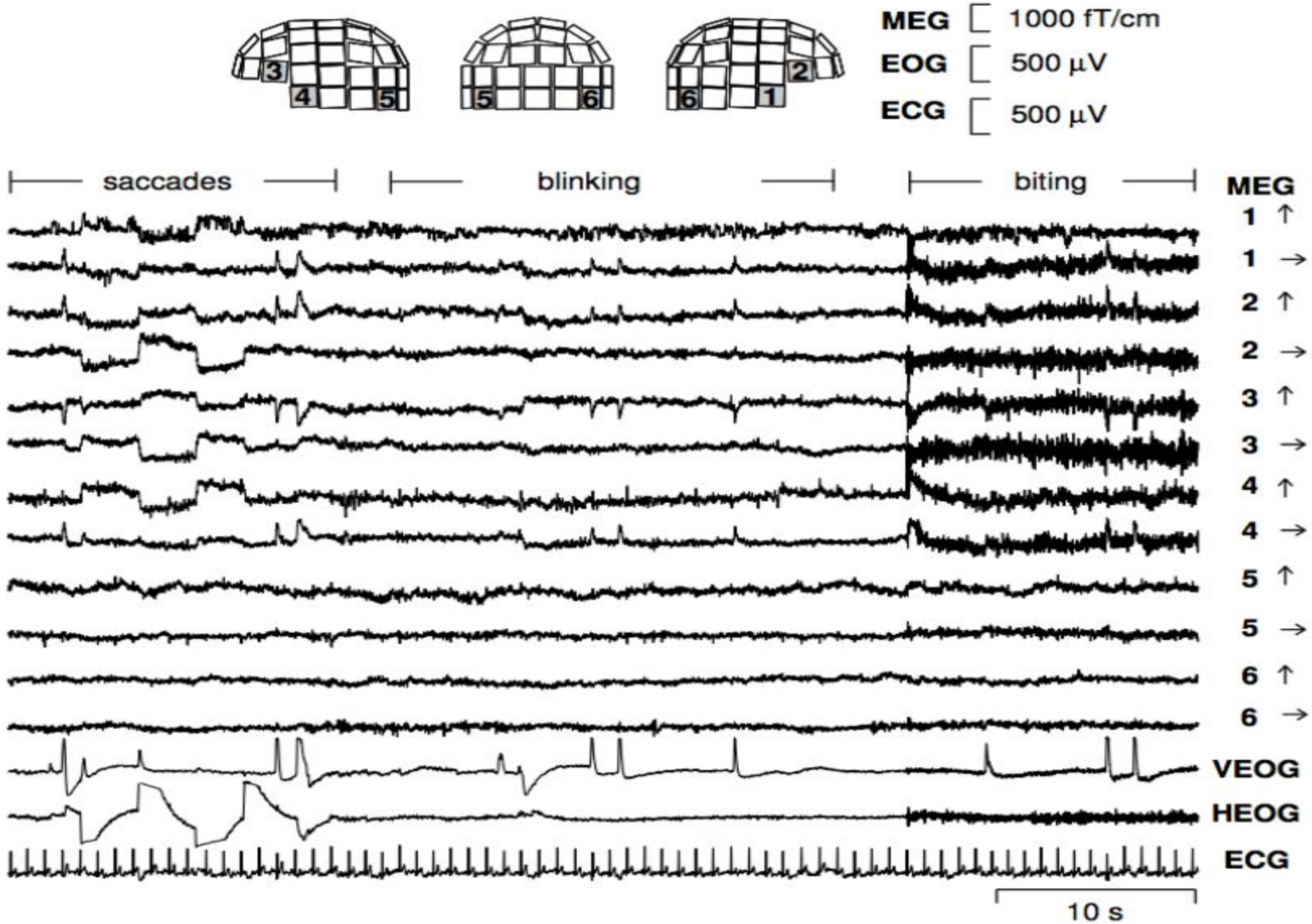
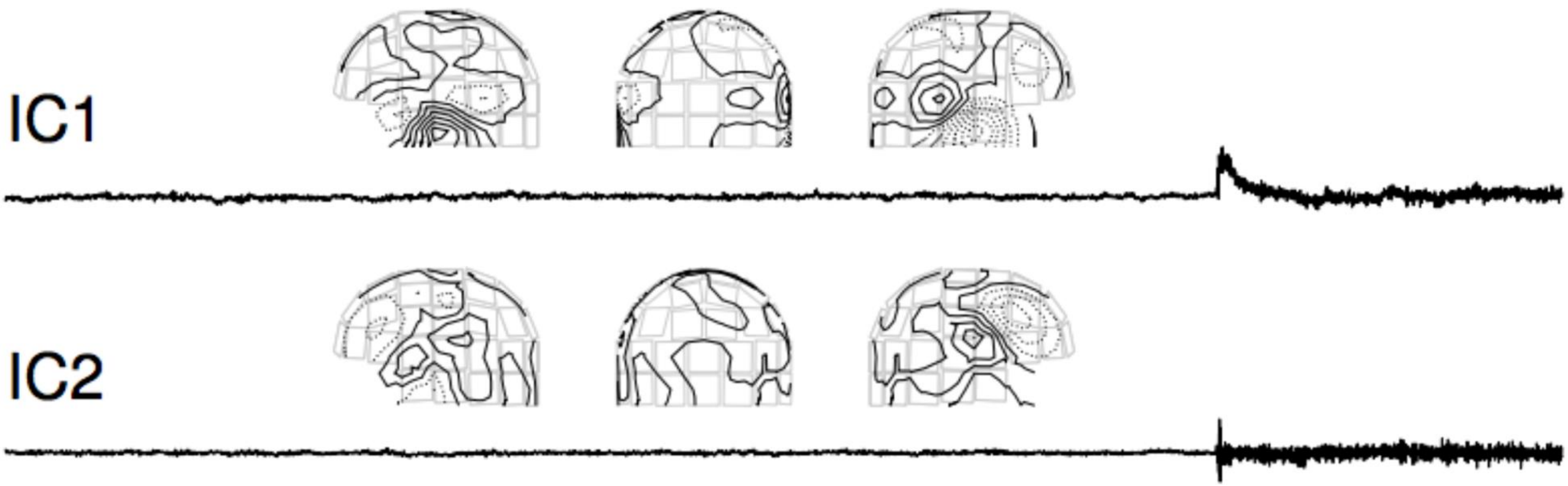


Figure: Samples of MEG signals, showing artifacts produced by blinking, saccades, biting and cardiac cycle. For each of the 6 positions shown, the two orthogonal directions of the sensors are plotted.

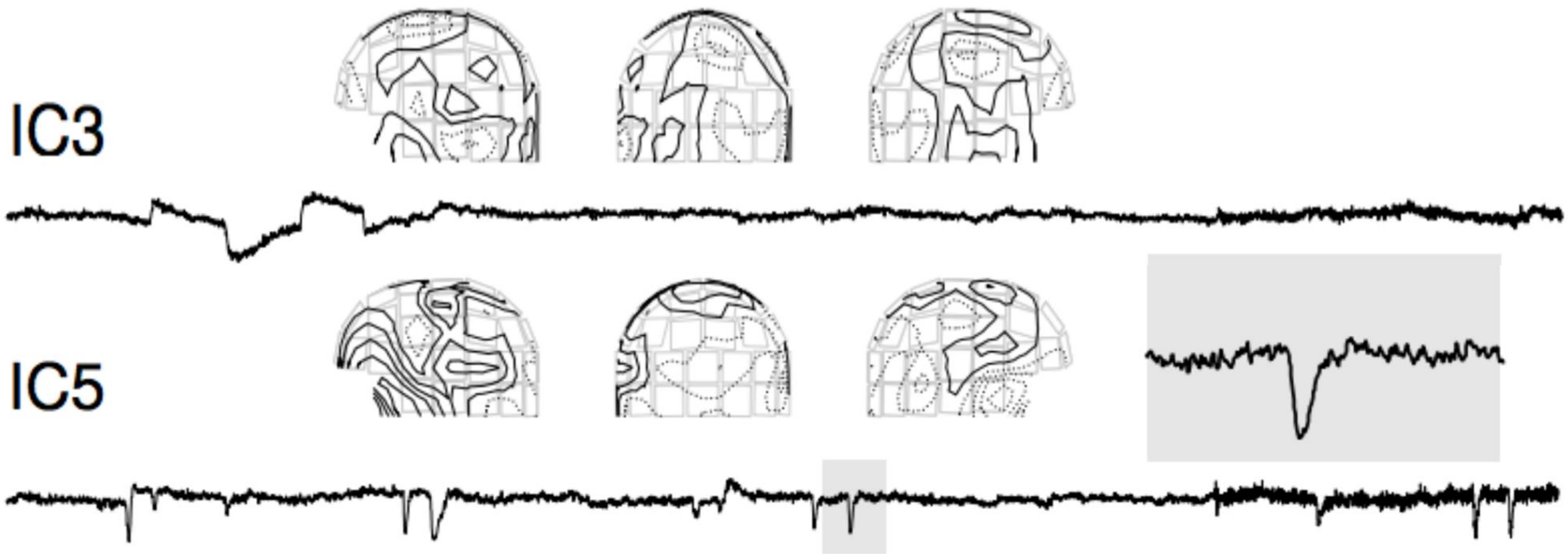
MEG application

There are 9 ICA found from the recorded data



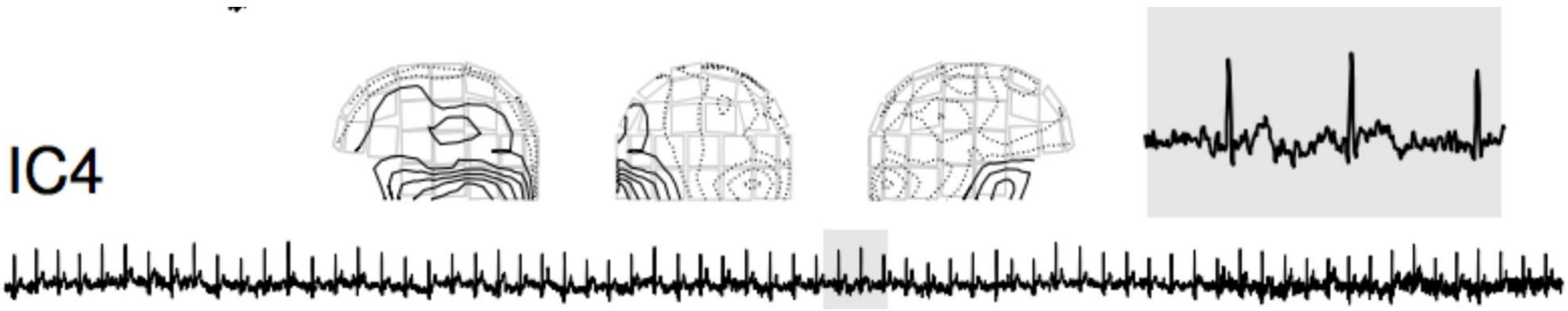
➔ Clearly due to the muscular activity originated for the biting

MEG application



➔ Showing Horizontal eye movement IC3 and the eye blinks IC5

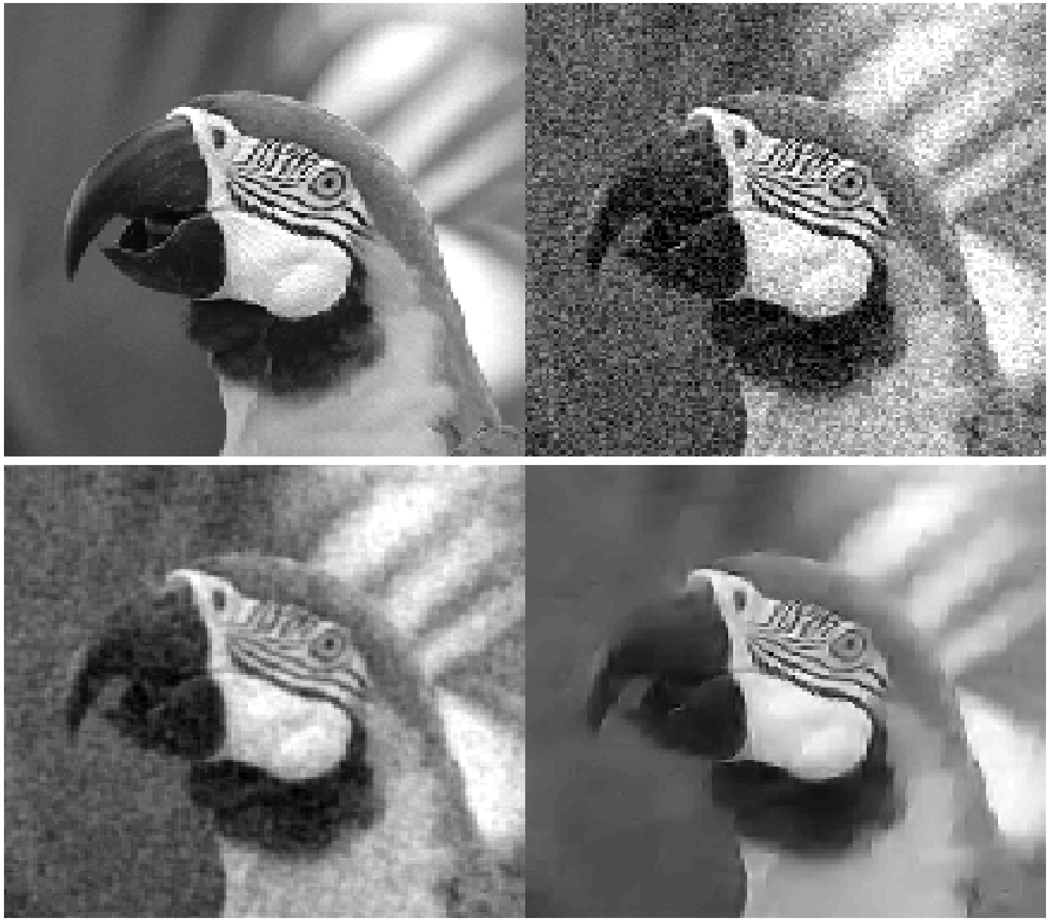
MEG application



➔IC4 is clearly extracted to represents the cardiac artifact.

Image denoise

Original image

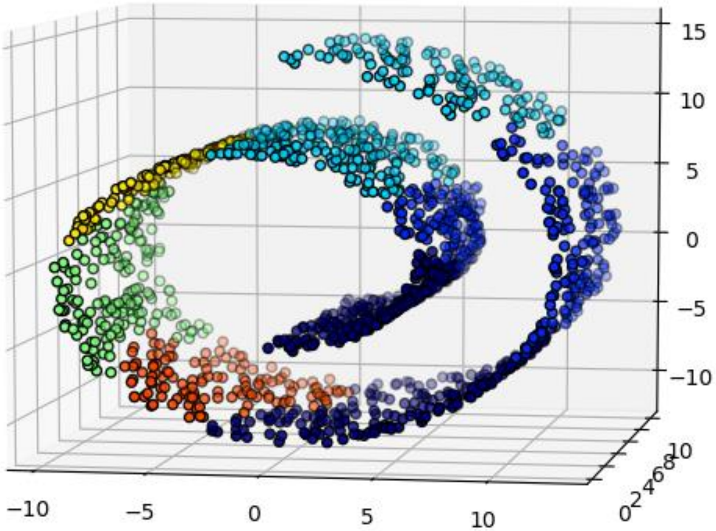


Noisy image

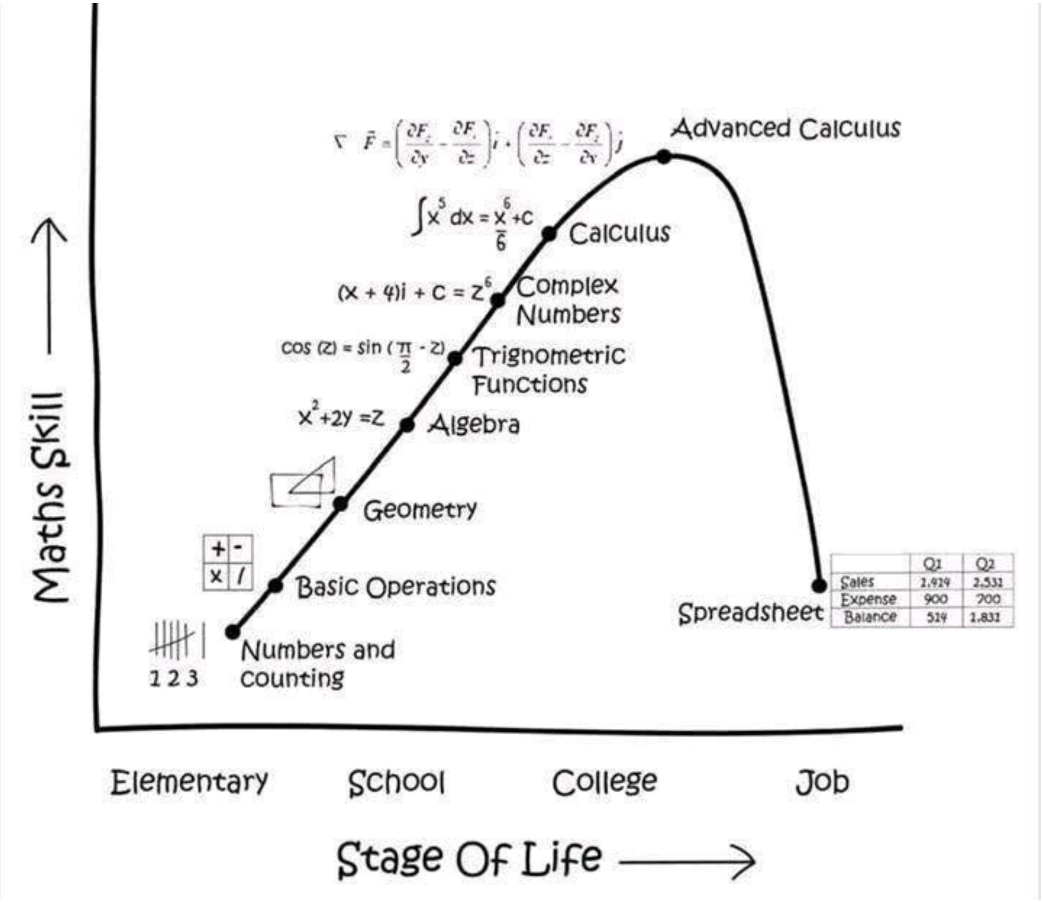
Wiener filtering

ICA filtering

- Is projection always good?
  - Not really! Example: Swiss roll toy dataset
  - **Nonlinear** methods should be considered



Questions?



Maths Skill
Advanced Calculus
Calculus
Complex Numbers
Trignometric Functions
Algebra
Geometry
Basic Operations
Numbers and counting
1 2 3
Spreadsheet
Q1
Q2
Sales
Expense
Balance
Elementary
School
College
Job
Stage Of Life