INF 4300 – Digital Image Analysis

REPETITION

Classification

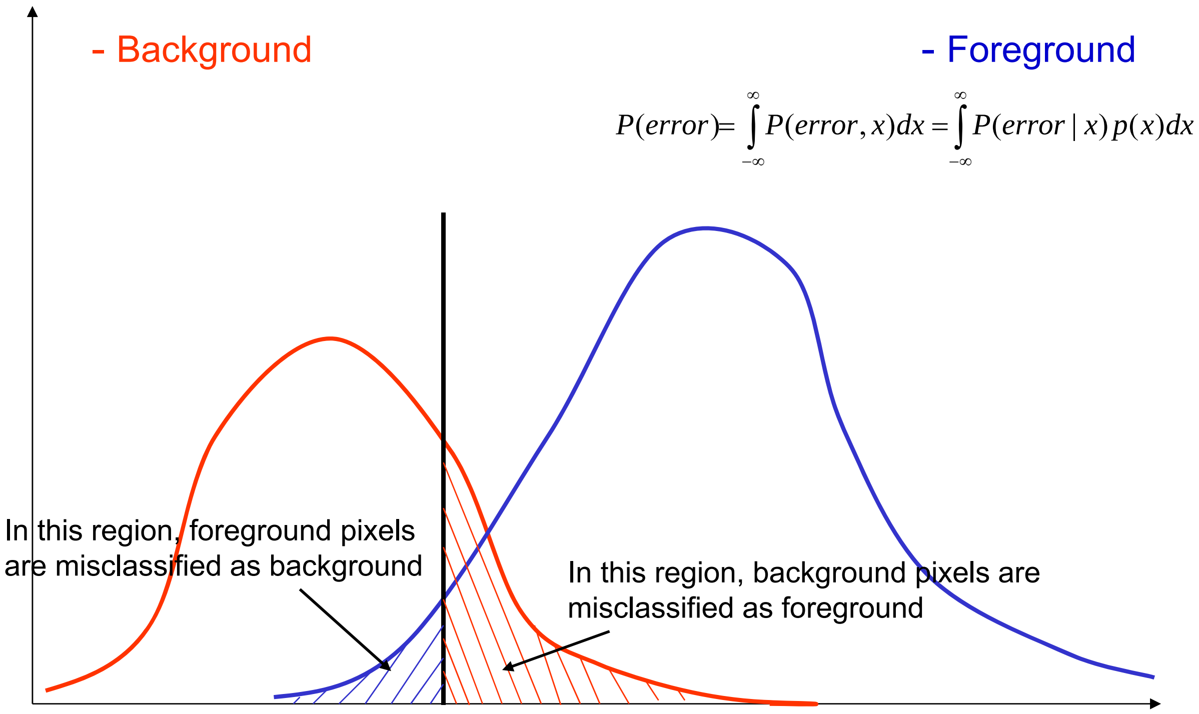
PCA and Fisher's linear discriminant

Morphology

Segmentation

Anne Solberg 14.11.2016

Back to classification error for thresholding



$$P(error) = \int_{-\infty}^{\infty} P(error, x)dx = \int_{-\infty}^{\infty} P(error \mid x)p(x)dx$$

In this region, foreground pixels are misclassified as background

In this region, background pixels are misclassified as foreground

Minimizing the error

$$P(error) = \int_{-\infty}^{\infty} P(error, x)dx = \int_{-\infty}^{\infty} P(error \mid x)\, p(x)dx$$

- When we derived the optimal threshold, we showed that the minimum error was achieved for placing the threshold (or *decision boundary* as we will call it now) at the point where

$$P(\omega_1|x) = P(\omega_2|x)$$

- This is still valid.

Discriminant functions

- The decision rule

$$\text{Decide } \omega_1 \text{ if } \mathrm{P}(\omega_1 \mid \mathbf{x}) > P(\omega_\mathrm{j} \mid \mathbf{x}), \text{for all } j \neq i$$

can be written as assign **x** to ω_1 if

$$g_i(\mathbf{x}) > g_j(\mathbf{x})$$

- The classifier computes J discriminant functions g_i(**x**) and selects the class corresponding to the largest value of the discriminant function.
- Since classification consists of choosing the class that has the largest value, a scaling of the discriminant function g_i(**x**) by f(g_i(**x**)) will not effect the decision if f is a monotonically increasing function.
- This can lead to simplifications as we will soon see.

Equivalent discriminant functions

- The following choices of discriminant functions give equivalent decisions:

$$g_i(\mathbf{x}) = P(\omega_i \mid \mathbf{x}) = \frac{p(\mathbf{x} \mid \omega_i)P(\omega_i)}{p(\mathbf{x})}$$

$$g_i(\mathbf{x}) = p(\mathbf{x} \mid \omega_i)P(\omega_i)$$

$$g_i(\mathbf{x}) = \ln p(\mathbf{x} \mid \omega_i) + \ln P(\omega_i)$$

- The effect of the decision rules is to divide the feature space into c decision regions R_1,......R_c.
- If $g_i(\mathbf{x})>g_j(\mathbf{x})$ for all $j\neq i$, then **x** is in region R_i.
- The regions are separated by decision boundaries, surfaces in features space where the discriminant functions for two classes are equal

The conditional density p(**x**| ω_s)

- Any probability density function can be used to model p(**x**| ω_s)
- A common model is the multivariate Gaussian density.
- The multivariate Gaussian density:

$$p(\mathbf{x} \mid \omega_s) = \frac{1}{(2\pi)^{n/2}\left|\mathbf{\Sigma}_s\right|^{1/2}} \exp\left[-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu}_s)^t \mathbf{\Sigma}_s^{-1}(\mathbf{x}-\boldsymbol{\mu}_s)\right]$$

- If we have d features, μ_s is a vector of length d and and Σ_s a d×d matrix (depends on class *s*)

$$\boldsymbol{\mu}_S = \begin{bmatrix} \mu_{1s} \\ \mu_{2s} \\ \\ \\ \mu_{ns} \end{bmatrix} \qquad \mathbf{\Sigma}_S = \begin{bmatrix} \sigma_{11} & \sigma_{12} & . & . & \sigma_{1n} \\ \sigma_{21} & \sigma_{22} & . & . & . \\ \sigma_{31} & \sigma_{11} & . & . & . \\ . & . & . & . & . \\ \sigma_{n1} & \sigma_{n2} & . & \sigma_{nn-1} & \sigma_{nn} \end{bmatrix}$$

Symmetric d×d matrix
σ_{ii} is the variance of feature i
σ_{ij} is the covariance between feature i and feature j
Symmetric because $\sigma_{ij} = \sigma_{ji}$

- $|\Sigma_s|$ is the determinant of the matrix Σ_s, and Σ_s^{-1} is the inverse

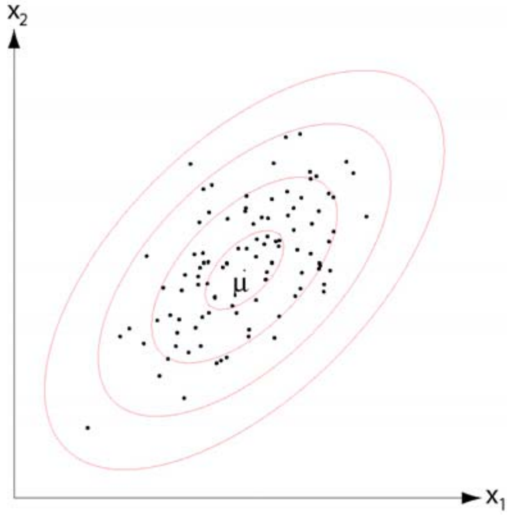
The covariance matrix and ellipses

- In 2D, the Gaussian model can be thought of as approximating the classes in 2D feature space with ellipses.
- The mean vector $\mu=[\mu_1, \mu_2]$ defines the the center point of the ellipses.
- σ_{12}, the covariance between the features defines the orientation of the ellipse.
- σ_{11} and σ_{22} defines the width of the ellipse.

$$\Sigma_S = \begin{bmatrix} \sigma_{11} & \sigma_{12} \\ \sigma_{21} & \sigma_{22} \end{bmatrix}$$

- The ellipse defines points where the probability density is equal
  - Equal in the sense that the distance to the mean as computed by the Mahalanobis distance is equal.
  - The Mahalanobis distance between a point x and the class center μ is:

$$r^2 = (x-\mu)^T \Sigma^{-1} (x-\mu)$$



The main axes of the ellipse is determined by the eigenvectors of Σ. The eigenvalues of Σ gives their length.

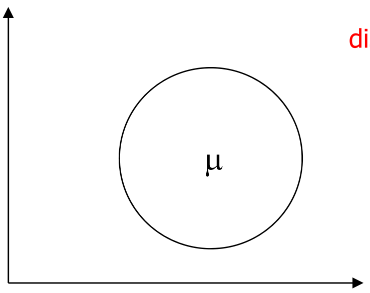
Euclidean distance vs. Mahalanobis distance

- Euclidean distance between point x and class center μ:

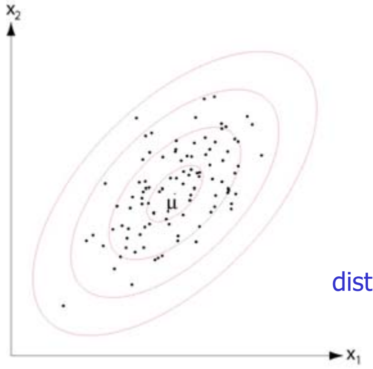
$$(x-\mu)^T (x-\mu) = \|x-\mu\|^2$$

- Mahalanobis distance between x and μ:

$$r^2 = (x-\mu)^T \Sigma^{-1} (x-\mu)$$



Points with equal distance to μ lie on a circle.



Points with equal distance to μ lie on an ellipse.

Discriminant functions for the normal density

- We saw last lecture that the minimum-error-rate classification can be computed using the discriminant functions

$$g_i(\mathbf{x}) = \ln p(\mathbf{x} \,|\, \omega_i) + \ln P(\omega_i)$$

- With a multivariate Gaussian we get:

$$g_i(\mathbf{x}) = -\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu}_i)^t \Sigma_i^{-1}(\mathbf{x}-\boldsymbol{\mu}_i) - \frac{d}{2}\ln 2\pi - \frac{1}{2}\ln|\Sigma_i| + \ln P(\omega_i)$$

- Let ut look at this expression for some special cases:

Case 1: $\Sigma_j=\sigma^2 I$

- The discriminant functions simplifies to **linear** functions using such a shape on the probability distributions

$$g_j(\mathbf{x}) = -\frac{1}{2(\sigma^2 I)}(\mathbf{x}-\boldsymbol{\mu}_j)^T(\mathbf{x}-\boldsymbol{\mu}_j) - \frac{d}{2}\ln(2\pi) - \frac{1}{2}\ln\left|\sigma^2 I\right| + \ln P(\omega_j)$$

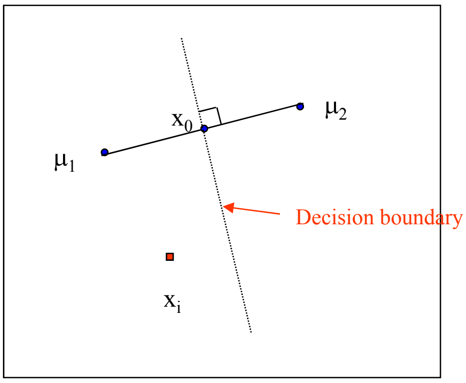
$$= -\frac{1}{2(\sigma^2 I)}(\mathbf{x}^T\mathbf{x} - 2\boldsymbol{\mu}_j^T\mathbf{x} + \boldsymbol{\mu}_j^T\boldsymbol{\mu}_j) - \frac{d}{2}\ln(2\pi) - \frac{1}{2}\ln\left|\sigma^2 I\right| + \ln P(\omega_j)$$

Common for all classes, no need to compute these terms

Since $\boldsymbol{x}^T\boldsymbol{x}$ is common for all classes, an equivalent $g_j(\boldsymbol{x})$ is a linear function of ***x*:** .

$$\frac{1}{(\sigma^2)}\boldsymbol{\mu}_j^T\mathbf{x} - \frac{1}{2(\sigma^2)}\boldsymbol{\mu}_j^T\boldsymbol{\mu}_j + \ln P(\omega_j)$$

- The discriminant function (when $\Sigma_j=\sigma^2 I$) that defines the border between class 1 and 2 in the feature space is a straight line.
- The discriminant function intersects the line connecting the two class means at the point $x_0=(\mu_1- \mu_2)/2$ (if we do not consider prior probabilities).
- The discriminant function will also be normal to the line connecting the means.



Case 2: Common covariance, $\Sigma_j= \Sigma$

- An equivalent formulation of the discriminant functions is

$$
\begin{aligned}
&g_i(\mathbf{x}) = \mathbf{w}_i^t\mathbf{x} + wi_0 \\
&\text{where } \mathbf{w}_i = \boldsymbol{\Sigma}^{-1}\boldsymbol{\mu}_i \\
&\text{and } wi_0 = -\frac{1}{2}\boldsymbol{\mu}_i^t\boldsymbol{\Sigma}^{-1}\boldsymbol{\mu}_i + \ln P(\omega_i)
\end{aligned}
$$

- The decision boundaries are again hyperplanes.
- The decision boundary has the equation:

$$
\begin{aligned}
&\mathbf{w}^T(\mathbf{x}-\mathbf{x}_0) = 0 \\
&\mathbf{w} = \Sigma^{-1}(\mu_i - \mu_j) \\
&x_0 = \frac{1}{2}(\mu_i + \mu_j) - \frac{\ln\left[(P(\omega_i)/(P(\omega_j)\right]}{(\mu_i - \mu_j)^T\Sigma^{-1}(\mu_i - \mu_j)}(\mu_i - \mu_j)
\end{aligned}
$$

- Because $\mathbf{w}_i= \boldsymbol{\Sigma}^{-1}(\mu_i- \mu_j)$ is not in the direction of $(\mu_i- \mu_j)$, the hyperplane will not be orthogonal to the line between the means.

Case 3:, Σ_j=arbitrary

- The discriminant functions will be quadratic:

$$g_i(\mathbf{x}) = \mathbf{x}^t \mathbf{W}_i \mathbf{x} + \mathbf{w}_i^t \mathbf{x} + wi_0$$

$$\text{where } \mathbf{W}_i = -\frac{1}{2}\mathbf{\Sigma}_i^{-1}, \qquad \mathbf{w}_i = \mathbf{\Sigma}_i^{-1}\boldsymbol{\mu}_i$$

$$\text{and } wi_0 = -\frac{1}{2}\boldsymbol{\mu}_i{}^t \mathbf{\Sigma}_i^{-1}\boldsymbol{\mu}_i - \frac{1}{2}\ln|\mathbf{\Sigma}_i| + \ln P(\omega_i)$$

- The decision surfaces are hyperquadrics and can assume any of the general forms:
  - hyperplanes
  - hypershperes
  - pairs of hyperplanes
  - hyperellisoids,
  - Hyperparaboloids,..
- The next slides show examples of this.
- In this general case we cannot intuitively draw the decision boundaries just by looking at the mean and covariance.

Distance measures used in feature selection

- In feature selection, each feature combination must be ranked based on a criterion function.
- Criteria functions can either be distances between classes, or the classification accuracy on a validation test set.
- If the criterion is based on e.g. the mean values/covariance matrices for the training data, distance computation is fast.
- Better performance at the cost of higher computation time is found when the classification accuracy on a validation data set (different from training and testing) is used as criterion for ranking features.
  - This will be slower as classification of the validattion data needs to be done for every combination of features.

Method 2 - Sequential backward selection

- Select l features out of d
- Example: 4 features x_1,x_2,x_3,x_4
- Choose a criterion C and compute it for the vector $[x_1,x_2,x_3,x_4]^T$
- Eliminate one feature at a time by computing $[x_1,x_2,x_3]^T$, $[x_1,x_2,x_4]_T$, $[x_1,x_3,x_4]^T$ and $[x_2,x_3,x_4]^T$
- Select the best combination, say $[x_1,x_2,x_3]^T$.
- From the selected 3-dimensional feature vector eliminate one more feature, and evaluate the criterion for $[x_1,x_2]^T$, $[x_1,x_3]_T$, $[x_2,x_3]^T$ and select the one with the best value.
- Number of combinations searched:

  1+1/2((d+1)d-l(l+1))

Method 3: Sequential forward selection

- Compute the criterion value for each feature. Select the feature with the best value, say x_1.
- Form all possible combinations of features x1 (the winner at the previous step) and a new feature, e.g. $[x_1,x_2]^T$, $[x_1,x_3]^T$, $[x_1,x_4]^T$, etc. Compute the criterion and select the best one, say $[x_1,x_3]^T$.
- Continue with adding a new feature.
- Number of combinations searched: ld-l(l-1)/2.
  - Backwards selection is faster if l is closer to d than to 1.

k-Nearest-Neighbor classification

- A very simple classifier.
- Classification of a new sample x_i is done as follows:
  - Out of N training vectors, identify the *k* nearest neighbors (measured by Euclidean distance) in the training set, irrespectively of the class label.
  - Out of these *k* samples, identify the number of vectors k_i that belong to class ω_i , *i:1,2,....M* (if we have *M* classes)
  - Assign x_i to the class ω_i with the maximum number of k_i samples.
- *k* should be odd, and must be selected a priori.

K-means clustering

- Note: K-means algorithm normally means ISODATA, but different definitions are found in different books
- K is assumed to be known

1. Start with assigning K cluster centers
   - k random data points, or the first K points, or K equally spaces points
   - For k=1:K, Set μ_k equal to the feature vector x_k for these points.
2. Assign each object/pixel x_i in the image to the closest cluster center using Euclidean distance.
   - Compute for each sample the distance r2 to each cluster center:

$$r^2 = (x_i - \mu_k)^T (x_i - \mu_k) = \|x_i - \mu_k\|^2$$

   - Assign x_i to the closest cluster (with minimum *r* value)
3. Recompute the cluster centers based on the new labels.
4. Repeat from 2 until #changes<limit.

ISODATA K-means: splitting and merging of clusters are included in the algorithm

INF 4300
Linear feature transforms

Anne Solberg (anne@ifi.uio.no)

Today:

- Feature transformation through principal component analysis
- Fisher's linear discriminant function

Definitions: Correlation matrix vs. covariance matrix

- Σ_x is the covariance matrix of x

$$\Sigma_X = E\left[(x-\mu)(x-\mu)^T\right]$$

- R_x is the correlation matrix of x

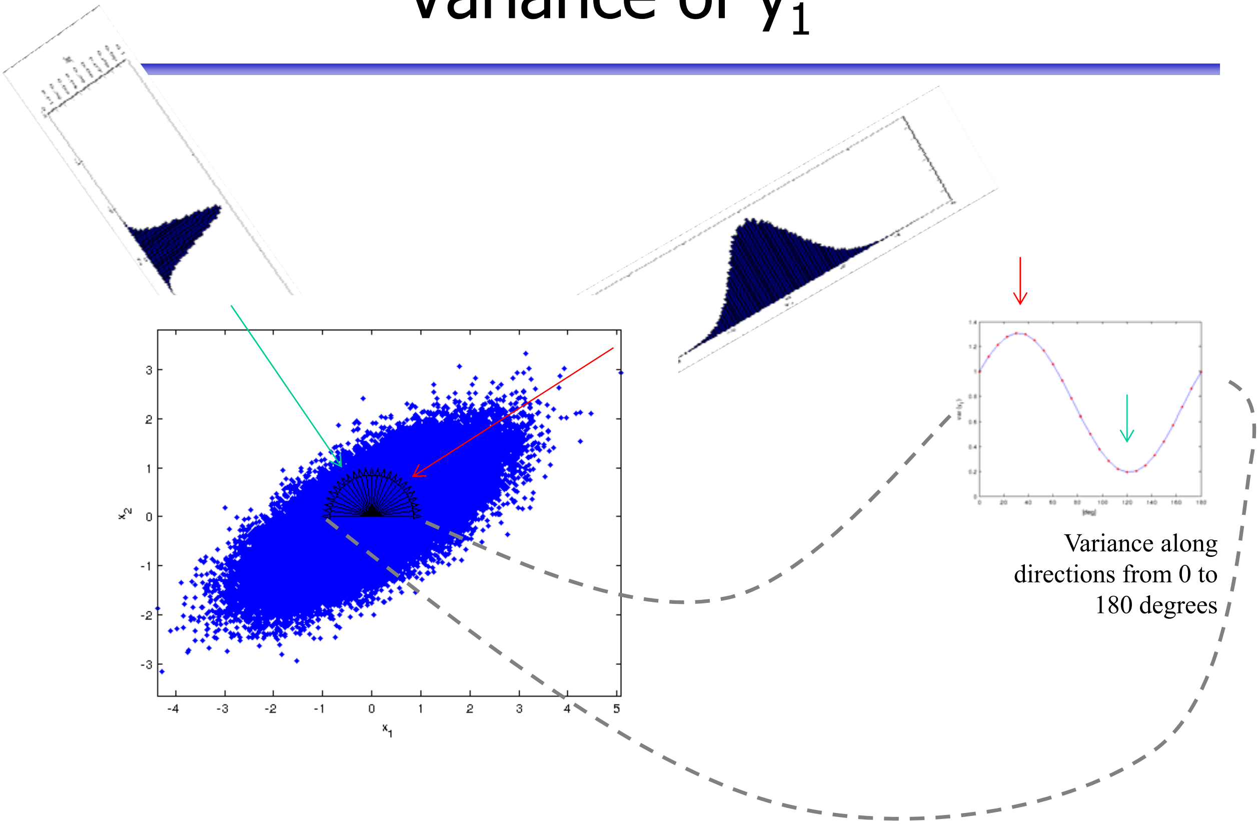
$$R_X = E\left[(x)(x)^T\right]$$

- $R_x=\Sigma_x$ if $\mu_x=0$.

Principal component or Karhunen-Loeve transform

- Let x be a feature vector.
- Features are often correlated, which might lead to redundancies.
- We now derive a transform which yields **uncorrelated** features.
- We seek a linear transform $y=A^Tx$, and the y_is should be uncorrelated.
- The y_is are uncorrelated if $E[y(i)y(j)^T]=0$, $i \neq j$.
- If we can express the information in x using uncorrelated features, we might need **fewer** coefficients.

Variance of y_1



Variance along directions from 0 to 180 degrees

Variance of y_1 cont.

- Assume mean of **x** is subtracted

$$\sigma_{y1}^2 = \frac{1}{N}\sum_i y_i^2$$
$$= \frac{1}{N}\sum_i (\mathbf{w}^T\mathbf{x}_i)^2 = \frac{1}{N}\sum_i \mathbf{w}^T\mathbf{x}_i\mathbf{x}_i^T\mathbf{w} = \mathbf{w}^T\underbrace{(\frac{1}{N}\sum_i \mathbf{x}_i\mathbf{x}_i^T)}_{\text{The sample covariance matrix / scatter matrix; }\mathbf{R}}\mathbf{w}$$
$$= \mathbf{w}^T\mathbf{R}\mathbf{w}$$

Called σ^2_w on some slides

Criterion function

- Goal: Find transform minimizing representation error
- We start with a single weight-vector, **w**, giving us a single feature, y_1
- Let $J(\mathbf{w}) = \mathbf{w}^T\mathbf{R}\mathbf{w} = \sigma_w^2$
- Now, let's find $$\max_{\boldsymbol{w}} J(\boldsymbol{w}) \quad s.t. \,||\boldsymbol{w}|| = 1$$

As we learned on the previous slide, maximizing this is equivalent to minimizing representation error

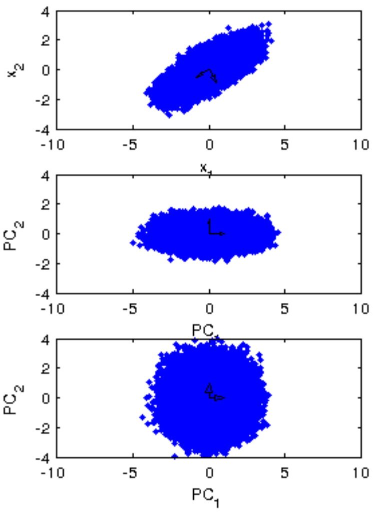
Principal component transform (PCA)

- Place the *m* «principle» eigenvectors (the ones with the largest eigenvalues) along the columns of A
- Then the transform **y** = $\mathbf{A}^T\mathbf{x}$ gives you the *m* first principle components
- The *m*-dimensional **y**
  - have uncorrelated elements
  - retains as much variance as possible
  - gives the best (in the mean-square sense) description of the original data (through the «image»/projection/reconstruction **Ay**)

Note: The eigenvectors themselves can often give interesting information

PCA is also known as Karhunen-Loeve transform

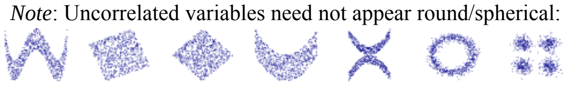
PCA and rotation and «whitening»



If we use all eigenvectors in the transform, **y** = $\mathbf{A}^t\mathbf{x}$, we simply rotate our data so that our new features are uncorrelated, i.e., cov(**y**) is a diagonal matrix.

If we as a next step scale each feature by their σ^{-1}, **y** = $\mathbf{D}^{(-1/2)}\mathbf{A}^t\mathbf{x}$, where **D** is a diagonal matrix of eigenvalues (i.e., variances), we get cov(**y**)=**I**. We say that we have «whitened» the data.

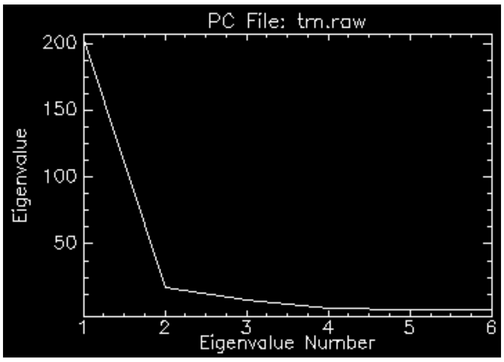
Note: Uncorrelated variables need not appear round/spherical:



Example cont: Inspecting the eigenvalues

The mean-square representation error we get with m of the N PCA-components is given as

$$E\left[\|x-\hat{x}\|^2\right] = \sum_{i=1}^{N-1}\lambda_i - \sum_{i=1}^{m}\lambda_i = \sum_{i=m}^{N-1}\lambda_i$$

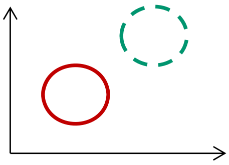


Plotting λ s will give indications on how many features are needed for representation

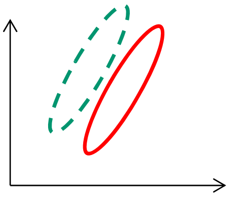
PCA and classification

- Reduce overfitting by detecting directions/components without any/very little variance

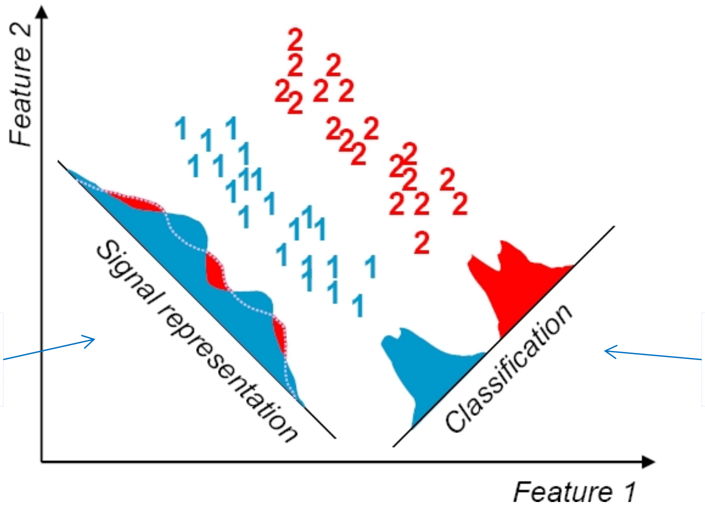
- Sometimes high variation means useful features for classification:



- .. and sometimes not:



Intro to Fisher's linear discriminant



PCA
(unsupervised)

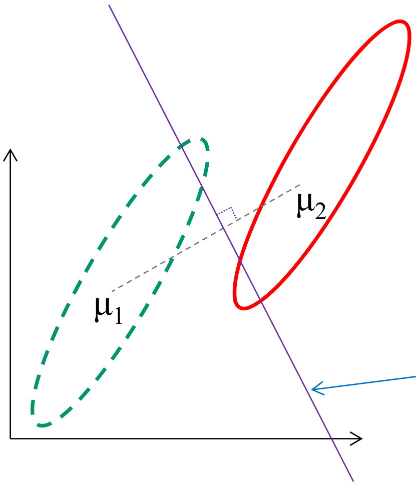
Fisher's LDA
(supervised)

Criterion function - a first attempt

- To find a good projection vector for classification, we need to define a measure of separation between the projections. This will be the criterion function $J(\boldsymbol{w})$
- A naive choice would be projected mean difference, $J(w) = |\tilde{\mu}_1 - \tilde{\mu}_2|^2$, s.t. $|\mathbf{w}|=1$.

This criterion does not consider variance in **y**.

Optimal only when cov(**x**) = $\sigma^2\mathbf{I}$ for all classes (then var(**y**) does not change with **w**).



w simply becomes a scaled difference in means (μ_1-μ_2)

Decision line (not optimal!)

A criterion function including variance

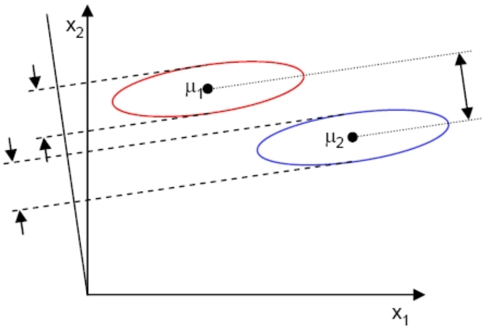
- Fisher's solution: Maximize a function that represents the difference between the means, scaled by a measure of the within-class scatter
- Define classwise scatter (scaled variance)

$$\tilde{s}_i^2 = \sum_{y \in \omega_i} (y - \tilde{\mu}_i)^2$$

- $\tilde{s}_1^2 + \tilde{s}_2^2$ is *within class scatter*
- Fisher's criterion is then

$$J(\mathbf{w}) = \frac{|\tilde{\mu}_1 - \tilde{\mu}_2|^2}{\tilde{s}_1^2 + \tilde{s}_2^2}$$

- We look for a projection where examples from the same class are close to each other, while at the same time projected mean values are as far apart as possible



Scatter matrices – M classes

- Within-class scatter matrix:

$$S_w = \sum_{i=1}^{M} P(\omega_i) S_i$$

$$S_i = E\big[(x-\mu_i)(x-\mu_i)_T\big]$$

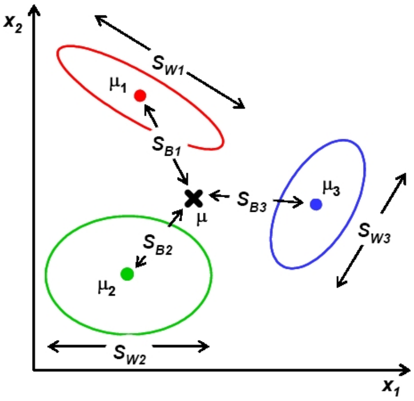
Weighted average of each class' sample covariance matrix

- Between-class scatter matrix:

$$S_b = \sum_{i=1}^{M} P(\omega_i)(\mu_i - \mu)(\mu_i - \mu)^T$$

$$\mu = \sum_{i=1}^{M} P(\omega_i)\mu_i$$

Sample covariance matrix for the means



Fisher criterion in terms of within-class and between-class scatter matrices:

$$J(\mathbf{w}) = \frac{\mathbf{w}^T \mathbf{S}_b \mathbf{w}}{\mathbf{w}^T \mathbf{S}_w \mathbf{w}}$$

Solving Fisher more directly

- Alternatively, you can notice that

$$J(\mathbf{w}) = \frac{\mathbf{w}^T \mathbf{S}_b \mathbf{w}}{\mathbf{w}^T \mathbf{S}_w \mathbf{w}}$$

- .. is a «generalized Rayleigh quotient» and look up the solution for its maximum, which is the principal eigenvector of

$$\boxed{\mathbf{S}_w^{-1}\mathbf{S}_b}$$

- The following solutions (orthogonal in $\mathbf{S}_{w,}$ i.e., $\mathbf{w}_i^T\mathbf{S}_w\mathbf{w}_j=0$, for $i \neq j$) are the next principal eigenvectors

Note that the obtained **w**s are identical (up to scaling) to those from the two-step procedure from the previous slides

Computing Fishers linear discriminant

- For l=M-1:
  - Form a matrix C such that its columns are the M-1 eigenvectors of $S_{xw}^{-1}S_{xb}$
  - Set

    $\hat{y} = C^T x$
  - This gives us the maximum J_3 value.
  - This means that we can reduce the dimension from m to M-1 without loss in class separability power (but only if J_3 is a correct measure of class separability.)
  - Alternative view: with a Bayesian model we compute the probabilities $P(\omega_i|x)$ for each class (i=1,...M). Once M-1 probabilities are found, the remaining $P(\omega_M|x)$ is given because the $P(\omega_i|x)$'s sum to one.

Computation: Case 2: l<M-1

- Form C by selecting the eigenvectors corresponding to the l largest eigenvalues of

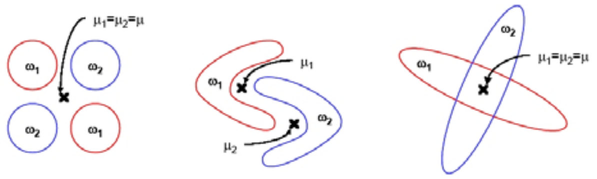
$$S_{xw}^{-1} S_{xb}$$

- We now have a loss of discriminating power since

$$J_{3,\hat{y}} < J_{3,x}$$

Limitations of Fisher's discriminant

- Its criterion function is based on all classes having a similarly-shaped Gaussian distribution
  - Any deviance from this could lead to problems / suboptimal or poor solutions
- It produces at most M-*1* (meaningful) feature projections
- One could «overfit» $\mathbf{S}_w$
- It will fail when the discriminatory information is not in the mean but in the variance of the data (failing to meet that stated in the first bulletpoint!)



INF 4300 – Digital Image Analysis

MORPHOLOGICAL IMAGE PROCESSING

Anne Solberg 31.10.2016

Opening

- Erosion of an image removes all structures that the structuring element cannot fit inside, and shrinks all other structures.
- Dilating the result of the erosion with the same structuring element, the structures that survived the erosion (were shrunken, not deleted) will be restored.
- This is called morphological opening:

$$f \circ S = \left(f \,\theta\, \mathrm{S} \right) \oplus S$$

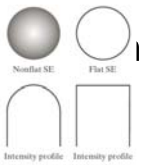
- The name tells that the operation can create an opening between two structures that are connected only in a thin bridge, without shrinking the structures (as erosion would do).

Closing

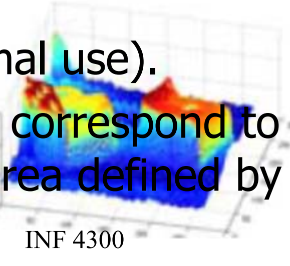
- A dilation of an object grows the object and can fill gaps.
- If we erode the result with the rotated structuring element, the objects will keep their structure and form, but small holes filled by dilation will not appear.
- Objects merged by the dilation will not be separated again.
- Closing is defined as $f \bullet S = \left(f \oplus \hat{S}\right) \theta \, \hat{S}$
- This operation can close gaps between two structures without growing the size of the structures like dilation would.

Gray level morphology

- We apply a simplified definition of morphological operations on gray level images
  - Grey-level erosion, dilation, opening, closing

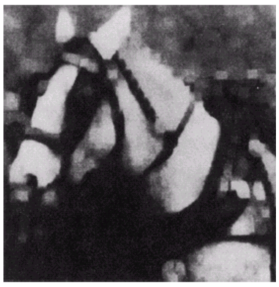


- Image f(x,y)
- Structuring element b(x,y)
  - May be nonflat or flat
- Assume symmetric, flat structuring element, origo at center (this is sufficient for normal use).
- Erosion and dilation then correspond to local minimum and maximum over the area defined by the structuring element

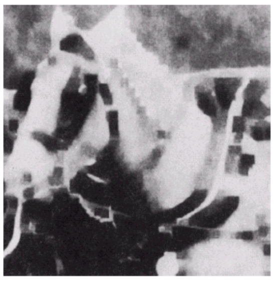


Gray level opening and closing

- Corresponding definition as for binary opening and closing
- Result in a filter effect on the intensity
- Opening: Bright details are smoothed
- Closing: Dark details are smoothed



$$f \circ S = \left(f \,\theta\, \mathrm{S}\right) \oplus S$$
$$= \max(\min(f))$$



$$f \bullet S = \left(f \oplus \mathrm{S}\right) \theta\, S$$
$$= \min(\max(f))$$

Top-hat transformation

- Purpose: detect (or remove) structures of a certain size.
- Top-hat: detects light objects on a dark background
  - also called white top-hat.
- Top-hat (image minus its opening):

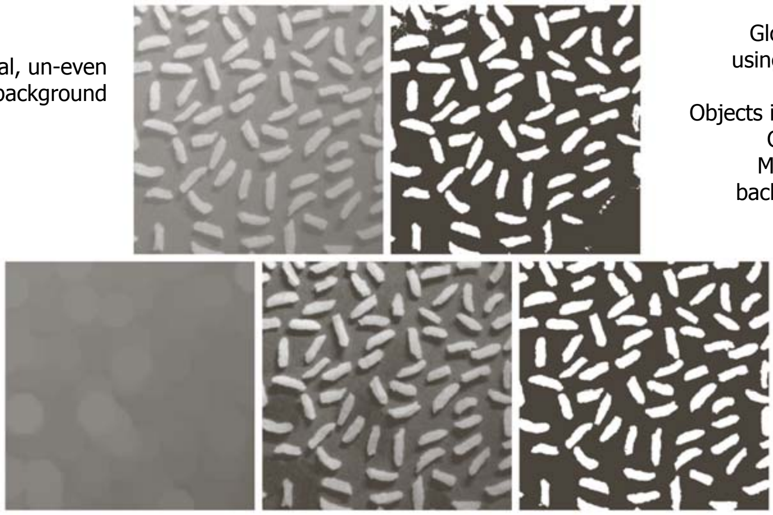
$$f - (f \circ b)$$

- Bottom-hat: detects dark objects on a bright background
  - also called black top-hat.
- Bottom-hat (closing minus image):

$$(f \bullet b) - f$$

- Very useful when correcting for uneven illumination or objects on a varying background ☺

Example – top-hat



Original, un-even background

Global thresholding using Otsu's method.

Objects in the lower right Corner disappear. Misclassification of background in upper right corner.

Opening with a 40x40 structuring element removes objects and gives an estimate of the background

Top-hat transform (original – opening)

Top-hat, thresholded with global threshold

Watershed – the idea

- A gray level image
  (or a gradient magnitude image
  or some other feature image)
  may be seen as a topographic relief,
  where increasing pixel value
  is interpreted as increasing height.
- Drops of water falling on a topographic relief will flow along paths to end up in local minima.
- The watersheds of a relief correspond to the limits of adjacent catchment basins of all the drops of water.

Watershed segmentation

- Can be used on images derived from:
  - The intensity image
  - Edge enhanced image
  - Distance transformed image (e.g. distance from object edge)
    - Thresholded image.
    - From each foreground pixel, compute the distance to a background pixel.
  - Gradient of the image
- Most common basis of WS: gradient image.

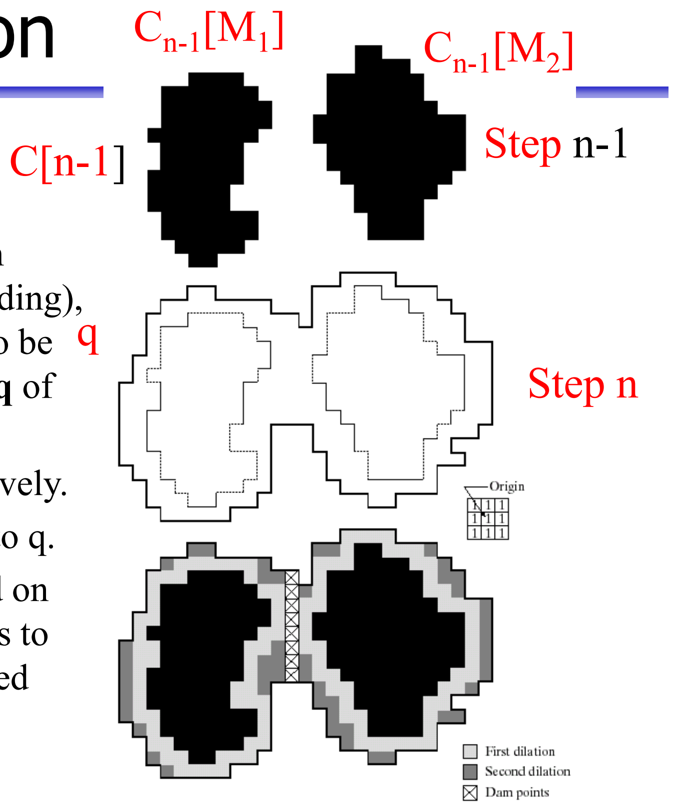
Watershed algorithm cont.

- The topography will be flooded with integer flood increments from n=min-1 to n=max+1.
- Let $C_n(M_i)$ be the set of coordinates of points in the catchment basin associated with M_i, flooded at stage n.
- This must be a connected component and can be expressed as $C_n(M_i) = C(M_i) \cap T[n]$ (only the portion of T[n] associated with basin M_i)
- Let C[n] be union of all flooded catchments at stage n:

$$C[n] = \bigcup_{i=1}^{R} C_n(M_i) \quad \text{and} \quad C[\max+1] = \bigcup_{i=1}^{R} C(M_i)$$

Dam construction

- Stage n-1: two basins forming separate connected components.
- To consider pixels for inclusion in basin k in the next step (after flooding), they must be part of T[n], and also be part of the connected component **q** of T[n] that $C_{n-1}[k]$ is included in.
- Use morphological dilation iteratively.
- Dilation of C[n-1] is constrained to q.
- The dilation can not be performed on pixels that would cause two basins to be merged (form a single connected component)

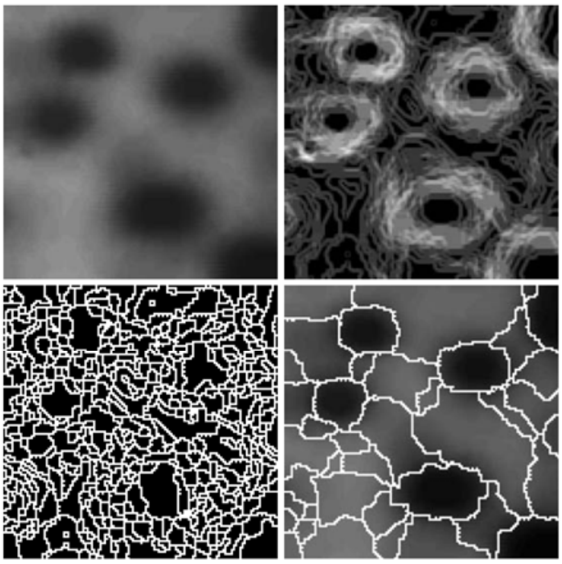


Watershed algorithm cont.

- Initialization: let C[min+1]=T[min+1]
- Then recursively compute C[n] from C[n-1]:
  - Let Q be the set of connected components in T[n].
  - For each component q in Q, there are three possibilities:

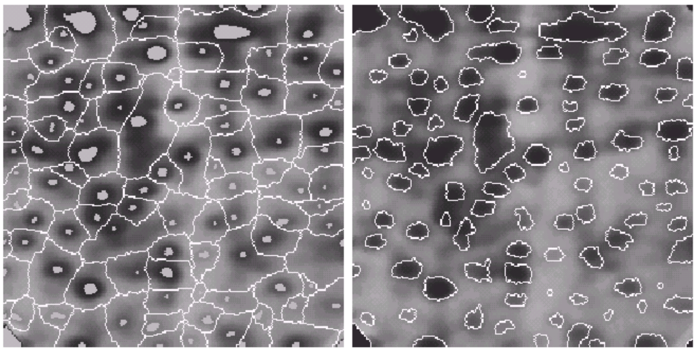
1. $q \cap C[n-1]$ is empty – new minimum
   Combine q with C[n-1] to form C[n].
2. $q \cap C[n-1]$ contains one connected component of C[n-1]
   q lies in the catchment basin of a regional minimum
   Combine q with C[n-1] to form C[n].

"Over-segmentation" or fragmentation



- Using the gradient image directly can cause fragmentation because of noise and small irrelevant intensity changes
- Improved by smoothing the gradient image or using markers

Solution: Watershed with markers



- A marker is an extended connected component in the image
- Can be found by intensity, size, shape, texture etc
- Internal markers are associated with the object (a region surrounded by bright point (of higher altitude))
- External markers are associated with the background (watershed lines)
- Segment each sub-region by some segmentation algorithm

How to find markers

- Apply filtering to get a smoothed image
- Segment the smooth image to find the internal markers.
  - Look for a set of point surrounded by bright pixels.
  - How this segmentation should be done is not well defined.
  - Many methods can be used.
- Segment smooth image using watershed to find external markers, with the restriction that the internal markers are the only allowed regional minima.
  The resulting watershed lines are then used as external markers.
- We now know that each region inside an external marker consists of a single object and its background.
- Apply a segmentation algorithm (watershed, region growing, threshold etc.) only inside each watershed.