FRIEDRICH-ALEXANDER
UNIVERSITÄT
ERLANGEN-NÜRNBERG

NATURWISSENSCHAFTLICHE
FAKULTÄT

Clustering via Estimation Maximization (EM Clustering)

Lecture “Mathematical Data Science” 2021/2022

Frauke Liers
Friedrich-Alexander-Universität Erlangen-Nürnberg
October 26, 2021

DISCRETE
OPTIMIZATION

Expectation-Maximization Clustering Algorithm (EM)

- further reading: The Elements of Statistical Learning, Chapter 14
- Recall K-means has implicit assumptions (clusters are convex & roughly equally sized) that may not be satisfied.
- alternative way of thinking: decide for each data point the probability with which it belongs to a certain cluster, i.e., a 'soft' clustering. allows clusters of different size, can detect correlations
- problem: this probability distribution is unknown.
- task: estimate probability distribution, improve estimate iteratively

Mixture of Gaussian Distributions

- make quite general assumption: This unknown distribution is a mixture, i.e., superposition, of K many (multi-dimensional) Gaussian distributions. Means that probability function is of the form

$$\mathbb{P}(x|\underline{p}, \underline{\mu}, \underline{\Sigma}) = \sum_{k=1}^{K} p_k \cdot \mathcal{N}(x|\mu_k, \Sigma_k)$$

- with

$$\begin{aligned} p &= (p_1, \dots, p_K), && p_k \in \mathbb{R} \text{ probability vector} \\ \mu &= (\mu_1, \dots, \mu_K), && \mu_k \in \mathbb{R}^n \text{ vector of means} \\ \Sigma &= (\Sigma_1, \dots, \Sigma_K), && \Sigma_k \in \mathbb{R}^{n\times n} \text{ covariance matrix} \end{aligned}$$

where n is the dimension of a data point x, i.e. $x \in \mathbb{R}^n$.
recall Gaussian distribution in dimension M with mean vector $\mu \in \mathbb{R}^M$,
variance $\Sigma \in \mathbb{R}^{M\times M}$: $\mathcal{N}(x|\mu, \Sigma) = \frac{1}{\sqrt{(2\pi)^M \det(\Sigma)}} \exp(-\frac{1}{2}(x-\mu)^\top \Sigma^{-1}(x-\mu))$

Recall Covariance Matrix

Let $x = (x_1, \dots, x_n)^\top$ be random vector, finite variance and mean. Let covariance matrix $\Sigma = (\Sigma_{x_i,x_j}) \in \mathbb{R}^{n \times n}$ be defined as $\Sigma_{x_i,x_j} = E(x_i - E(x_i))(x_j - E(x_j))$ where E denotes expected values $\mu_X = E(X)$.

- represents important statistical information, in particular correlation between data
- is a real, quadratic, symmetric matrix
- is positive-semidefinite matrix

More Details on Mixture of Gaussians

$\mathbb{P}(x|\underline{p},\underline{\mu},\underline{\Sigma}) = \sum_{k=1}^{K} p_k \cdot \mathcal{N}(x|\mu_k,\Sigma_k)$ Clustering task: Given data points, estimate $\underline{p},\underline{\mu},\underline{\Sigma}$.
Output: Yields for each data point n and for each cluster k an estimated probability that n generated from k.
Assign each data point the mean with highest probability.
Advantage: also clusters are possible that are (partially) contained in each other:

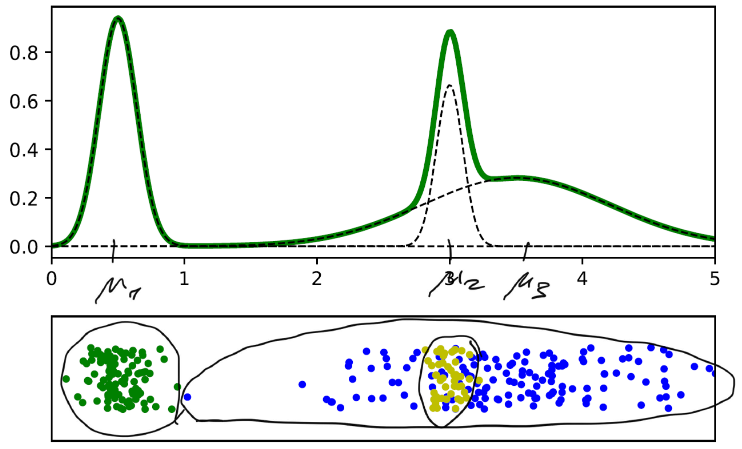


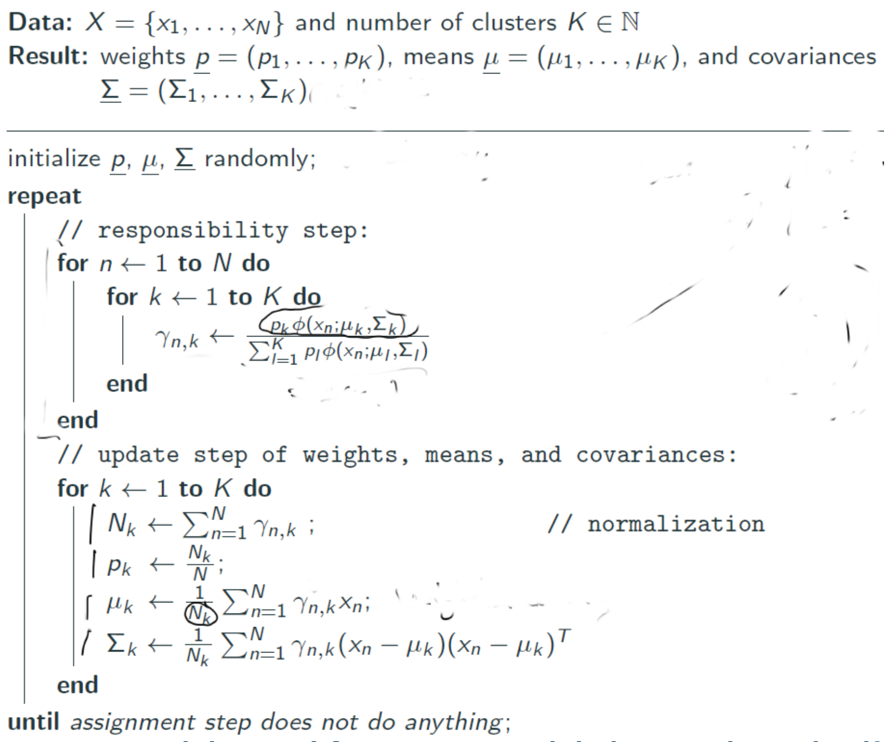
Figure: $K = 3$ Gaussian Mixture in 1d and generated data.

Responsibility Calculation

Define *responsibility of cluster k for x* by observed relative frequencies, i.e., probability that x is generated by Gaussian k.

$$\gamma(x;k) = \frac{p_k \mathcal{N}(x|\mu_k, \Sigma_k)}{\sum_{j=1}^{K} p_j \mathcal{N}(x|\mu_j, \Sigma_j)}$$

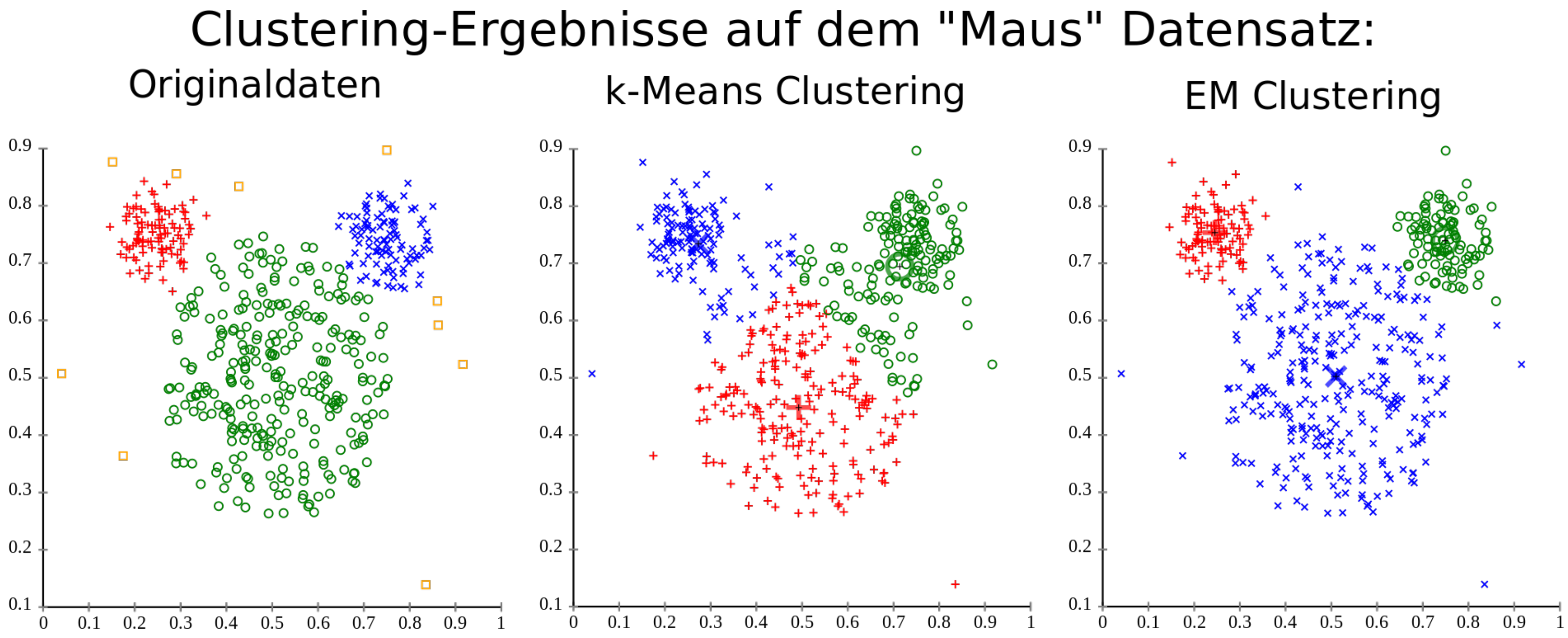
Outline of Expected Maximization Clustering



```
Data: $X = \{x_1, \dots, x_N\}$ and number of clusters $K \in \mathbb{N}$
Result: weights $\underline{p} = (p_1, \dots, p_K)$, means $\underline{\mu} = (\mu_1, \dots, \mu_K)$, and covariances
        $\underline{\Sigma} = (\Sigma_1, \dots, \Sigma_K)$
initialize $\underline{p}$, $\underline{\mu}$, $\underline{\Sigma}$ randomly;
repeat
    // responsibility step:
    for $n \leftarrow 1$ to $N$ do
        for $k \leftarrow 1$ to $K$ do
            $\gamma_{n,k} \leftarrow \frac{p_k \phi(x_n; \mu_k, \Sigma_k)}{\sum_{l=1}^{K} p_l \phi(x_n; \mu_l, \Sigma_l)}$
        end
    end
    // update step of weights, means, and covariances:
    for $k \leftarrow 1$ to $K$ do
        $N_k \leftarrow \sum_{n=1}^{N} \gamma_{n,k}$ ;                    // normalization
        $p_k \leftarrow \frac{N_k}{N}$;
        $\mu_k \leftarrow \frac{1}{N_k} \sum_{n=1}^{N} \gamma_{n,k} x_n$;
        $\Sigma_k \leftarrow \frac{1}{N_k} \sum_{n=1}^{N} \gamma_{n,k} (x_n - \mu_k)(x_n - \mu_k)^T$
    end
until assignment step does not do anything;
```

Compare this to K-means which works similarly. Indeed, K-means is a special case of EM.

A visual comparison



drawback: **K-means and EM only find local optima.** (recall already K-means problem is NP-hard...)
play around with skikit-learn (machine learning library in python)

Hierarchical Clustering Methods

- do not need to specify number of clusters K as input
- need a measure of dissimilarity, e.g., 'distance' between (disjoint) groups of observations, based on pairwise dissimilarities
- clusters at each levels of hierarchy are created by merging clusters at next lower level
- lowest level contains one point each, highest level contains all points
- agglomerative (bottom-up) or divisive (top-down) methods
- e.g., agglomeartive: start at bottom, at each level recursively merge a selected pair of clusters into a single cluster. Next higher level contains one cluster less. Merge those two clusters with smallest dissimilarity. $\Rightarrow N - 1$ levels in hierarchy
- often drawn by rooted binary tree, i.e., contains a node as root, each node has not more than two children in the tree.

Dendrogram

from statistical learning book:

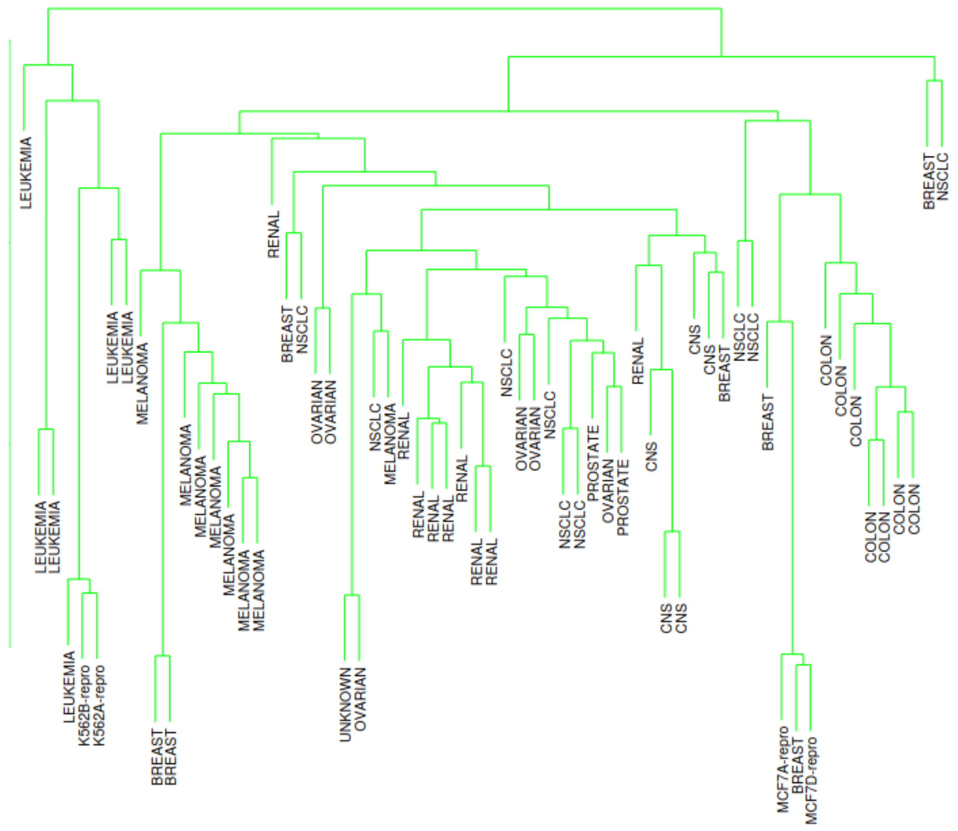


FIGURE 14.12. *Dendrogram from agglomerative hierarchical clustering with average linkage to the human tumor microarray data.*

Dendrogram

- a word of caution: small changes in data can lead to quite different dendrograms
- is hierarchical structure actually intrinsic to the data?

Principal Component Analysis PCA

Principal Component Analysis (PCA):

- first idea by Karl Pearson in 1906
- improvements by Harold Hotelling in the 1930s
- widespread use since raise of **computers**

Applications:

- Multivariate statistics
- Cluster analysis
- Data reduction
- Feature extraction
- Image processing
- ...

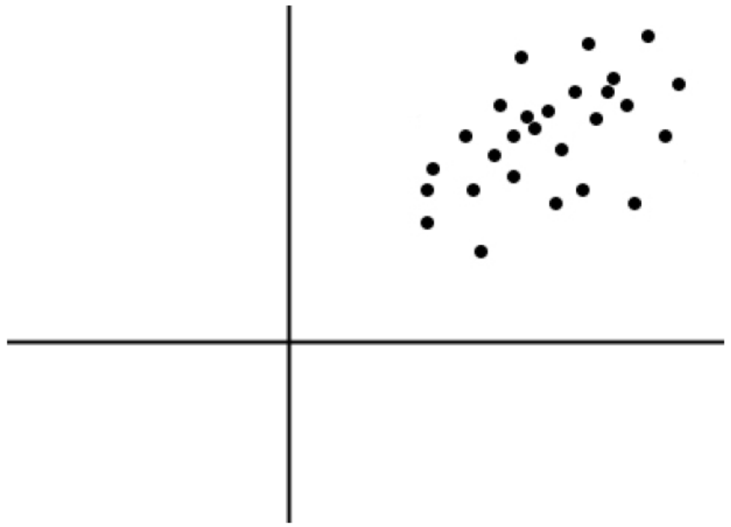
Prelimininaries

- general approach from multivariate statistics
- structures large data sets through eigenvalues and covariances
- represents data through principal components, i.e. linear combinations of statistical variables

What is given?

- input data set with $N \in \mathbb{N}$ points $x^{(1)}, \ldots, x^{(N)} \in \mathbb{R}^M$
- no a-priori knowledge about data needed (e.g., cluster label)
- statistically interpretable as N observations of M random variables.(e.g., we have measured M features for N people / objects.)

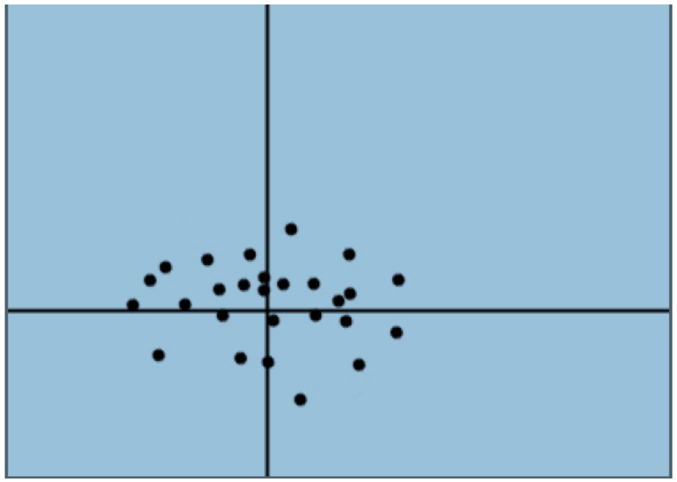
Objectives of PCA



What is the goal?

- Structure identification in the data
- Extraction of meaningful features
- Data reduction to most expressive information, i.e. project data points in

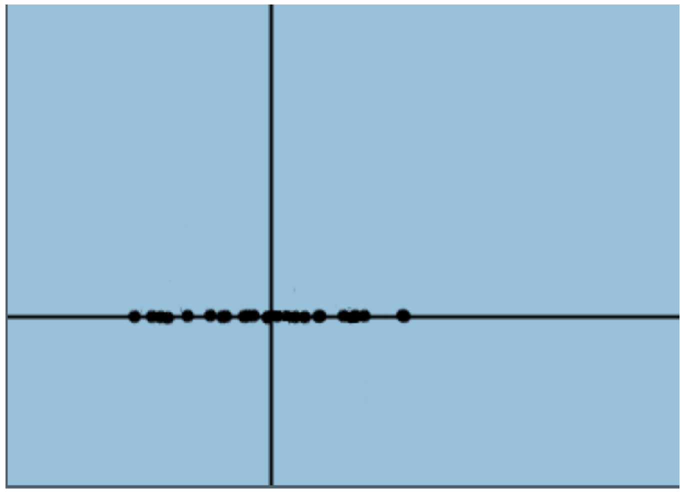
Objectives of PCA



What is the goal?

- Structure identification in the data
- Extraction of meaningful features
- Data reduction to most expressive information, i.e. project data points in k-dimensional space, with $k < M$, such that no or not much information gets lost.

Objectives of PCA



What is the goal?

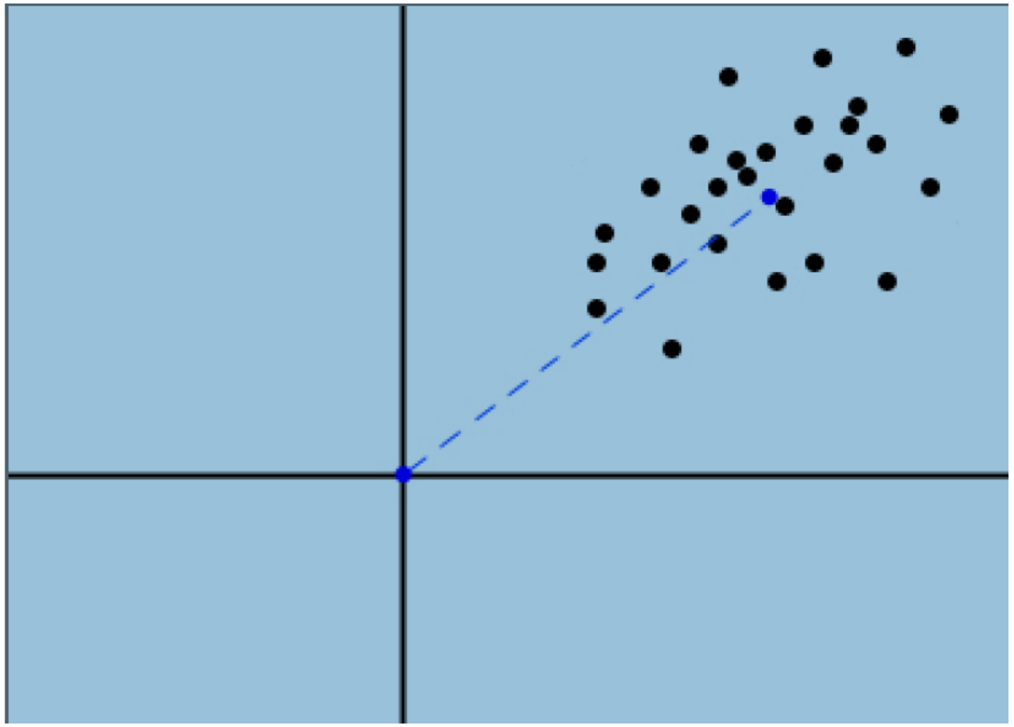
- Structure identification in the data
- Extraction of meaningful features
- Data reduction to most expressive information, i.e. project data points in k-dimensional space, with $k < M$, such that no or not much information gets lost.

Computing PCA: Data centering

Centering the data in the origin

$\rightarrow$ Computation of mean value $\overline{X} = \frac{1}{N}\sum_{i=1}^{N} x^{(i)}$

In the following: $y^{(i)} = x^{(i)} - \overline{X}, i = 1, \ldots, N$ centered data



Computing PCA: Covariance matrix

Computation of covariance matrix $C \in \mathbb{R}^{M \times M}$**:**

$$\boxed{C := \frac{1}{N} \sum_{i=1}^{N} y^{(i)} {y^{(i)}}^T}$$

$$\begin{aligned} C_{k,l} &= \frac{1}{N} \sum_{i=1}^{N} {y_k}^{(i)} {y_l}^{(i)} = \frac{1}{N} \sum_{i=1}^{N} ({y_k}^{(i)} - 0)({y_l}^{(i)} - 0) \\ &= \frac{1}{N} \sum_{i=1}^{N} ({y_k}^{(i)} - \overline{Y_k})({y_l}^{(i)} - \overline{Y_l}) =: Cov(y_k, y_l) \end{aligned}$$

Recall Linear Algebra Lectures: Diagonalisation of C

Aim: Alternative data representation: $y^{(i)} \in \mathbb{R}^M \to z^{(i)} \in \mathbb{R}^k$,

- based on orthogonal vectors ('principal components')
- vectors should be aligned with directions of highest variance
- data representation should be **uncorrelated**
  $\to$ $Cov(z_j, z_l) = 0$ for $j \neq l$
  $\to$ diagonalisation of matrix C

(Finite-dimensional) spectral theorem from Linear Algebra:

Let $C \in \mathbb{R}^{M \times M}$ be a real, symmetric matrix. Then there exists an orthogonal matrix S, such that:

$$S^T C S = \begin{pmatrix} \lambda_1 & & 0 \\ & \ddots & \\ 0 & & \lambda_N \end{pmatrix},$$

for which $\lambda_1, \ldots, \lambda_M \in \mathbb{R}$ are the eigenvalues of C.
The columns of S are orthonormal eigenvectors of C.

Computing PCA: Solving the eigenvalue problem

We get the wanted alternative data representation by computing **eigenvalues** and respective **eigenvectors** of C.
Thus, we need to (numerically) solve the eigenvalue problem:

$$\lambda v = Cv$$

Recall: A solution can be found by various methods :

- roots of characteristic polynomial
- QR algorithm
- Jacobi eigenvalue algorithm
- singular value decomposition
- etc.

Observations:

- C positive semi-definite $\Rightarrow$ only non-negative eigenvalues
- $\lambda_j \equiv$ data variance along direction of eigenvector $v^{(j)}$
- eigenvectors form a new local coordinate system

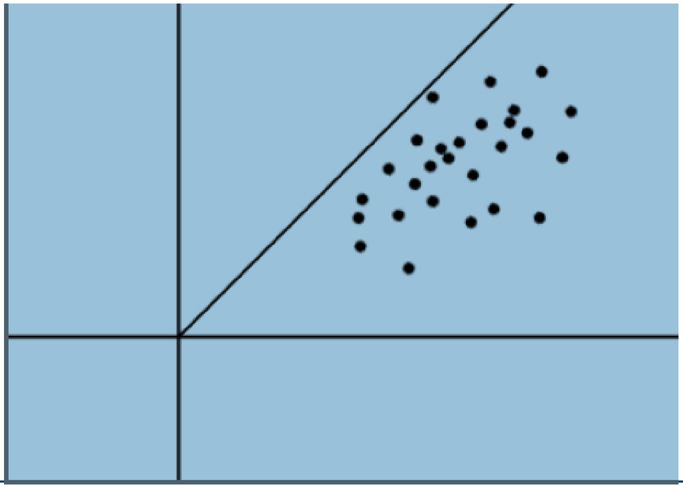
Computing PCA: Solving the eigenvalue problem

A simple example:

- Data is distributed within hyperplane parallel to plane *span*(e_1, e_2)
  $\rightarrow$ no variance in direction e_3 (no depth)
- easy to recognize from eigenvalue λ_3, because $S^T CS$ leads to:

$$D = \begin{pmatrix} \lambda_1 > 0 & 0 & 0 \\ 0 & \lambda_2 > 0 & 0 \\ 0 & 0 & \lambda_3 = 0 \end{pmatrix}$$

- Selection of eigenvalues $\lambda_1, \lambda_2 > 0$ leads to data reduction



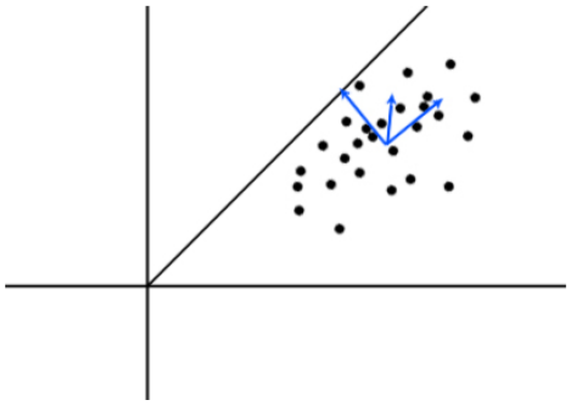
Computing PCA: Solving the eigenvalue problem

A simple example:

- Data is distributed within hyperplane parallel to plane $span(e_1, e_2)$
  $\rightarrow$ no variance in direction e_3 (no depth)
- easy to recognize from eigenvalue λ_3, because $S^T CS$ leads to:

$$D = \begin{pmatrix} \lambda_1 > 0 & 0 & 0 \\ 0 & \lambda_2 > 0 & 0 \\ 0 & 0 & \lambda_3 = 0 \end{pmatrix}$$

- Selection of eigenvalues $\lambda_1, \lambda_2 > 0$ leads to data reduction



FRIEDRICH-ALEXANDER UNIVERSITÄT ERLANGEN-NÜRNBERG
NATURWISSENSCHAFTLICHE FAKULTÄT

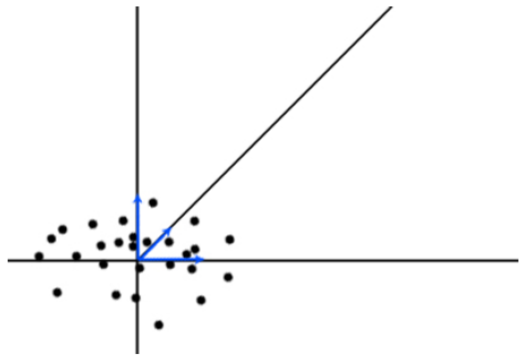
Computing PCA: Solving the eigenvalue problem

A simple example:

- Data is distributed within hyperplane parallel to plane $span(e_1, e_2)$
  $\rightarrow$ no variance in direction e_3 (no depth)
- easy to recognize from eigenvalue λ_3, because $S^T CS$ leads to:

$$D = \begin{pmatrix} \lambda_1 > 0 & 0 & 0 \\ 0 & \lambda_2 > 0 & 0 \\ 0 & 0 & \lambda_3 = 0 \end{pmatrix}$$

- Selection of eigenvalues $\lambda_1, \lambda_2 > 0$ leads to data reduction



The actual PCA

Define transformation matrix:

$$T := (v^{(1)}, \dots, v^{(k)}) \in \mathbb{R}^{M \times k},$$

for which $v^{(1)}, \dots, v^{(k)}$ are the respective eigenvectors of the $1 \leq k \leq M$ largest eigenvalues.

Principal component analysis:

- transform the data: $z^{(i)} := T^T y^{(i)} = T^T(x^{(i)} - \overline{X})$ for $i = 1, \dots, N$
- $z^{(i)} \in \mathbb{R}^k$ contains the most relevant information (features) of the input data
- The components $z_j^{(i)}, j = 1, \dots, k$ are called **principal components**
- If T is quadratic $(k = M) \Rightarrow$ PCA is simply a rotation in $\mathbb{R}^M$

The principal components of the input data are typically used as features in **classification** or **clustering tasks**.

Summary of PCA

For given input data $x^{(1)}, \dots, x^{(N)} \in \mathbb{R}^M$ the PCA can be computed as

The (linear) PCA algorithm

1. Compute mean value of data $\overline{X} = \sum_{i=1}^{N} x^{(i)}$
2. Center data via $y^{(i)} = x^{(i)} - \overline{X}$
3. Compute covariance matrix $C = \frac{1}{N} \sum_{i=1}^{N} y^{(i)} {y^{(i)}}^T$
4. Determine the M eigenvalues and eigenvectors of C numerically
5. Select $1 \leq k \leq M$ respective eigenvectors $v^{(1)}, \dots, v^{(k)}$ of the k largest non-vanishing eigenvalues
6. Assemble selected eigenvectors $v^{(1)}, \dots, v^{(k)}$ columnwise to matrix $T \in \mathbb{R}^{M \times k}$
7. Compute principal components for each centered input point $y^{(i)} \in \mathbb{R}^M$ via:
$$T^T y^{(i)} = z^{(i)} \in \mathbb{R}^k$$

Properties of the PCA

Data reconstruction

Reconstructing the centered input data from its principal components is (partially) possible via:

$$Tz^{(i)} = \tilde{y}^{(i)}, \text{ for } i = 1, \ldots, N$$

It is clear that $\tilde{y}^{(i)} = y^{(i)}$ iff $\lambda_j = 0$ for $k < j < M$

Otherwise: Loss of information

Additional problems:

- computational complexity is: $\mathcal{O}(M^3)$ for eigenvalue decomposition + $\mathcal{O}(NM^2)$ for calculation of covariance matrix
  $\rightarrow$ numerically expensive for large M (dimension of data space)
- number of principal components (and hence possible features) is bounded by M
  example: $x \in \mathbb{R}^2 \Rightarrow$ max. two principal components
- (linear) PCA does not allow for extraction of nonlinear features

Conclusions and Outlook

- PCA is a good tool for data reduction and feature extraction
- can be interpreted as **linear transformation** of input data to a feature space
- computational complexity mainly depends on dimension M of data space
- PCA is restricted to linear features

Thank you for your attention!