CS484

Computational Vision

unsupervised and semi-supervised (interactive) image segmentation with low-level features

Low-level Segmentation

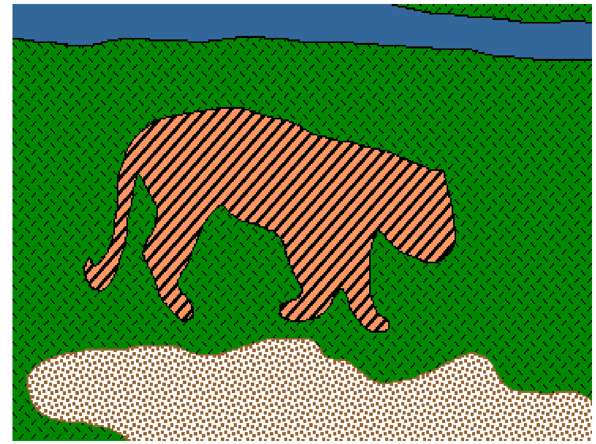
or grouping, partitioning, etc...

MAIN GOAL: in a simpler context of basic low-dimensional image features, e.g. colors or edges, understand **standard general techniques for grouping data points**, e.g. pixels, **with minimal supervision or unsupervised**

LATER: understand how to automatically build/learn complex "deep" features representing high-level (e.g. semantic) information in images and individual pixels. We will also discuss how to group them with or without supervision.

informal overview of

Image Segmentation



Goal:

find coherent "**blobs**" or specific "**objects / classes**"

low-level segmentation
(e.g. color-consistent region, smooth edge-aligned boundaries, coherent motion,...)

high-level / **semantic** segmentation
(e.g. humans, trees, cars, …)

computed from given image only, or also using a large training dataset?

unsupervised, semi-supervised, fully supervised ?

this topic

later topics *11* & *12*

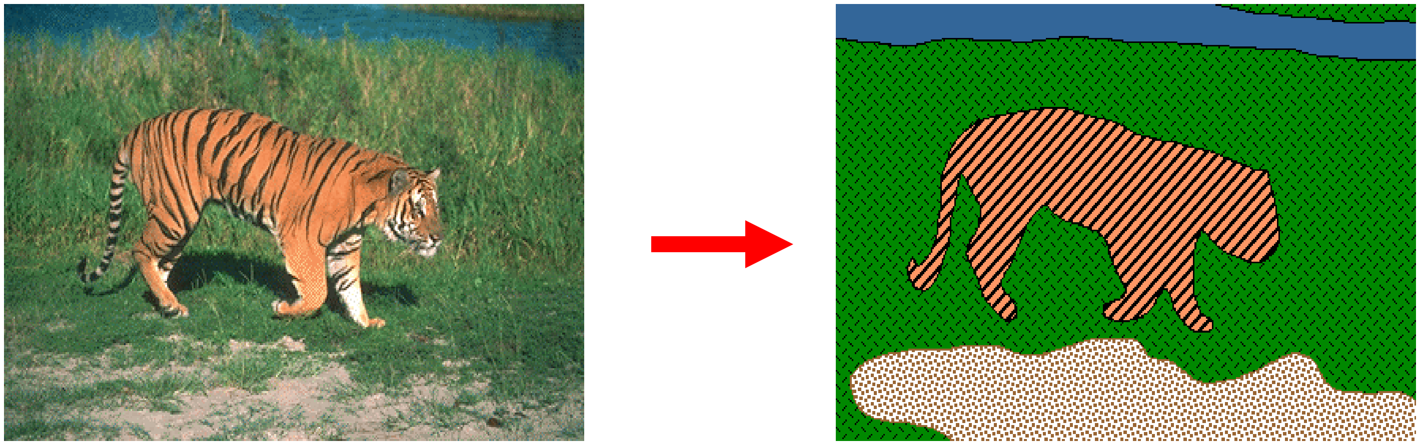
Low-level Image Segmentation

- Examples
  - *unsupervised* (background subtraction, color quantization, superpixels)
  - *semi-supervised* (photo-shop, medical image analysis)
- "Naïve" low-level segmentation
  - thresholding, region growing, etc
- Loss functions for (low-level) segmentation
  - region-based, boundary-based (geometric or shape)
  - **Clustering** criteria for general data points (basics from ML)
    - *K*-means, *K*-medians, *K*-modes, mean-shift
    - variance and distortion clustering, robust metrics
    - hard vs. soft clustering, probabilistic formulations, entropy, likelihoods, EM, GMM
    - parametric & non-parametric methods, kernel/spectral clustering, data embeddings
  - **Regularization** of segments (surfaces/shapes)
    - graphical models, active contours, geometric (shape/surface) regularization
    - combining with likelihood models and clustering methods
    - interactive segmentation (seeds/scribbles, boxes)

part I

part II

Two general groups of properties for (low-level) segmentation



- A: coherent **segment's appearance (region)**
  - consistent colors/texture, etc later: consistency with semantic class
  - agreement with known color distribution/density or likelihood model
- B: coherent **segment's shape (boundary)**
  - alignment to contrast edges later: consistency with semantic boundaries
  - boundary regularity / smoothness (low-level "shape prior")
  - consistency with expected shape (e.g. square, star, convex - higher-level priors)

Low-level Image Segmentation

first

"Naïve" segmentation techniques

A [based on appearance/color]: thresholding, likelihood ratio test

B [based on boundaries/contrast-edges]: region growing

Szeliski, Sec 5.2

Other readings: Sonka et al. Ch. 5
Gonzalez and Woods, Ch. 10

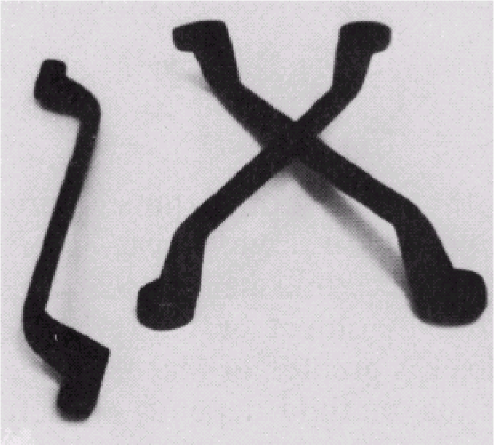
Naive Approach to *Appearance* (A)

SEGMENT'S APPEARANCE

-

Coherent color "blobs"

- Simplest way to define blob coherence is as similarity in brightness or color:



The tools become blobs

The house, grass, and sky make different blobs

Why is this useful?

AIBO RoboSoccer (VelosoLab)

Ideal Segmentation

can recognize objects with known ? simple ? color ? models

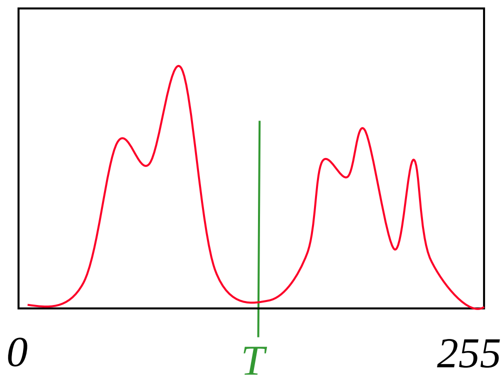
Result of a naive segmentation method

(first learn how to get this, then how to get better results)

even
if

known
simple
color

Basic ideas



low-level "appearance" features

intensities / colors

partition intensity histogram:

- **thresholding**
- log-likelihood ratio test

properties, assumptions:

- point processing, location is ignored
- i.i.d. assumption for colors in each blob/region
- assumes good "separation" of colors in each blob

(segmentation ← intensities/colors)

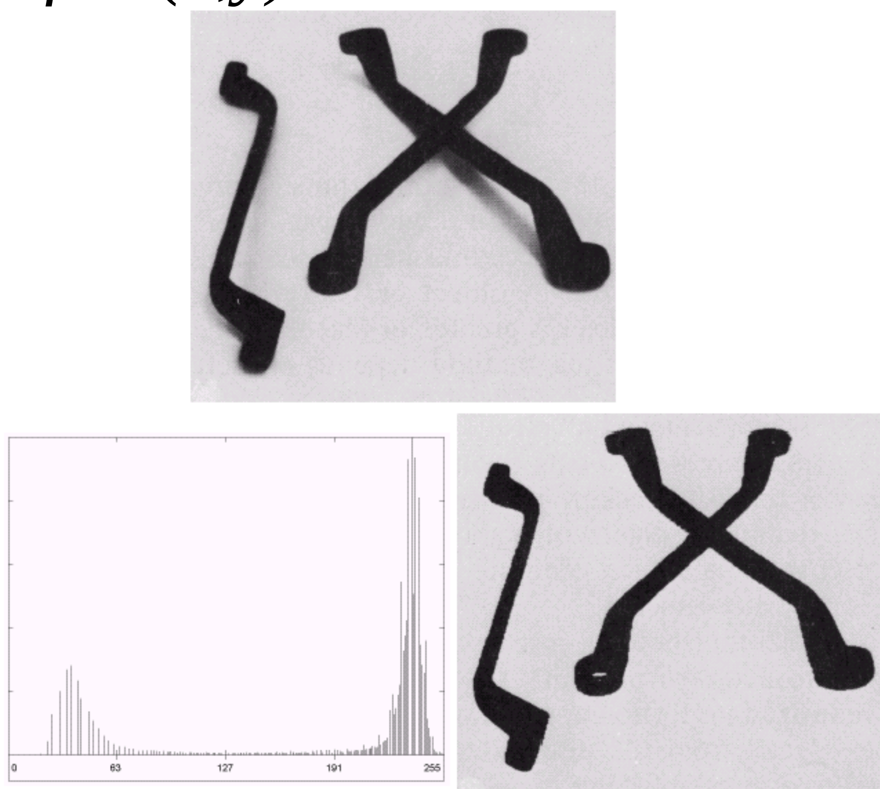
Thresholding

- Basic segmentation operation:

  $$mask(x,y) = 1 \text{ if } im(x,y) > T$$
  $$mask(x,y) = 0 \text{ if } im(x,y) < T$$

- T is threshold
  - user-defined
  - or automatic

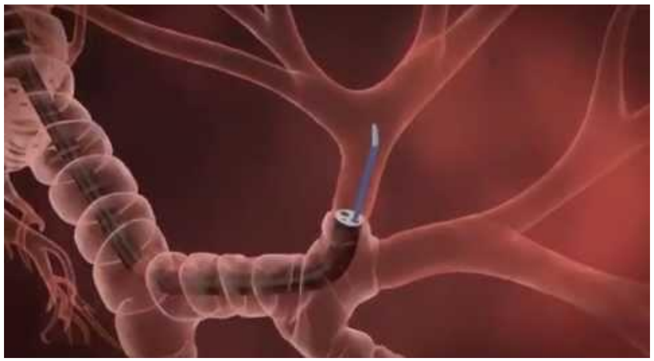
- Same as histogram partitioning:



a
b c

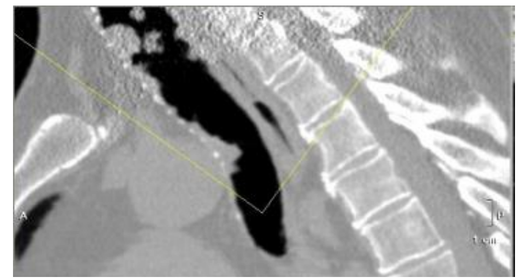
FIGURE 10.28 (a) Original image. (b) Image histogram. (c) Result of global thresholding with T midway between the maximum and minimum gray levels.

Sometimes works well…



bronchoscopy, colonoscopy, etc.

a) threshold CT volume -> binary mask

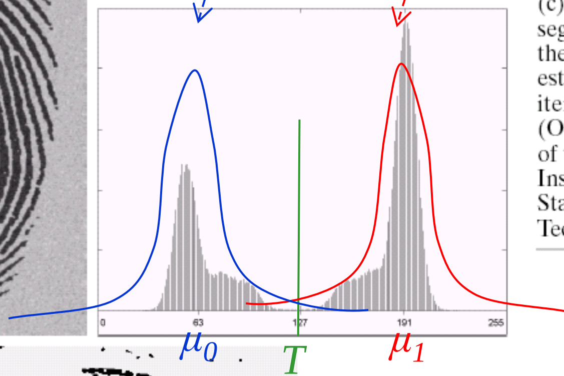


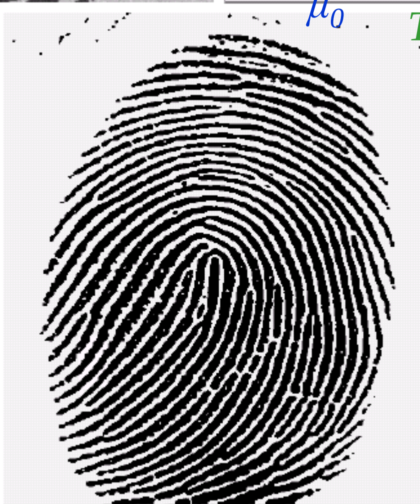
from real device to
non-invasive virtual test

Virtual
bronchoscopy,
colonoscopy,
etc.

UNIVERSITY OF WATERLOO

Sometimes works well…





a b
c

FIGURE 10.29 (a) Original image. (b) Image histogram. (c) Result of segmentation with the threshold estimated by iteration. (Original courtesy of the National Institute of Standards and Technology.)

Example: assume known probability distributions

$$P_1 = N(\mu_1, \sigma)$$

$$P_0 = N(\mu_0, \sigma)$$

$$r \geq 0 \quad \Leftrightarrow \quad I \geq T$$

$$T = \frac{\mu_1 + \mu_0}{2}$$

Here thresholding could be derived as **likelihood ratio test**

$$r_p \quad := \quad \log \frac{P_1(I_p)}{P_0(I_p)}$$

P_1 and P_0 are known color models for object and background

$r_p \geq 0 \quad \Rightarrow$ pixel p is object

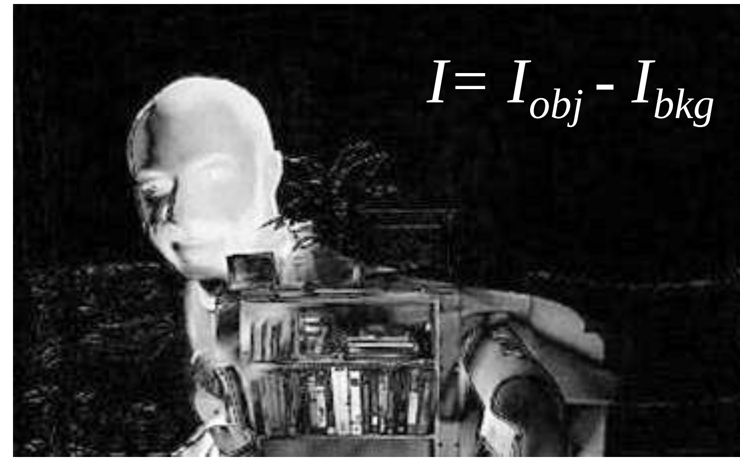
$r_p < 0 \quad \Rightarrow$ pixel p is background

Bayesian classification based on **Bayes posterior Pr(k|I)~Pr(I|k)** assuming **given data density models, i.i.d., equal priors**

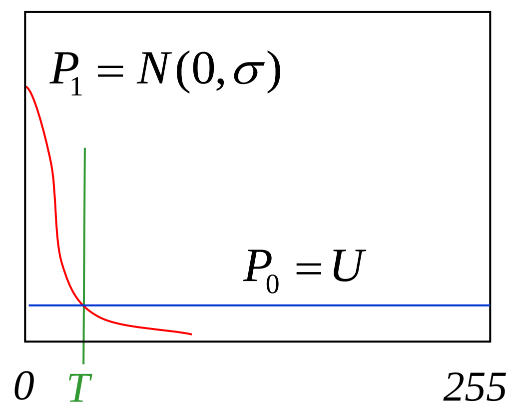
Sometimes works well…

-

=



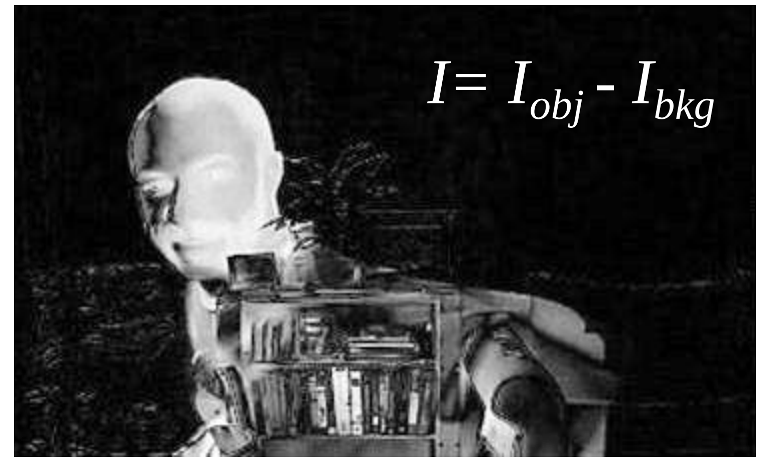
background
subtraction



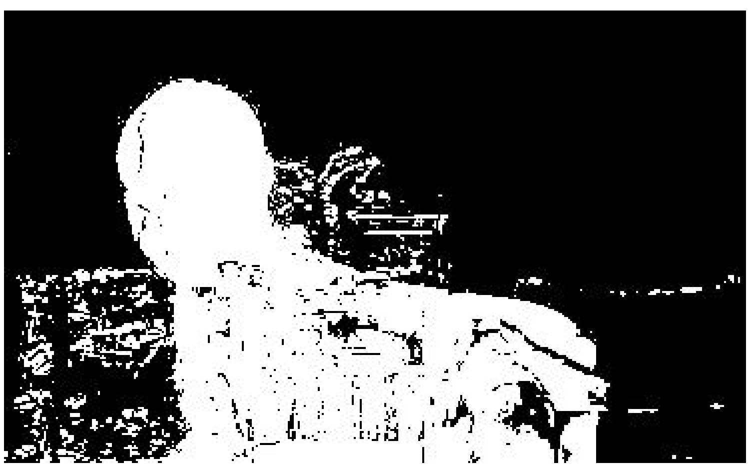
Sometimes works well… more often not

-

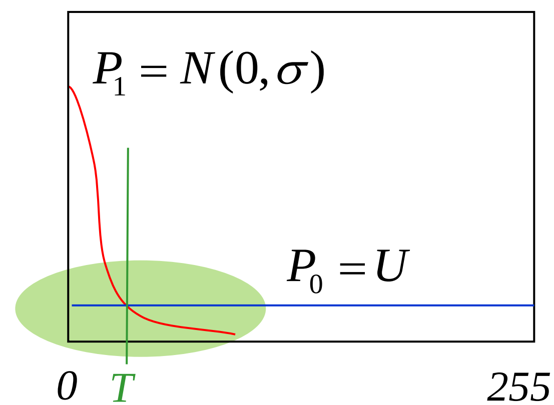
=



background subtraction



Threshold intensities below T

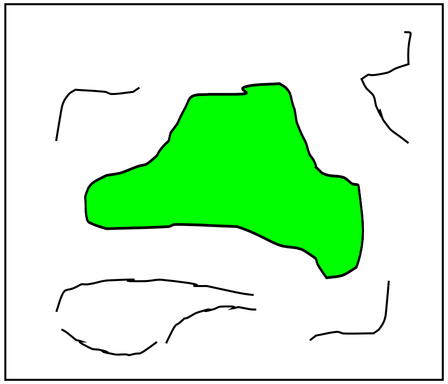


problems when color models have **overlapping support**

Naive Approach to *Boundary* (B)

SEGMENT'S BOUNDARY

Canny edges

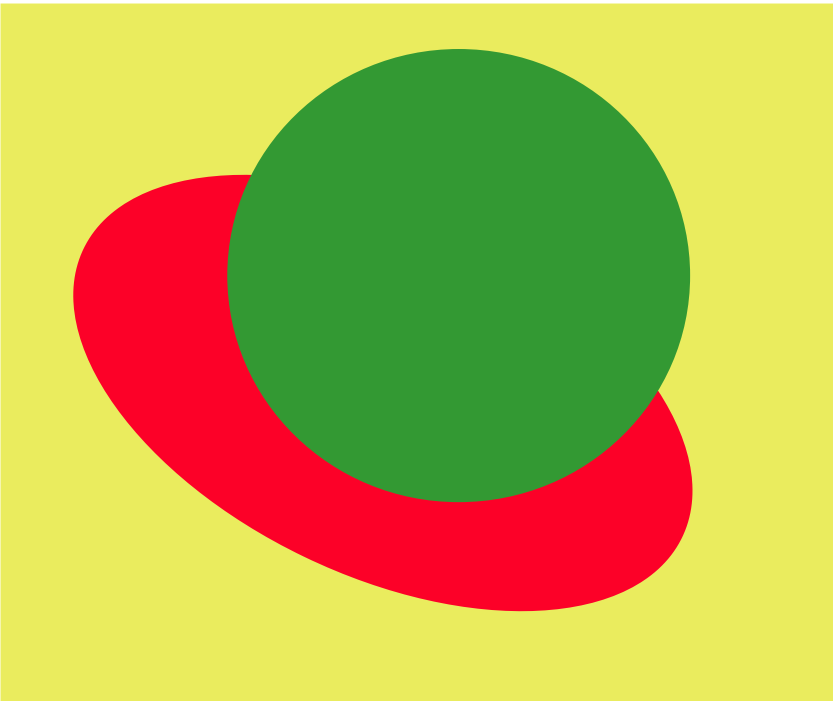


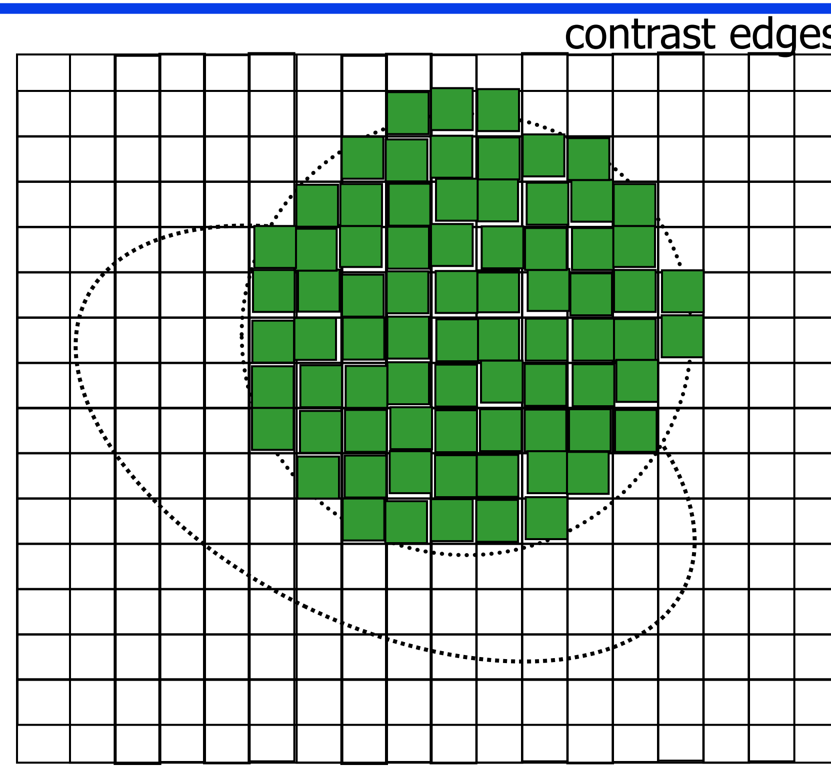
The most basic approaches attempt to find subsets of pixels completely surrounded by strong intensity edges

- **region growing**
- **watersheds**

(segmentation ← contrast edges)

Region growing



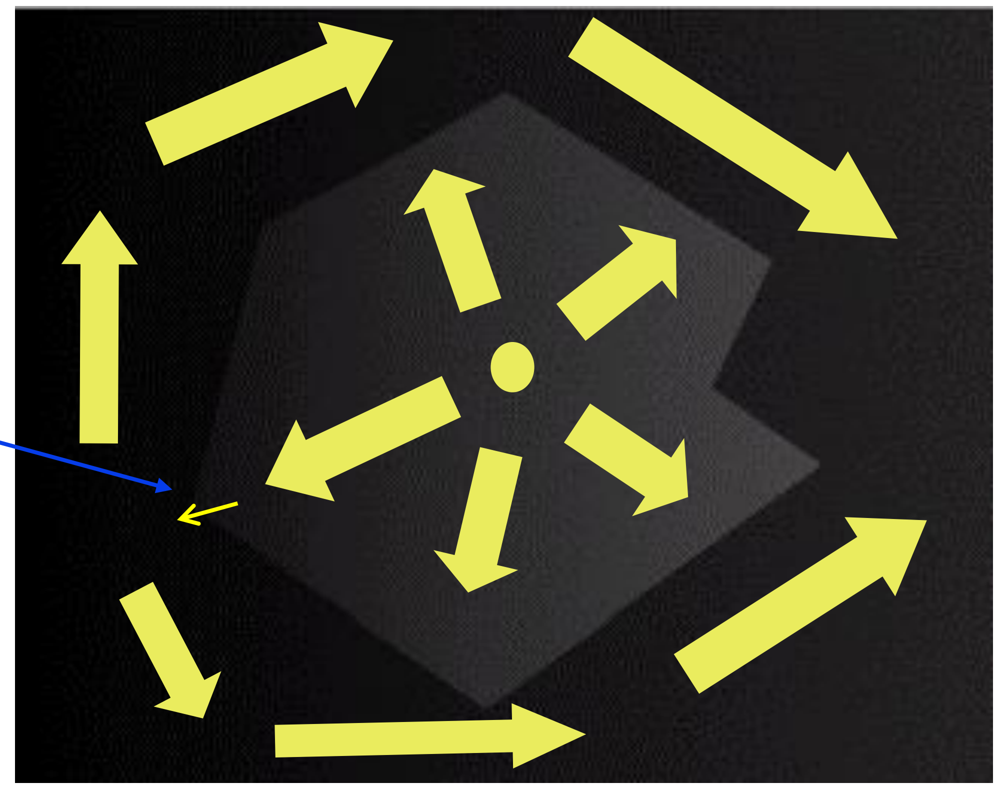


- Interactive initialization: assumes some initial set of pixel(s) K (seeds)
- For any pixel p in K add all its neighbors q such that $|Ip-Iq| < T$
- Iterate the step above until no neighbors of points in K satisfy $|Ip-Iq| < T$

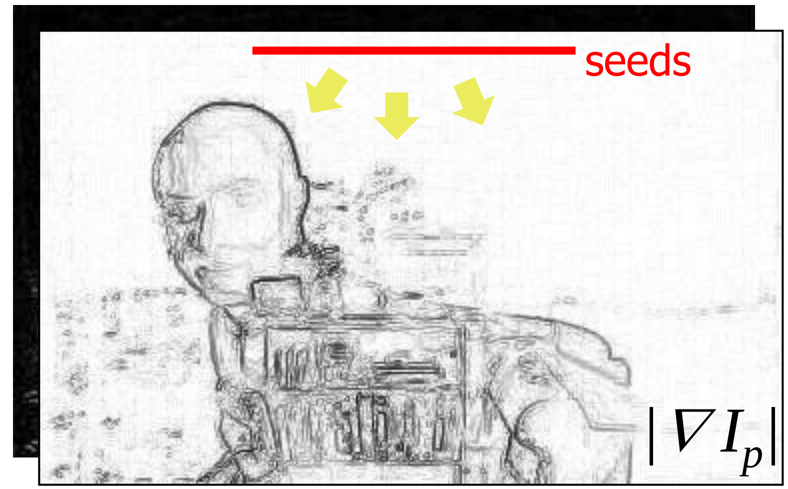
- **Breadth-First Search** (BFS) over grid neighbors (p,q) s.t. $|Ip-Iq|<T$
- Method stops at high-contrast edges (p,q) such that $|Ip-Iq|>T$

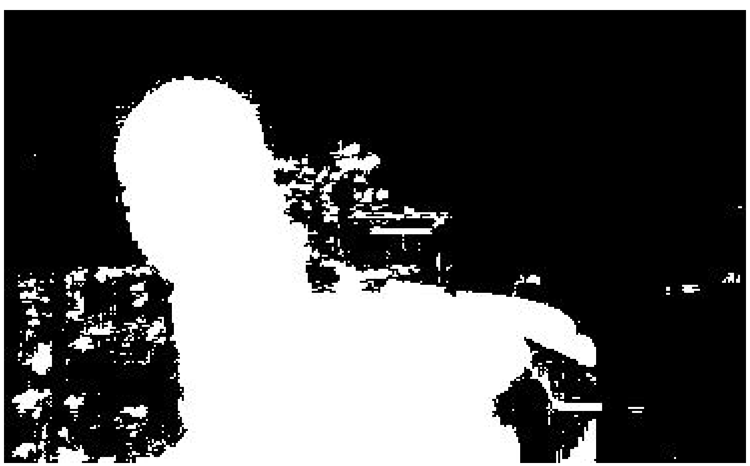
What can go wrong with region growing ?

Region growth may “leak” through **a single** **week spot** in the boundary



Region growing





Breadth First Search (seeds) :

$$|\nabla I_p| < T$$

Region growing

See region *leaks* into sky due to a weak boundary between them

From procedurally-defined towards “objective” segmentation

Should learn about **loss functions** (objectives) representing:

- Color/feature appearance consistency
  - replacing manually-selected thresholds by automatic partitioning of features
- Boundary/shape regularity
  - contrast edge continuation (fixing “ leaks”)
  - shape priors/regularization (addresses i.i.d. assumption)

- Combining multiple objectives (losses)

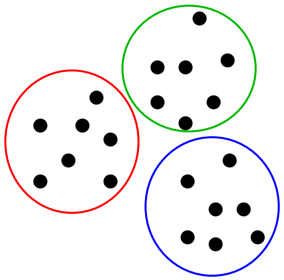
From procedurally-defined towards "objective" segmentation

Should learn about **objectives** (loss functions)

- Part I: **Clustering criteria**
  - general ML methods for unsupervised or weakly-supervised partitioning of data/features (low-level or deep)
  - can be used for image segmentation
- Part II: **Shape regularization models**
  - specific to image segmentation
  - appearance consistency
  - boundary/surface regularity (we focus on graphical formulations)

Highly informal **comments on terminology**

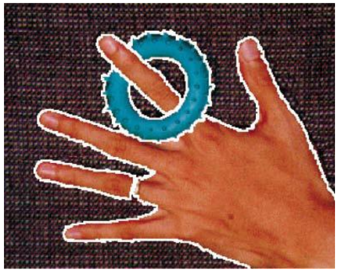
Clustering – general techniques (from ML) for partitioning arbitrary data/features $\{ f_p \} \subset R^N$ (of any dimensions) where p is index of a data point



e.g. movies in some feature space (length, director, tags,…)

Image Segmentation – methods specifically designed for partitioning image features $\{ f_p \} \subset R^N$ where p are <u>image grid pixels</u> $p = (x , y)$

- instead of an arbitrary collection $\{ f_p \}$ of data points, features are viewed as a sample from function $f(x,y) : R^2 \rightarrow R^N$



NOTE: general clustering techniques can be adapted to image segmentation problems. Thus, the boundary between the terms **clustering** and image **segmentation** is fuzzy.

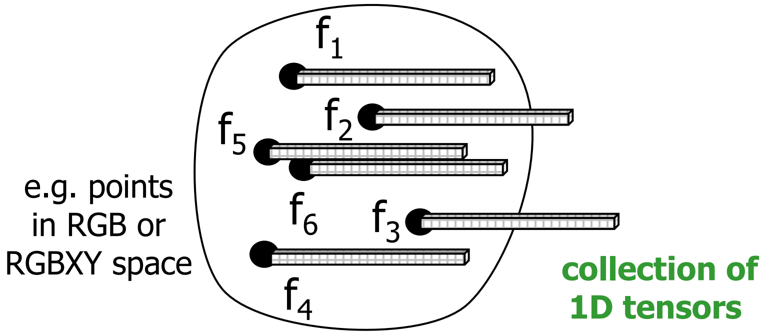
f_p - e.g. intensity I_p or color RGB_p or some "deep" feature at pixel $p = (x,y)$

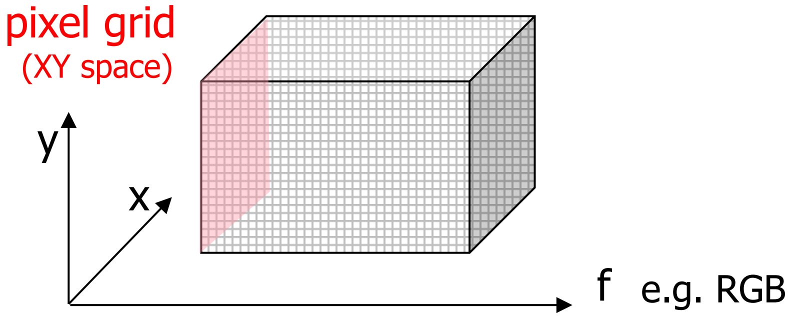
Alternative Views on Data Representation

Part I vs Part II

collection of feature vectors in R^n

3D tensor





general features
common in ML
K-means, GMM, general graph clustering, ...

features embedded in a regular 2D grid
common in computer vision
convolution, geometry, shape, spatial regularity, ...

We will learn: - clustering criteria and spatial regularization models
- how to combine different approaches

clustering objectives
Part I

Clustering Methods for General Data

(basics from ML with applications in Computer Vision)

UNIVERSITY OF WATERLOO

clustering objectives Part I

Clustering Methods for General Data

(basics from ML with applications in Computer Vision)

general criteria for unsupervised data clustering

- *K*-means, distortion clustering, probabilistic clustering, EM, GMM
- parametric vs non-parametric formulations
- kernel and spectral methods, graph clustering criteria

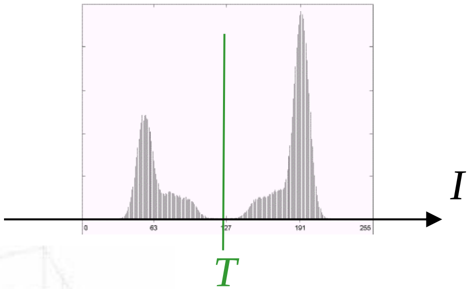
Szeliski, Sec 5.3

examples in image analysis

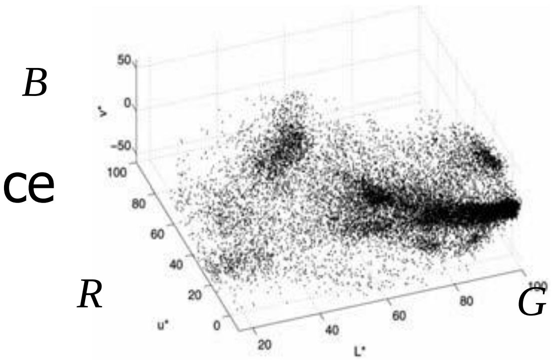
- color quantization, super-pixels, unsupervised segmentation

Motivation

- In 1D feature spaces (gray-scale intensities) it may be possible to set decision boundaries manually.



But, not easy in 3D feature space

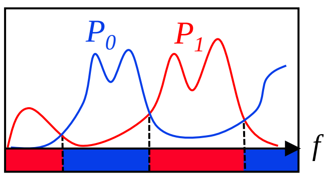


Note: in N dimensional feature space, the closest analogue of thresholding is a linear decision boundary (a hyperplane) defined by N+1 parameters.

- One can use log-likelihood ratio test if (color) distributions/densities for segments, *e.g.* P_1 and P_0, are known.

Bayesian decisions

$$\log \frac{P_1(f)}{P_0(f)} \geq 0$$

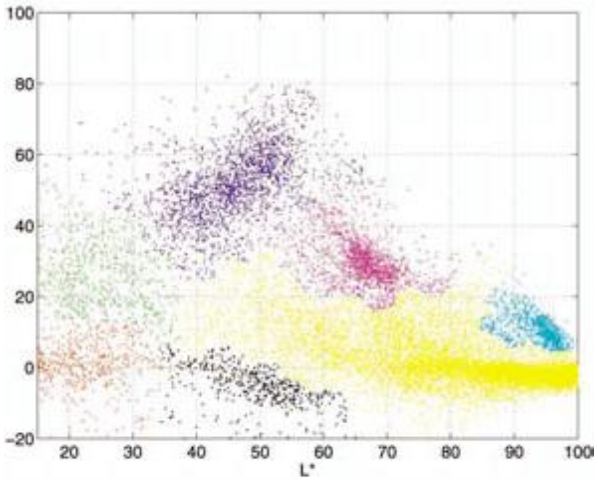


can yield arbitrarily complex decision boundaries in any feature space

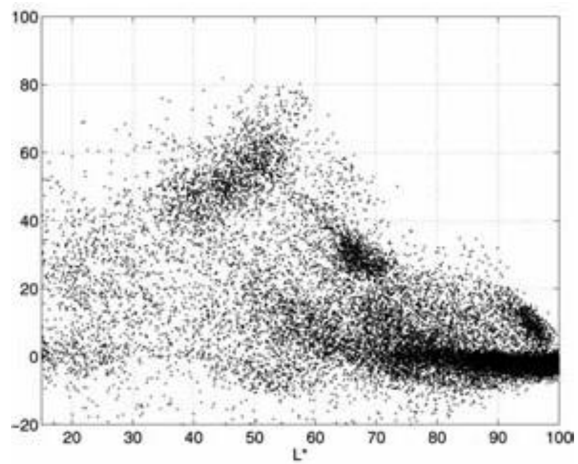
Does not work if distributions are not known

Motivation

decision boundaries for ND features
could be arbitrarily complex (surfaces)



?

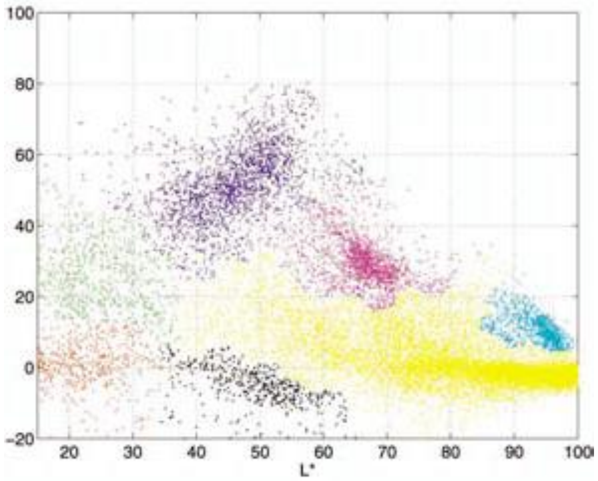


Example: break data points (e.g. RGB or RGBXY space) into a few clusters

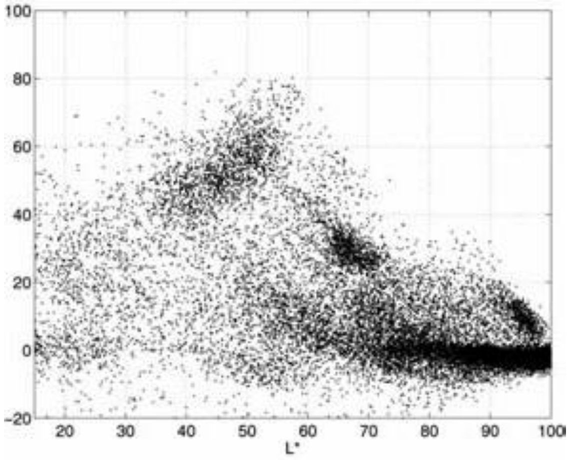
- color quantization
- superpixels
- semantic segmentation (later topic)

Motivation

decision boundaries for ND features
could be arbitrarily complex (surfaces)



?



Need automatic data *clustering* methods

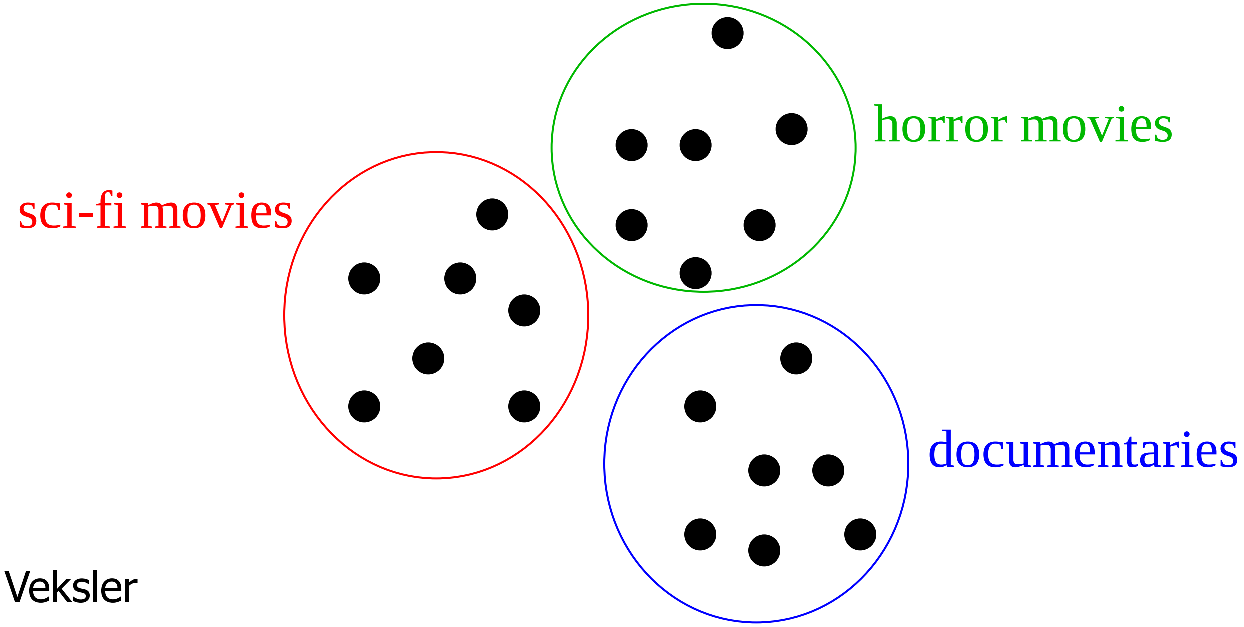
Szeliski, Sec 5.3

- parametric methods: *e.g.* K-means, soft K-means, GMM …
  - Note: many such methods are *generative* (estimate distribution parameters jointly with clustering the data)
- non-parametric: *e.g.* kernel K-means, graph partitioning, mean-shift …

Time permitting, we may also cover some deep clustering methodologies at the end of the course.

General Grouping or Clustering
(a.k.a. *unsupervised learning*)

- Have data points (samples, a.k.a. feature vectors, examples, etc.) $\boldsymbol{f}_1, \ldots, \boldsymbol{f}_p, \ldots$
- Cluster similar points into groups
  - points are **not** pre-labeled
  - think of clustering as "discovering" labels



slides from Olga Veksler

How does this Relate to Image Segmentation?

- Represent image pixels as feature vectors $\boldsymbol{f}_1,\ldots,\boldsymbol{f}_p,\ldots$ or $\{ f_p \mid p \in \Omega \}$ ← set of all pixels or indices
  - For example, each pixel can be represented as
    - intensity, gives one dimensional (1D) feature vectors
    - color, gives three-dimensional (3D) feature vectors, e.g. RGB
    - color + coordinates, gives five-dimensional (5D) feature vectors, e.g. RGBXY
- Cluster them into ***K*** clusters, i.e. ***K*** segments

input image

9 4 2	7 3 1	8 6 8
8 2 4	5 8 5	3 7 2
9 4 5	2 9 3	1 4 4

feature vectors for clustering based on color

[9 4 2] [7 3 1] [8 6 8]

[8 2 4] [5 8 5] [3 7 2]

[9 4 5] [2 9 3] [1 4 4]

RGB (or LUV) space clustering

How does this Relate to Image Segmentation?

- Represent image pixels as feature vectors $\boldsymbol{f}_1, \ldots, \boldsymbol{f}_p, \ldots$ or $\{ f_p \mid p \in \Omega \}$ ← set of all pixels or indices
  - For example, each pixel can be represented as
    - intensity, gives one dimensional (1D) feature vectors
    - color, gives three-dimensional (3D) feature vectors, e.g. RGB
    - color + coordinates, gives five-dimensional (5D) feature vectors, e.g. RGBXY
- Cluster them into ***K*** clusters, i.e. ***K*** segments

input image

9 4 2	7 3 1	8 6 8
8 2 4	5 8 5	3 7 2
9 4 5	2 9 3	1 4 4

feature vectors for clustering based on color and image coordinates

[9 4 2 0 0] [7 3 1 0 1] [8 6 8 0 2]

[8 2 4 1 0] [5 8 5 1 1] [3 7 2 1 2]

[9 4 5 2 0] [2 9 3 2 1] [1 4 4 2 2]

RGBXY (or LUVXY) space clustering

K-means Clustering: Objective Function

- Probably the most popular clustering algorithm
  - assumes the number of clusters is given - ***K***
- optimizes (approximately) the following **objective function** for variables S^k and μ_k

$$SSE(S,\mu) = \sum_{k=1}^{K} \sum_{p \in S^k} \|f_p - \mu_k\|^2$$

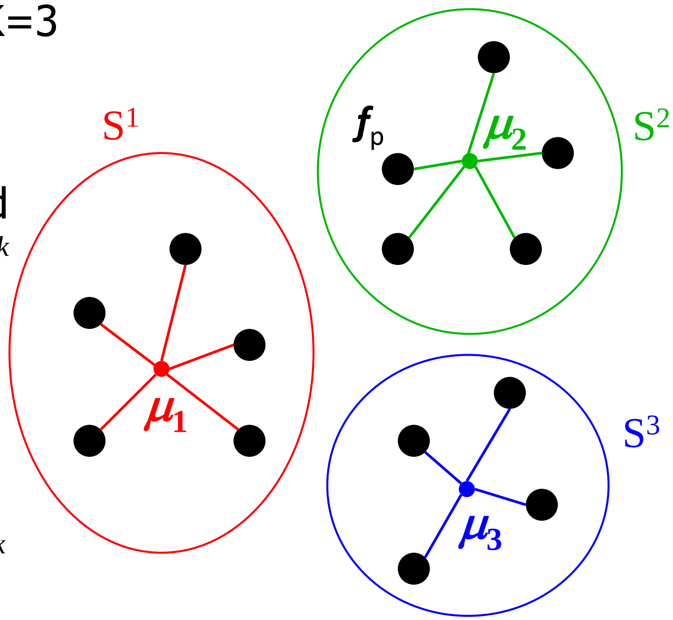
sum of squared errors from cluster center μ_k

(both S^k and μ_k are **unknown**, to be computed)

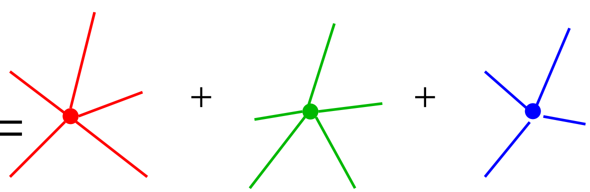
assume given K=3

consider some fixed clusters S^k

consider their centers μ_k



$SSE =$



optimization "variables"

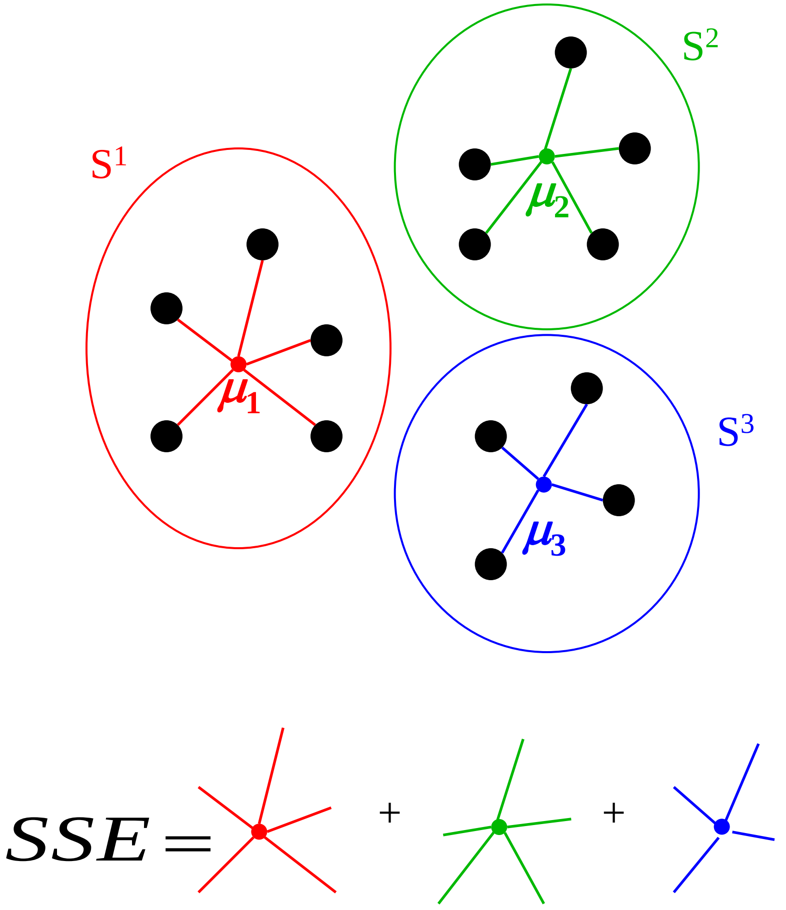
subsets $S = (S^1, \ldots, S^K)$

centers $\mu = (\mu_1, \ldots, \mu_K)$

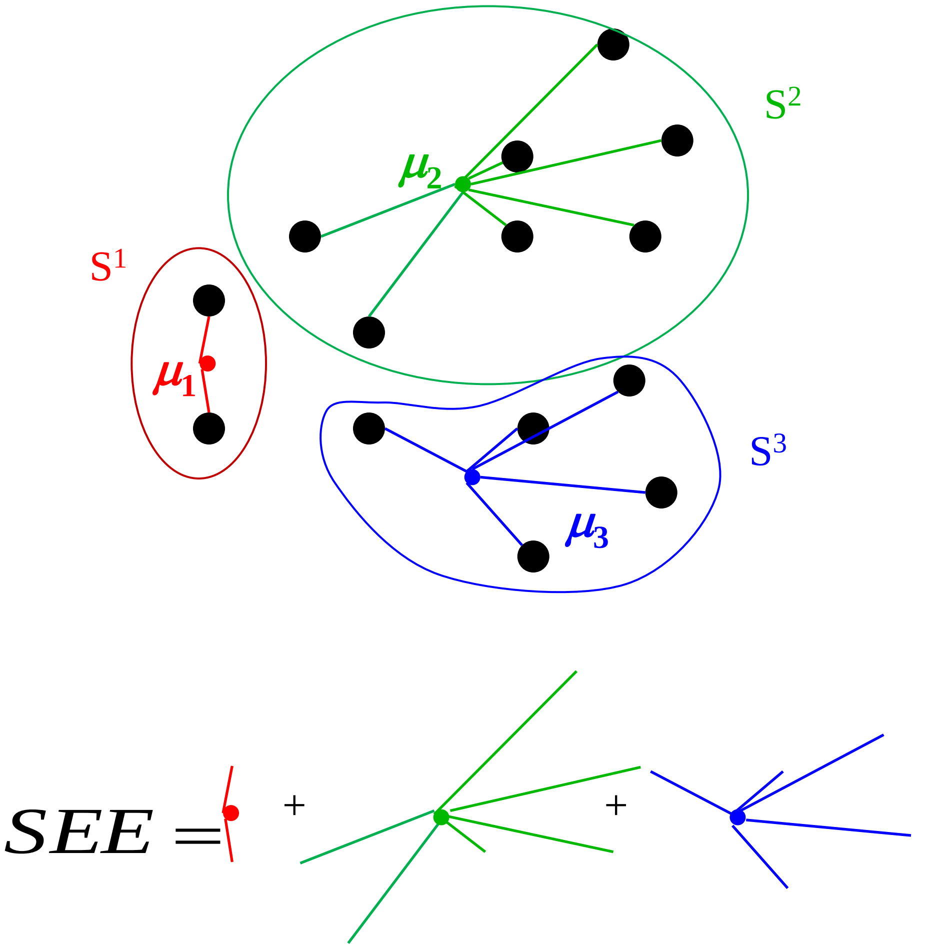
Q: given cluster S^k, what best describes <u>optimal center</u> $\hat{\mu}_k = \arg\min_{\mu_k} \sum_{p \in S^k} \|f_p - \mu_k\|^2$?

A: center of mass **B**: average **C**: median **D**: least squares solver

K-means Clustering: Objective Function



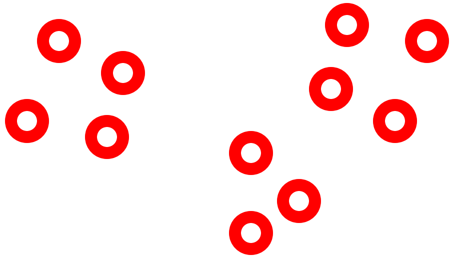
Good (tight) clustering
=> smaller value of SSE



Bad (loose) clustering
=> larger value of *SSE*

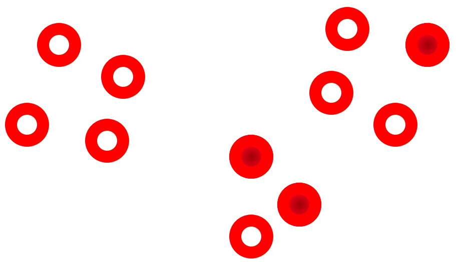
K-means Clustering: Algorithm

- Initialization step
  1. pick ***K*** cluster centers randomly (e.g. from data points)



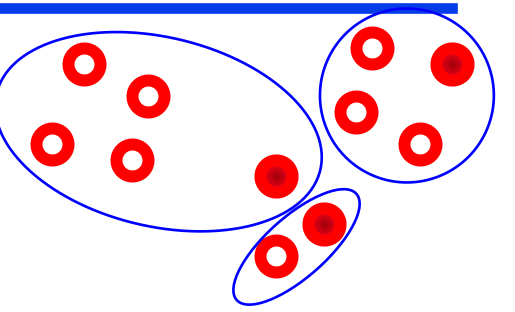
K-means Clustering: Algorithm

- Initialization step
  1. pick ***K*** cluster centers randomly (e.g. from data points)

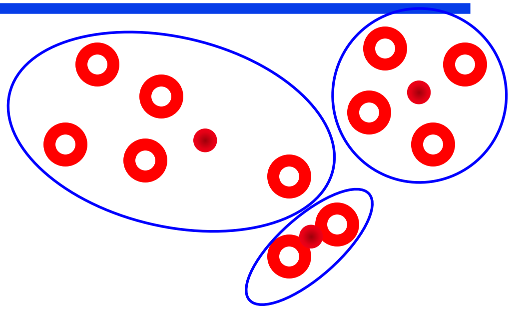


K-means Clustering: Algorithm

- Initialization step
  1. pick ***K*** cluster centers randomly
  2. assign each sample to its closest center



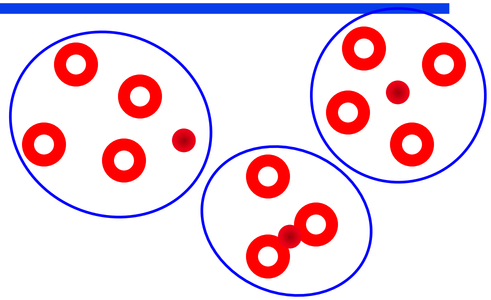
K-means Clustering: Algorithm



- Initialization step
  1. pick ***K*** cluster centers randomly
  2. assign each sample to its closest center

- Iteration steps
  1. compute centers as cluster means $\mu_k = \frac{1}{|S^k|} \sum_{p \in S^k} f_p$

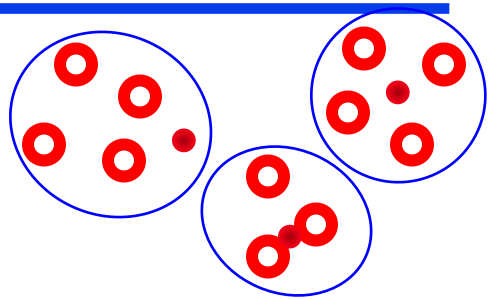
K-means Clustering: Algorithm



- Initialization step
  1. pick ***K*** cluster centers randomly
  2. assign each sample to its closest center

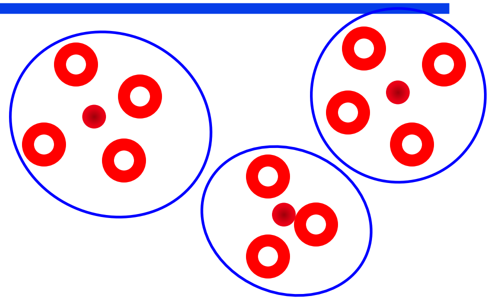
- Iteration steps
  1. compute centers as cluster means $\mu_k = \frac{1}{|S^k|} \sum_{p \in S^k} f_p$
  2. re-assign each point f_p to the closest mean

K-means Clustering: Algorithm



- Initialization step
  1. pick ***K*** cluster centers randomly
  2. assign each sample to its closest center
- Iteration steps
  1. compute centers as cluster means $\mu_k = \frac{1}{|S^k|} \sum_{p \in S^k} f_p$
  2. re-assign each point f_p to the closest mean
- Iterate until clusters stop changing

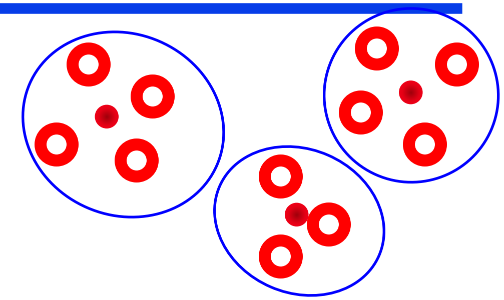
K-means Clustering: Algorithm



- Initialization step
  1. pick ***K*** cluster centers randomly
  2. assign each sample to its closest center
- Iteration steps
  1. compute centers as cluster means $\mu_k = \frac{1}{|S^k|} \sum_{p \in S^k} f_p$
  2. re-assign each point f_p to the closest center
- Iterate until clusters stop changing

K-means Clustering: Algorithm

- Initialization step
  1. pick ***K*** cluster centers randomly
  2. assign each sample to its closest center



- Iteration steps
  1. compute centers as cluster means $\mu_k = \frac{1}{|S^k|} \sum_{p \in S^k} f_p$
  2. re-assign each point f_p to the closest mean
- Iterate until clusters stop changing

Lloyd's algorithm (1957)

- Each step decreases the value of the objective function

$$E(S, \mu) = \sum_{k=1}^{K} \sum_{p \in S^k} \| f_p - \mu_k \|^2$$

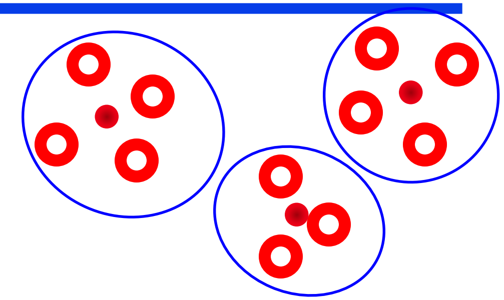
optimization variables

$$S = (S^1, \ldots, S^K)$$
$$\mu = (\mu_1, \ldots, \mu_K)$$

block-coordinate descent: step 1 optimizes $\{\mu_k\}$ for fixed $\{S_k\}$, step 2 optimizes $\{S_k\}$ for fixed $\{\mu_k\}$

K-means Clustering: Algorithm

- Initialization step
  1. pick ***K*** cluster centers randomly
  2. assign each sample to its closest center



- Iteration steps
  1. compute centers as cluster means $\mu_k = \frac{1}{|S^k|} \sum_{p \in S^k} f_p$
  2. re-assign each point f_p to the closest mean
- Iterate until clusters stop changing

Lloyd's algorithm (1957)

- Each step decreases the value of the objective function

$$E(S,\mu) = \sum_{k=1}^{K} \sum_{p \in S^k} \|f_p - \mu_k\|^2$$

optimization variables

$$S = (S^1, \ldots, S^K)$$
$$\mu = (\mu_1, \ldots, \mu_K)$$

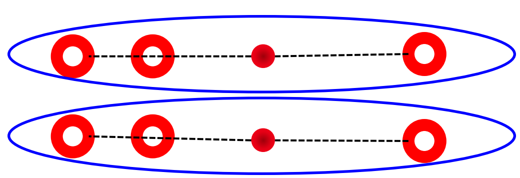
block-coordinate descent: step 1 optimizes $\{\mu_k\}$ for fixed $\{S_k\}$, step 2 optimizes $\{S_k\}$ for fixed $\{\mu_k\}$

K-means: Approximate Optimization

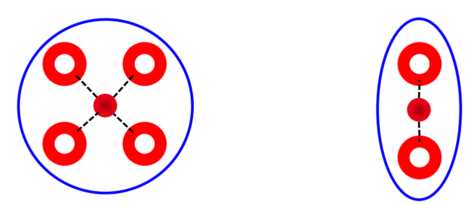
- K-means is fast and (sometimes) works well in practice
- But can get stuck in a local minimum of objective $\boldsymbol{E_K}$
  - not surprising, since the exact optimization of its objective is NP-hard

initialization

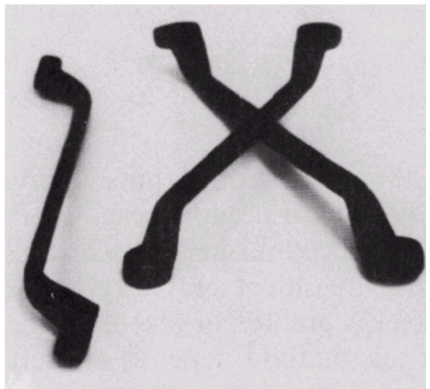
converged to local min

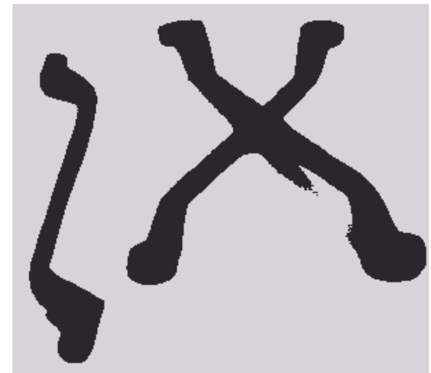


global minimum

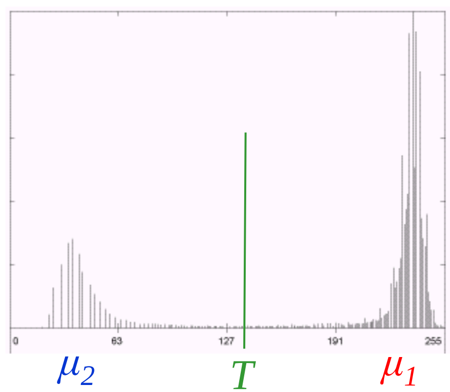


K-means clustering examples: Segmentation





$$I_p \in R^1$$



here K-means finds compact clusters of pixels' intensities

In this case K-means (K=2) implicitly finds a good threshold (between 2 clusters)

K-means for colors (RGB features): Segmentation?

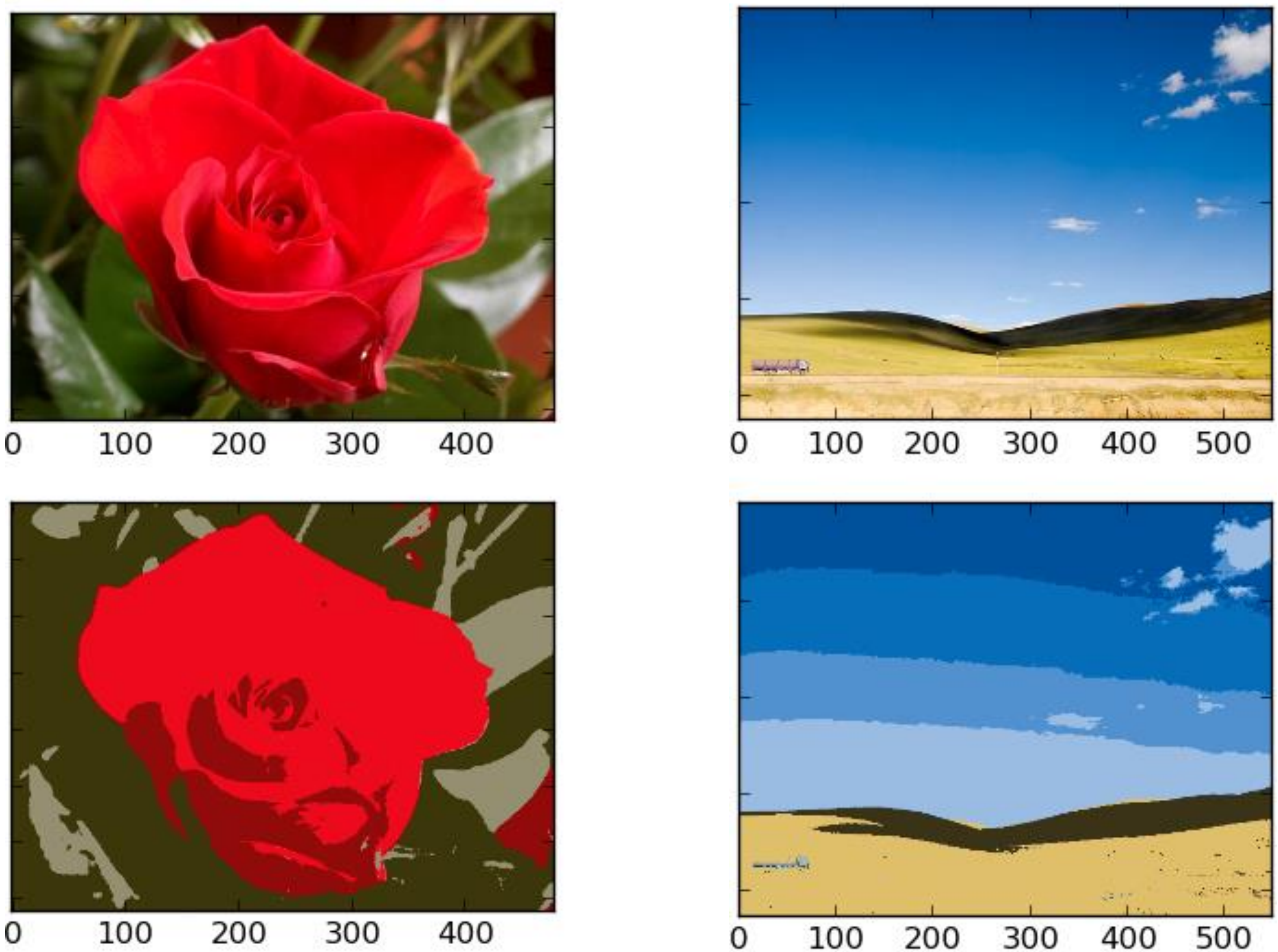
k **= 3**

(**mean** color is used to show each segment/cluster)

k **= 5**

k **= 10**

K-means for colors (RGB features): Color Quantization



NOTE
bias to
equal-size
clusters

Where "size" can mean both clusters' cardinalities and clusters' diameters in the feature space

K-means clustering examples: Adding XY features

color quantization

RGB features

superpixels

RGBXY features

Voronoi cells

XY features only

TECHNICAL NOTE

$$\|f_p - \mu\|^2_{RGBXY} \;=\; \|R_p - \mu^R\|^2 + \|G_p - \mu^G\|^2 + \|B_p - \mu^B\|^2 \;+\; \lambda\;\left[\|X_p - \mu^X\|^2 + \|Y_p - \mu^Y\|^2\right]$$

can be fixed or estimated, see "elliptic" K-means estimating (co)variances, later

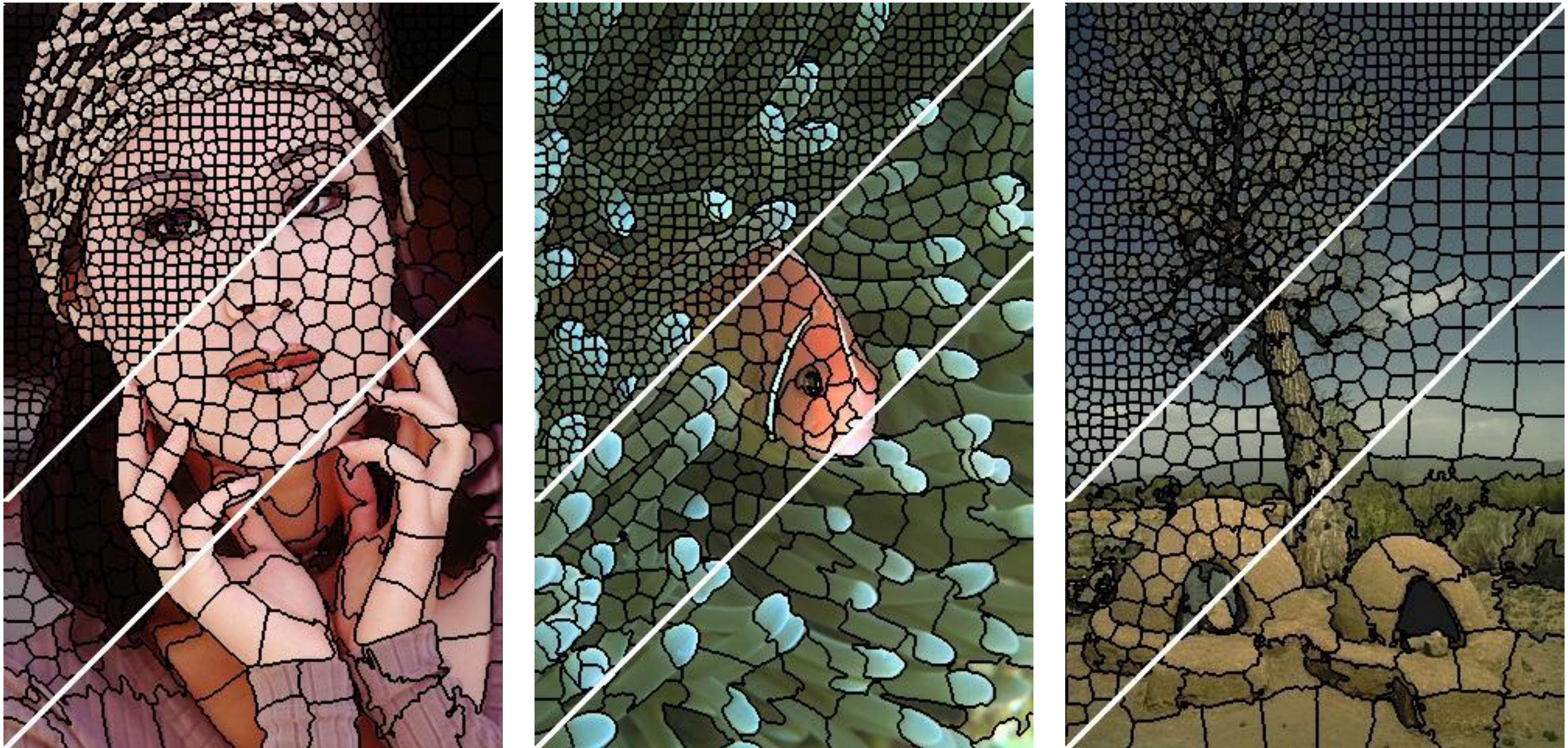
$$\equiv\; \underbrace{(f_p - \mu)^\top \Sigma^{-1} (f_p - \mu)}_{\|f_p - \mu\|^2_\Sigma} \qquad \text{for} \quad \Sigma = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & \frac{1}{\lambda} & 0 \\ 0 & 0 & 0 & 0 & \frac{1}{\lambda} \end{bmatrix}$$

a simple special case of *Mahalanobis distance*

K-means clustering examples: Superpixels

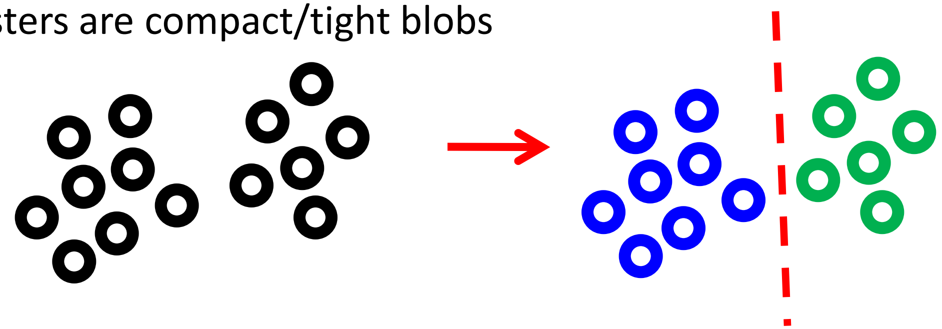
- Apply K-means to RGBXY features

[SLIC superpixels, Achanta et al., PAMI 2011]

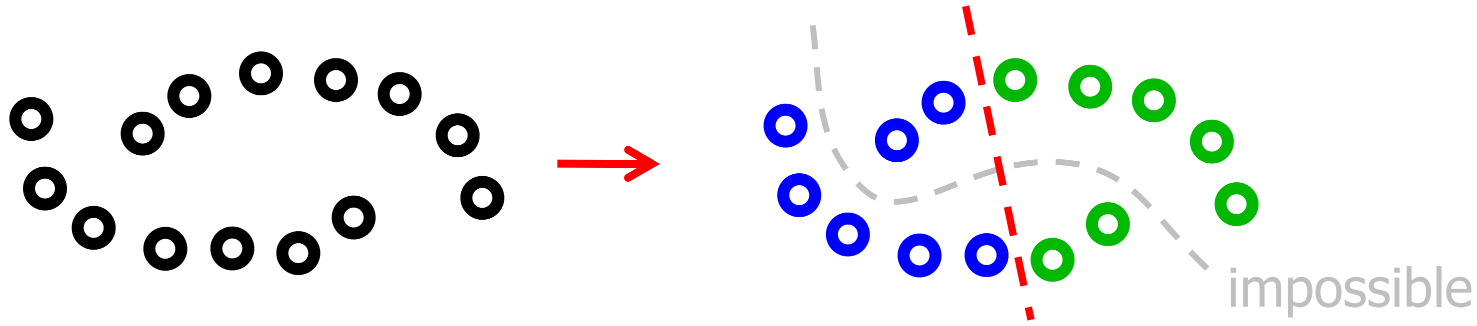


K-means Properties

- Works well when clusters are compact/tight blobs



- Fails to find non-compact clusters

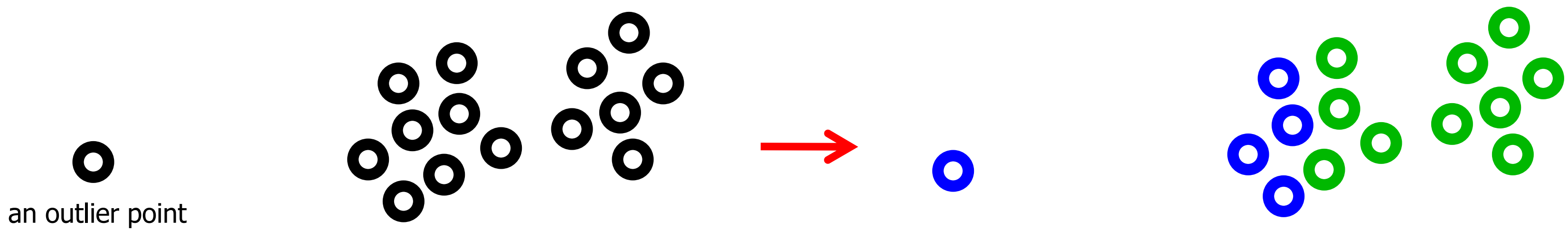


K-means can only produce <u>linear boundaries</u> between clusters (why?)

Thus, K-means does not work well if clusters can not be separated by a line/plane.

K-means Properties

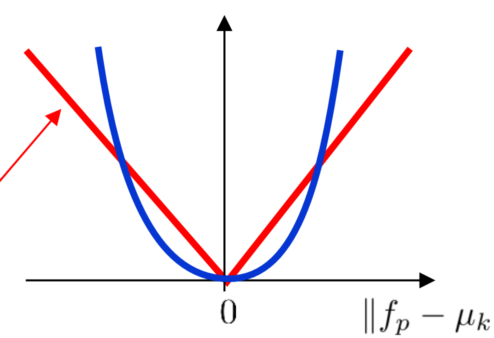
- Sensitive to outliers



Explanation: squared distance error grows too fast making any outlier extremely costly. This also explains non-robustness of a "sample mean" statistic.

$$SSE(S,\mu) = \sum_{k=1}^{K} \sum_{p \in S^k} \|f_p - \mu_k\|^2$$

Possible solution: replace squared distances by absolute distances that grow at a slower pace.



$$SAE(S,\mu) = \sum_{k=1}^{K} \sum_{p \in S^k} \|f_p - \mu_k\|$$

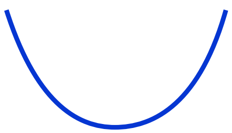
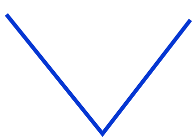
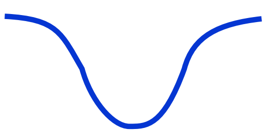
Interestingly, in this case the optimal value of μ_k is the "median" of set S^k instead of its "mean"

(generalization)

Distortion Clustering

can use different "distortion" measures

$$E(S,\mu) = \sum_{k=1}^{K} \sum_{p \in S_k} \|f_p - \mu_k\|_d$$

examples of distortion measure $\|\cdot\|_d$			interpretation of parameters μ_k
	$\|\cdot\|_d = \|\cdot\|^2$	squared L_2 norm	**K-means**
	$\|\cdot\|_d = \|\cdot\|$	absolute L_2 norm	**K-medians**
	$\|\cdot\|_d = 1 - \exp(-\|\cdot\|^2)$		**K-modes**

NOTE: there are other generalizations of K-means using a **probabilistic interpretation of SSE**

K-means as probabilistic clustering

(probabilistic model parameter fitting)

For fixed clusters, optimization of parameters μ_k this can be seen as...

- *maximum likelihood* estimation of Gaussian density parameters μ_k
- *Gaussian classifier* estimation (see any intro Machine Learning courses)

Since clusters are also estimated...

- K-means can be seen as ***unsupervised Gaussian classification***

$$E(S,\mu) \;=\; \sum_{k=1}^{K} \sum_{p \in S^k} \|f_p - \mu_k\|^2$$

equivalent (easy to check)

sum of negative log-likelihoods (**NLL** loss)

negative log-likelihood can be seen as "probabilistic" distortion measure

probabilistic K-means

general formulation since any parametric density functions can be used

$$E(S,\mu) \;=\; -\sum_{k=1}^{K} \sum_{p \in S^k} \log P(f_p \,|\, \mu_k) \;+\; const$$

multi-variate (i.e. $x, \mu \in R^N$)

Gaussian distribution

(simple special case $\Sigma = \sigma^2 \mathbf{I}$)

$$P(x|\mu) \;=\; \frac{1}{\sqrt{(2\pi\sigma^2)^N}} \;\exp -\frac{\|x-\mu\|^2}{2\,\sigma^2}$$

Towards soft clustering…

Fuzzy K-means

NOTE:
optimal S_p for this "relaxed" loss are ***one-hot*** distributions $\boldsymbol{S}_p \in \Delta^K_{\mathbf{v}}$ *vertices of probability simplex*
e.g., $\boldsymbol{S}_p = (0,1,0,0,0)$

Why ?

$$E(S,\mu) \;=\; -\sum_{k=1}^{K}\sum_{p} S_p^k \,\log P(f_p\,|\,\mu_k)$$

Let's represents segmentation using relaxed segmentation variables $\boldsymbol{S}_p$

categorical distribution at point p over K clusters

$$\boldsymbol{S}_p \;:=\; \{S_p^k :\; S_p^k \geq 0,\; \sum_k S_p^k = 1\} \in \Delta^K$$

"probability simplex"

NOTE:
"probabilistic" formulation but clusters S^k are "**hard**" ("deterministic")

$$E(S,\mu) \;=\; -\sum_{k=1}^{K}\sum_{p\in S^k} \log P(f_p\,|\,\mu_k)$$

multi-variate (i.e. $x,\mu \in R^N$)
Gaussian distribution
(simple special case $\Sigma = \sigma^2\,\mathbf{I}$)

$$P(x|\mu) \;=\; \frac{1}{\sqrt{(2\pi\sigma^2)^N}}\;\exp - \frac{\|x-\mu\|^2}{2\,\sigma^2}$$

Towards soft clustering…

Fuzzy K-means

standard measure of "chaos" in any distribution $\boldsymbol{p} \in \Delta^K$

$$H(\boldsymbol{p}) := -\sum_k p^k \log p^k$$

now, optimal S_p for positive temperatures $T > 0$
are "soft" distributions in the interior of the simplex $\boldsymbol{S}_p \in \Delta^K$

fuzzy or soft K-means

entropy of distribution $\boldsymbol{S}_p$

$$E(S,\mu) \;=\; -\sum_{k=1}^{K}\sum_{p} S_p^k \,\log P(f_p \,|\, \mu_k) \;-\; T\sum_p H(\boldsymbol{S}_p)$$

Let's represents segmentation using relaxed segmentation variables $\boldsymbol{S}_p$

categorical distribution at point p over K clusters

$$\boldsymbol{S}_p \;:=\; \{S_p^k : S_p^k \geq 0, \; \sum_k S_p^k = 1\} \in \Delta^K$$

"*probability simplex*"

"hard" K-means

$$E(S,\mu) \;=\; -\sum_{k=1}^{K}\sum_{p\in S^k} \log P(f_p \,|\, \mu_k)$$

multi-variate (i.e. $x, \mu \in R^N$)
Gaussian distribution
(simple special case $\Sigma = \sigma^2 \mathbf{I}$)

$$P(x|\mu) \;=\; \frac{1}{\sqrt{(2\pi\sigma^2)^N}} \;\exp - \frac{\|x-\mu\|^2}{2\,\sigma^2}$$

Towards soft clustering…

Gaussian Mixture Models (GMM)

Consider another **probabilistically motivated** approach to soft clustering…

$$E(S,\mu) \;=\; -\sum_p \log \sum_{k=1}^{K} S_p^k \, P(f_p \,|\, \mu_k)$$

Let's represents segmentation using relaxed segmentation variables $\boldsymbol{S}_p$

categorical distribution at point p over K clusters

$$\boldsymbol{S}_p \;:=\; \{S_p^k : \; S_p^k \geq 0, \; \sum_k S_p^k = 1\} \in \Delta^K$$

"probability simplex"

"hard" K-means

$$E(S,\mu) \;=\; -\sum_{k=1}^{K} \sum_{p \in S^k} \log P(f_p \,|\, \mu_k)$$

multi-variate (i.e. $x, \mu \in R^N$)
Gaussian distribution
(simple special case $\Sigma = \sigma^2 \mathbf{I}$)

$$P(x|\mu) \;=\; \frac{1}{\sqrt{(2\pi\sigma^2)^N}} \exp - \frac{\|x-\mu\|^2}{2\,\sigma^2}$$

Towards soft clustering…

Gaussian Mixture Models (GMM)

Consider another **probabilistically motivated** approach to soft clustering…

point specific distributions S_p are replaced by fixed "prior" distribution ρ over clusters

$$\underbrace{\sum_k \rho_k \, P(x \mid \mu_k)}_{P_{gmm}(x|\rho,\mu)}$$

MLE estimation of GMM model parameters

sum of log-likelihoods (NLL) — GMM density with K modes → $P_{gmm}(x|\rho,\mu)$

$$E(\rho,\mu) \;=\; -\sum_p \log \sum_{k=1}^{K} \rho_k \, P(f_p \mid \mu_k)$$

segmentation variables S_p are **hidden** now ☹

can estimate according to the *Bayes* rule

$$S_p^k \;=\; \frac{\rho_k P(f_p|\mu_k)}{\sum_{i=1}^{K} \rho_i P(f_p|\mu_i)}$$

"hard" K-means

sum of log-likelihoods (NLL) — K single mode Gaussians

$$E(S,\mu) \;=\; -\sum_{k=1}^{K} \sum_{p \in S^k} \log P(f_p \mid \mu_k)$$

multi-variate (i.e. $x, \mu \in R^N$) Gaussian distribution (simple special case $\Sigma = \sigma^2 \mathbf{I}$)

$$P(x|\mu) \;=\; \frac{1}{\sqrt{(2\pi\sigma^2)^N}} \, \exp - \frac{\|x-\mu\|^2}{2\,\sigma^2}$$

Towards soft clustering…

Gaussian Mixture Models (GMM)

Consider another **probabilistically motivated** approach to soft clustering…

$$\sum_k \rho_k \, P(x \mid \mu_k, \Sigma_k) = P_{gmm}(x|\rho,\mu,\Sigma)$$

MLE estimation of GMM model parameters

sum of log-likelihoods (NLL) — GMM density with K modes

$$E(\rho,\mu,\Sigma) = -\sum_p \log \sum_{k=1}^{K} \rho_k \, P(f_p \mid \mu_k, \Sigma_k)$$

segmentation variables S_p are **hidden** now ☹

can estimate according to the *Bayes* rule

$$S_p^k = \frac{\rho_k P(f_p|\mu_k)}{\sum_{i=1}^{K} \rho_i P(f_p|\mu_i)}$$

"hard" K-means

sum of log-likelihoods (NLL) — K single mode Gaussians

$$E(S,\mu,\Sigma) = -\sum_{k=1}^{K} \sum_{p\in S^k} \log P(f_p \mid \mu_k, \Sigma_k)$$

"elliptic K-means"

multi-variate (i.e. $x, \mu \in R^N$) Gaussian distribution (general covariance matrix Σ)

$$P(x|\mu,\Sigma) = \frac{1}{\sqrt{(2\pi)^N |\Sigma|}} \exp - \frac{\|x-\mu\|_\Sigma^2}{2}$$

Using *Mahalanobis* distance $\|x-\mu\|_\Sigma^2 := (x-\mu)^T \Sigma^{-1} (x-\mu)$

Towards soft clustering…

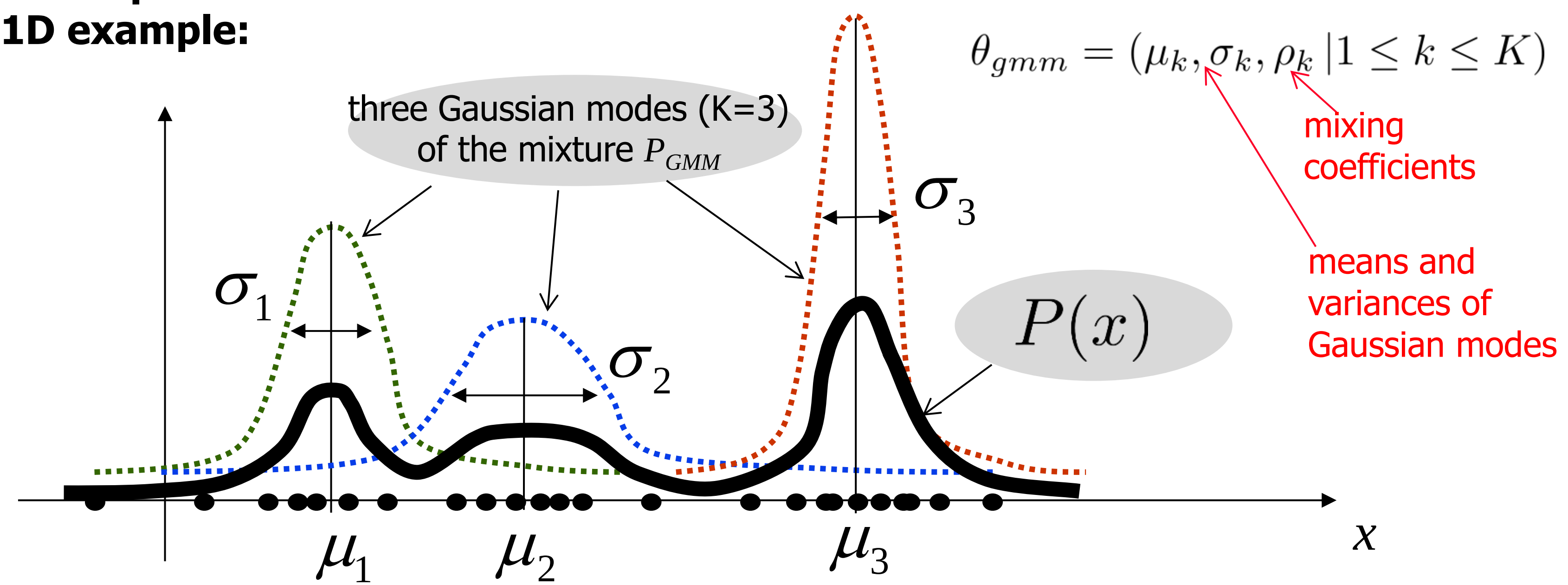
Gaussian Mixture Models (GMM)

- Soft clustering using Gaussian Mixture Model (GMM)

  - no “hard” assignments of points to K distinct (Gaussian) clusters S^k
  - all points are used to estimate parameters of one complex K-mode distribution (GMM)

GMMs estimate “true” data distributions
(continuous density analog of histograms)

simple 1D example:



GMM distribution: $$P_{gmm}(x \,|\, \theta) := \sum_k \rho_k \, P(x \,|\, \mu_k, \sigma_k)$$

Towards soft clustering…

Gaussian Mixture Models (GMM)

- Soft clustering using Gaussian Mixture Model (GMM)

  - no "hard" assignments of points to K distinct (Gaussian) clusters S^k
  - all points are used to estimate parameters of one complex K-mode distribution (GMM)

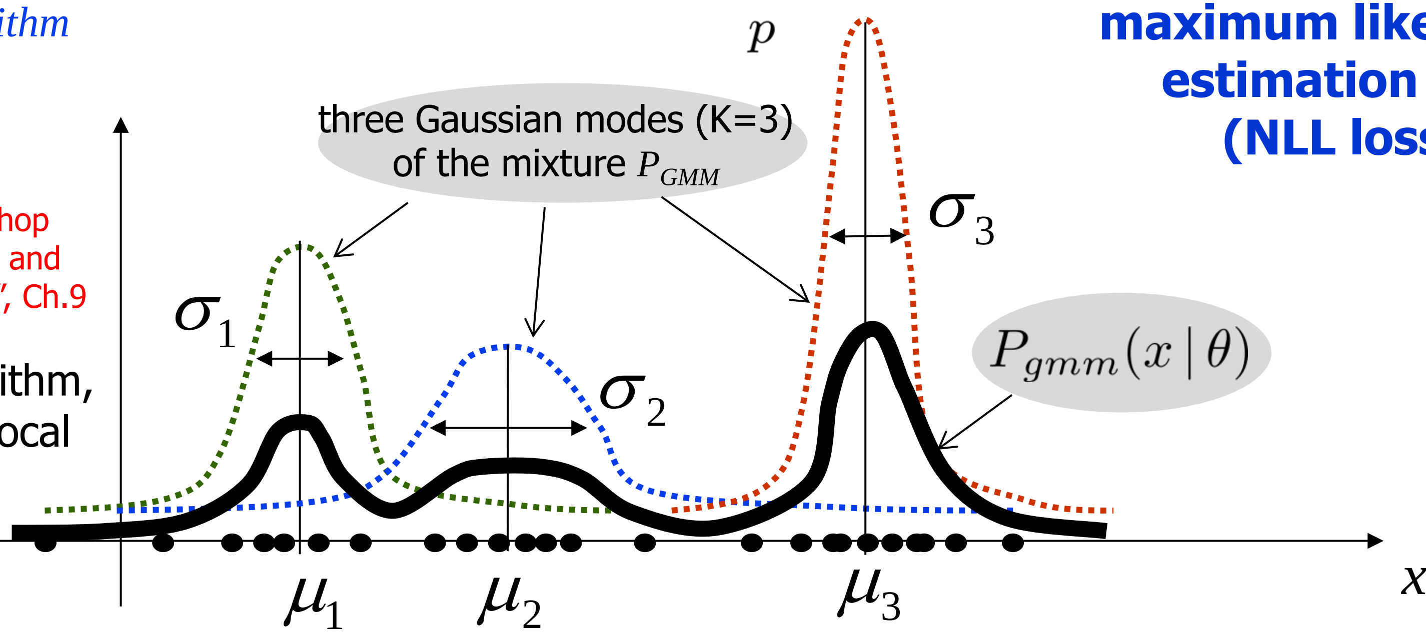
approximate optimization via *EM algorithm*

$$E_{gmm}(\theta) \; := \; -\sum_p \log P_{gmm}(x_p \,|\, \theta)$$

maximum likelihood estimation of θ (NLL loss)

see Szeliski Sec. 5.3.1 or Christopher Bishop "Pattern Recogn and Machine Learning", Ch.9

as Lloyd algorithm, sensitive to local minima



GMM distribution: $P_{gmm}(x \,|\, \theta) \; := \; \sum_k \rho_k \, P(x \,|\, \mu_k, \sigma_k)$

Towards soft clustering…

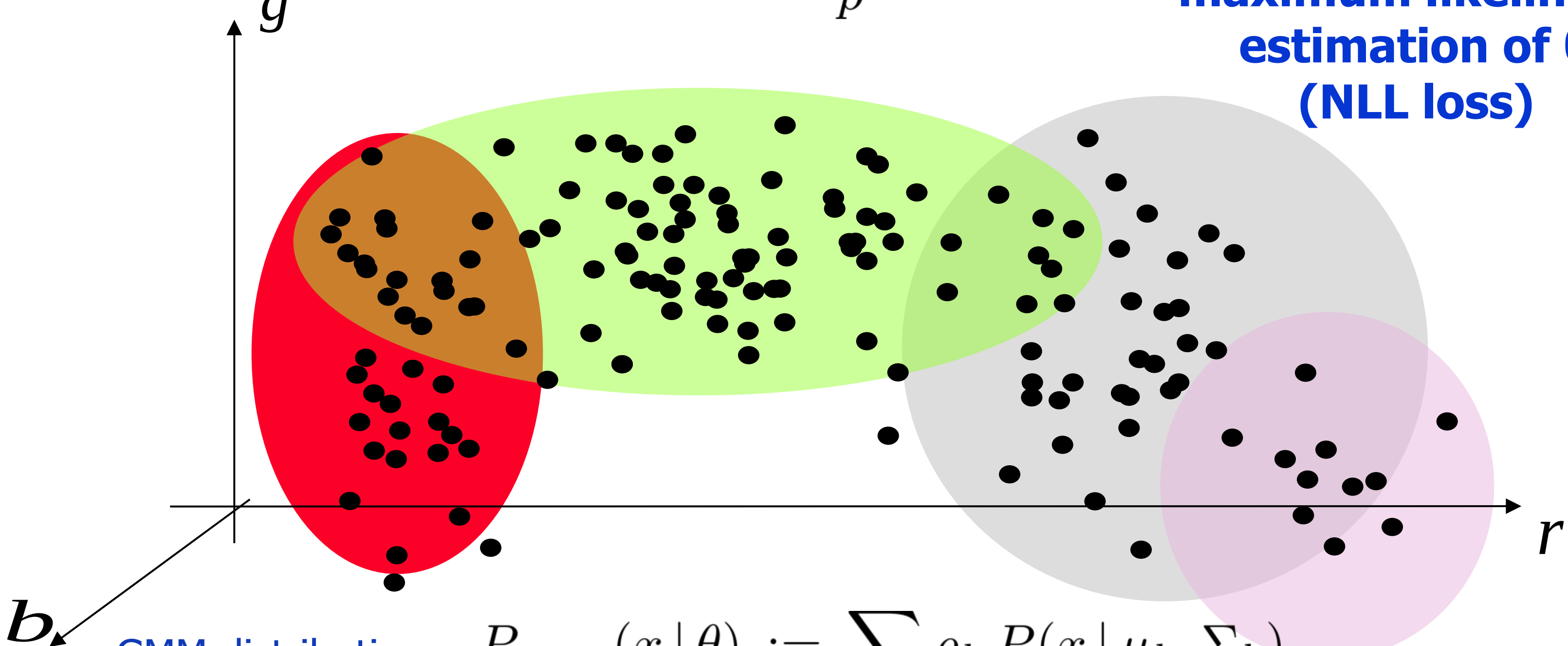
Gaussian Mixture Models (GMM)

- Soft clustering using Gaussian Mixture Model (GMM)

  - no "hard" assignments of points to K distinct (Gaussian) clusters S^k
  - all points are used to estimate parameters of one complex K-mode distribution (GMM)

$$E_{gmm}(\theta) := -\sum_p \log P_{gmm}(x_p \,|\, \theta)$$

maximum likelihood estimation of θ (NLL loss)



GMM distribution: $P_{gmm}(x \,|\, \theta) := \sum_k \rho_k \, P(x \,|\, \mu_k, \Sigma_k)$

OPTIONAL MATERIAL

UNIVERSITY OF WATERLOO

GMM estimation overview
Expectation-Maximization (EM)

GMM estimation - optimization of ML objective (sum of Negative Log Likelihoods, a.k.a. **NLL** loss)

$$E_{gmm}(\theta) := -\sum_p \log P_{gmm}(x_p \mid \theta) \equiv -\sum_p \log\left(\sum_k \rho_k P(x_p \mid \mu_k, \sigma_k)\right)$$

$$\stackrel{\forall S_p \in \Delta_K}{\equiv} -\sum_p \log\left(\sum_k S_p^k \frac{\rho_k P(x_p \mid \mu_k, \sigma_k)}{S_p^k}\right)$$

$\mathbf{E}_{S_p}$

In fact, **equality** holds specifically for

$$S_p^k = \frac{\rho_k P(x_p|\mu_k,\sigma_k)}{\sum_m \rho_m P(x_p|\mu_m,\sigma_m)}$$

(plug-in to check, very easy)

Jensen's inequality

move "log" inside expectation $\mathbf{E}$

$$\leq -\sum_p \underbrace{\sum_k S_p^k}_{\mathbf{E}_{S_p}} \log \frac{\rho_k P(x_p \mid \mu_k, \sigma_k)}{S_p^k}$$

$$= -\sum_p \sum_k S_p^k \log \rho_k - \sum_p \sum_k S_p^k \log P(x_p \mid \mu_k, \sigma_k) + \sum_p \sum_k S_p^k \log S_p^k$$

$$= -\sum_k \left(\sum_p S_p^k\right) \log \rho_k - \sum_k \sum_p S_p^k \log P(x_p \mid \mu_k, \sigma_k) - \sum_p \mathbf{H}(S_p)$$

entropy

GMM estimation overview

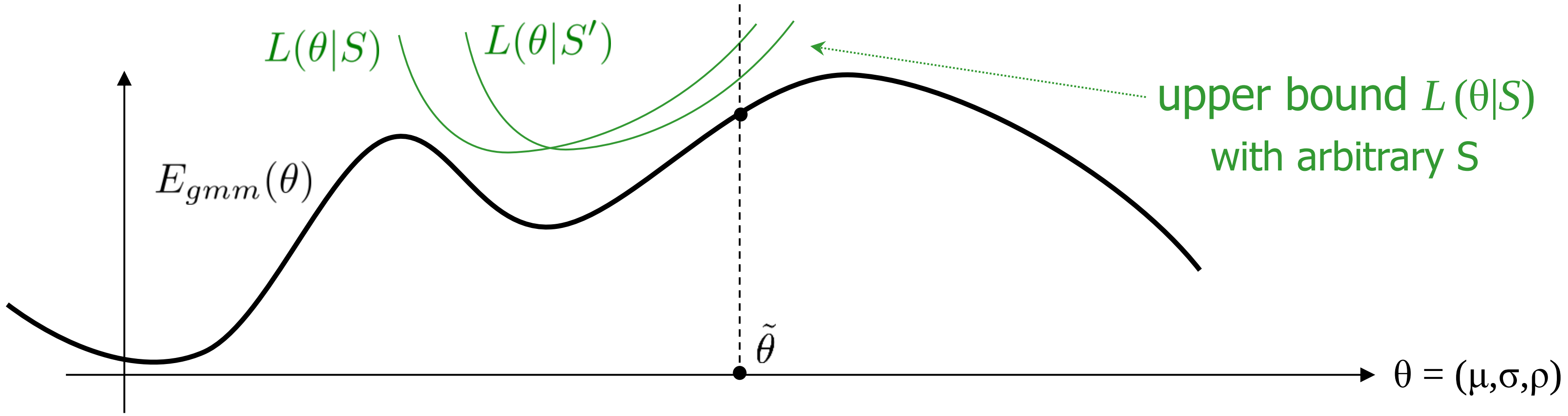
Expectation-Maximization (EM)

OPTIONAL MATERIAL

UNIVERSITY OF WATERLOO

GMM estimation - optimization of ML objective (sum of Negative Log Likelihoods, a.k.a. **NLL** loss)

$$E_{gmm}(\theta) \;:=\; -\sum_p \log P_{gmm}(x_p\,|\,\theta) \;\equiv\; -\sum_p \log\left(\sum_k \rho_k\, P(x_p\,|\,\mu_k,\sigma_k)\right)$$



$L(\theta|S)$ - for any S defines an upper bounds for $E_{gmm}(\theta)$

cluster cardinality term

fuzzy K-means loss (slide 54)

$$\leq\; -\sum_k \left(\sum_p S_p^k\right) \log \rho_k \;-\; \sum_k \sum_p S_p^k \log P(x_p\,|\,\mu_k,\sigma_k) \;-\; \sum_p \mathbf{H}(S_p)$$

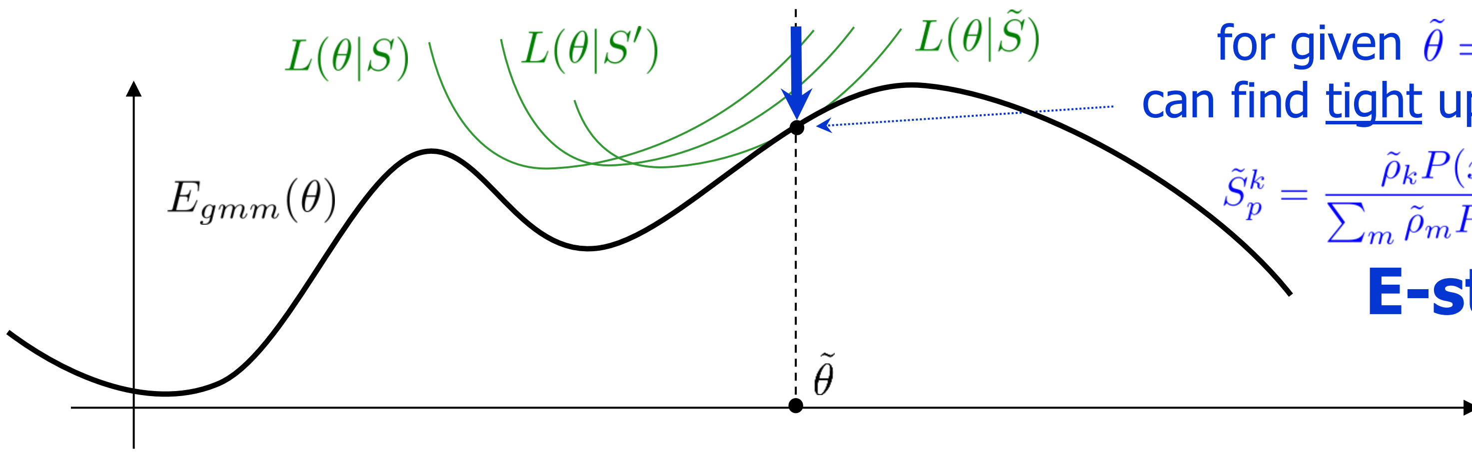
GMM estimation overview

Expectation-Maximization (EM)

OPTIONAL MATERIAL

GMM estimation - optimization of ML objective (sum of Negative Log Likelihoods, a.k.a. **NLL** loss)

$$E_{gmm}(\theta) \;:=\; -\sum_p \log P_{gmm}(x_p\,|\,\theta) \;\equiv\; -\sum_p \log\left(\sum_k \rho_k\, P(x_p\,|\,\mu_k,\sigma_k)\right)$$



for given $\tilde{\theta} = (\tilde{\mu}, \tilde{\sigma}, \tilde{\rho})$
can find tight upper bound

$$\tilde{S}_p^k = \frac{\tilde{\rho}_k P(x_p|\tilde{\mu}_k, \tilde{\sigma}_k)}{\sum_m \tilde{\rho}_m P(x_p|\tilde{\mu}_m, \tilde{\sigma}_m)}$$

E-step

$L(\theta|S)$ - for any S defines an upper bounds for $E_{gmm}(\theta)$

cluster cardinality term

fuzzy K-means loss (slide 54)

$$\le\; -\sum_k \left(\sum_p S_p^k\right) \log \rho_k \;-\; \sum_k \sum_p S_p^k \,\log P(x_p\,|\,\mu_k,\sigma_k) \;-\; \sum_p \mathbf{H}(S_p)$$

GMM estimation overview

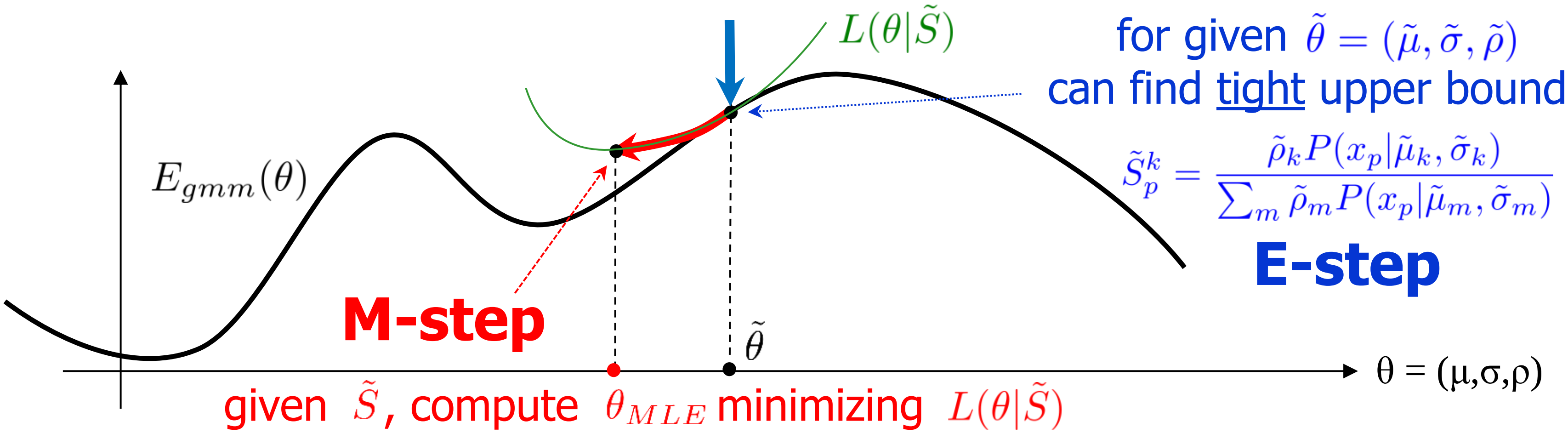
Expectation-Maximization (EM)

OPTIONAL MATERIAL

UNIVERSITY OF WATERLOO

GMM estimation - optimization of ML objective (sum of Negative Log Likelihoods, a.k.a. **NLL** loss)

$$E_{gmm}(\theta) := -\sum_p \log P_{gmm}(x_p \,|\, \theta) \;\equiv\; -\sum_p \log\left(\sum_k \rho_k\, P(x_p \,|\, \mu_k, \sigma_k)\right)$$



$L(\theta|S)$ - for any S defines an upper bounds for $E_{gmm}(\theta)$

cluster cardinality term

fuzzy K-means loss (slide 54)

$$\leq\; -\sum_k \left(\sum_p S_p^k\right) \log \rho_k \;-\; \sum_k \sum_p S_p^k \log P(x_p \,|\, \mu_k, \sigma_k) \;-\; \sum_p \mathbf{H}(S_p)$$

GMM estimation overview

Expectation-Maximization (EM)

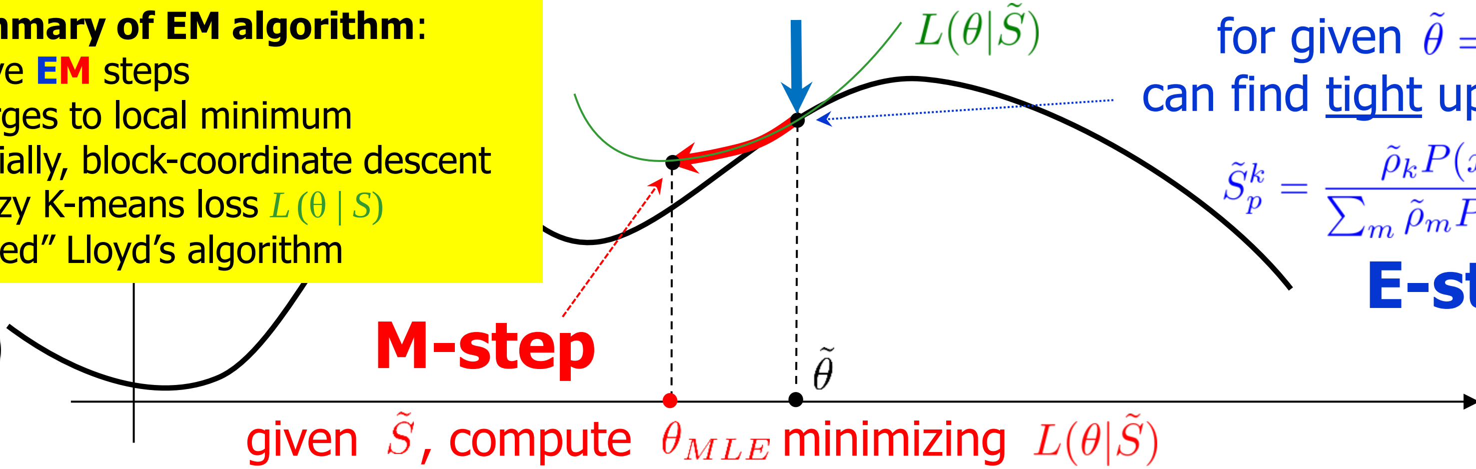
OPTIONAL MATERIAL

GMM estimation - optimization of ML objective (sum of Negative Log Likelihoods, a.k.a. **NLL** loss)

$$E_{gmm}(\theta) := -\sum_p \log P_{gmm}(x_p \,|\, \theta) \;\equiv\; -\sum_p \log\left(\sum_k \rho_k\, P(x_p \,|\, \mu_k, \sigma_k)\right)$$

Summary of EM algorithm:

- iterative **EM** steps
- converges to local minimum
- essentially, block-coordinate descent for fuzzy K-means loss $L(\theta \,|\, S)$
- "glorified" Lloyd's algorithm



for given $\tilde{\theta} = (\tilde{\mu}, \tilde{\sigma}, \tilde{\rho})$ can find tight upper bound

$$\tilde{S}_p^k = \frac{\tilde{\rho}_k P(x_p|\tilde{\mu}_k, \tilde{\sigma}_k)}{\sum_m \tilde{\rho}_m P(x_p|\tilde{\mu}_m, \tilde{\sigma}_m)}$$

E-step

$L(\theta|S)$ - for any S defines an upper bounds for $E_{gmm}(\theta)$

cluster cardinality term — fuzzy K-means loss (slide 54)

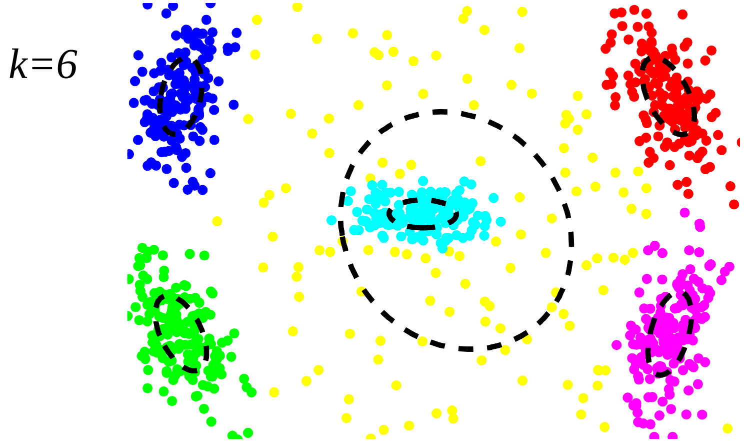
$$\le -\sum_k \left(\sum_p S_p^k\right) \log \rho_k \;-\; \sum_k \sum_p S_p^k \log P(x_p \,|\, \mu_k, \sigma_k) \;-\; \sum_p \mathbf{H}(S_p)$$

Gaussian clusters/modes in:

(basic) K-means vs. GMM (or fuzzy K-means)

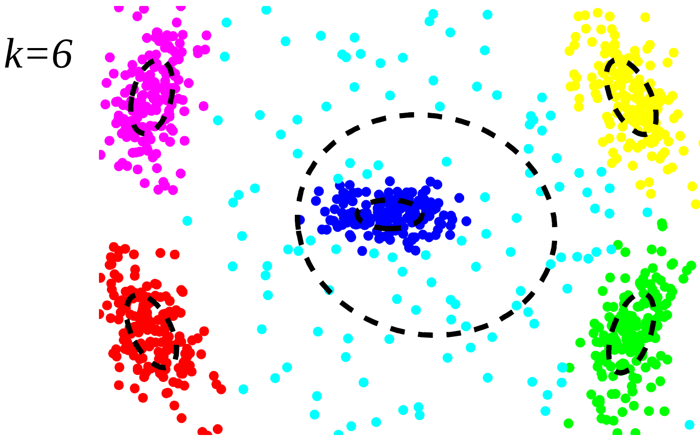
- *hard* assignment to clusters
  - separates data points into multiple Gaussian blobs
- only estimates means μ_i
  - (co)variance Σ_i can also be treated as cluster parameter (*elliptic K-means*) if using Gaussian log-likelihoods

- *soft* mode searching
  - estimates data distribution with multiple Gaussian modes
- estimates both mean μ_i and (co)variance Σ_i for each mode



(elliptic) K-means

color indicates assigned cluster



GMM

color indicates locally strongest mode

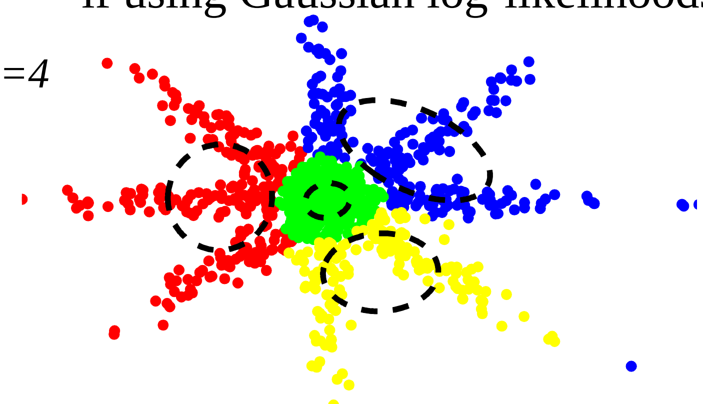
UNIVERSITY OF WATERLOO

Gaussian clusters/modes in:

(basic) K-means vs. GMM (or fuzzy K-means)

- *hard* assignment to clusters
  - separates data points into multiple Gaussian blobs
- only estimates means μ_i
  - (co)variance Σ_i can also be treated as cluster parameter (*elliptic K-means*) if using Gaussian log-likelihoods

k=4

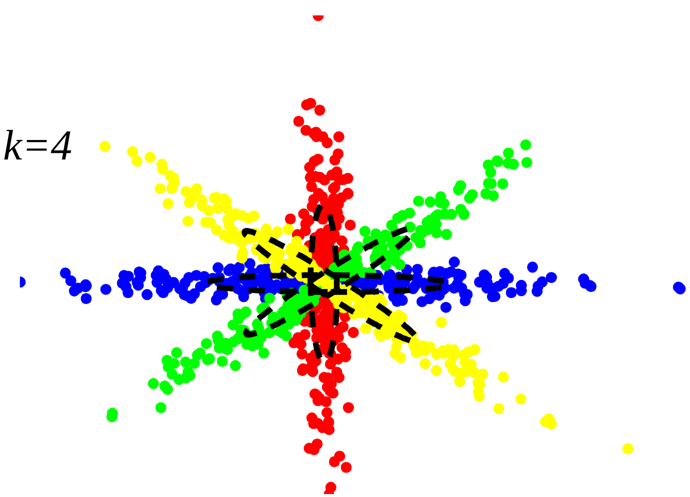


hard clustering may not work well when clusters overlap

(may not be a problem in image segmentation, since objects do not "overlap" in RGBXY)

- *soft* mode searching
  - estimates data distribution with multiple Gaussian modes
- estimates both mean μ_i and (co)variance Σ_i for each mode

k=4



While this is an optimal GMM, standard EM may converge to a bad solution (local minimum)

Gaussian clusters/modes in:

(basic) K-means vs. GMM (or fuzzy K-means)

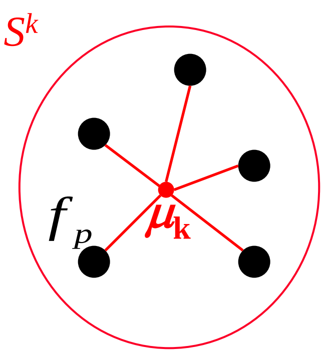
- *hard* assignment to clusters
  - separates data points into multiple Gaussian blobs
- only estimates means μ_i
  - (co)variance Σ_i can also be treated as cluster parameter (*elliptic K-means*) if using Gaussian log-likelihoods
- computationally cheap steps (block-coordinate descent, Lloyd's algorithm) **unless estimating covariances** Σ_k (elliptic case)
- sensitive to local minima
- **(implicitly) extends to high dimensional features** (*kernel K-means, non-parametric clustering*)

- *soft* mode searching
  - estimates data distribution with multiple Gaussian modes
- estimates both mean μ_i and (co)variance Σ_i for each mode
- expensive steps (mostly due to Σ_k) (iterative EM algorithm)
- sensitive to local minima
- **becomes slow to estimate Σ from high dimensional data, also needs lots of points**

K-means as non-parametric clustering

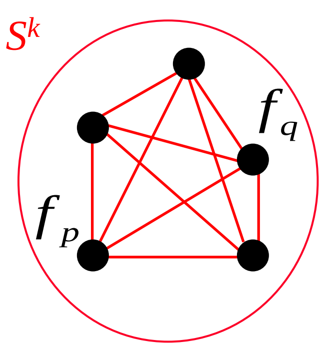
just plug-in expression

$$\mu_k = \frac{1}{|S^k|} \sum_{q \in S^k} f_q$$



$$E(S,\mu) \;=\; \sum_{k=1}^{K} \sum_{p \in S^k} \| f_p - \mu_k \|^2$$

equivalent (easy to check)

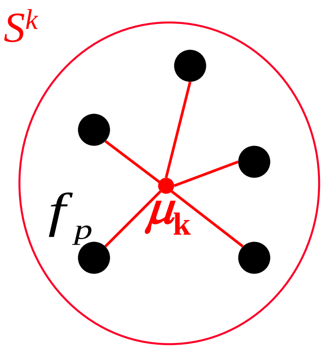


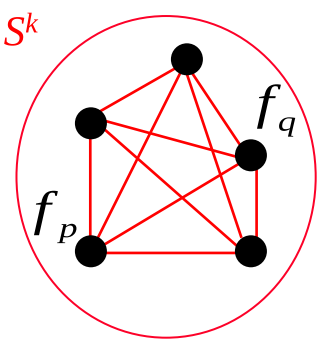
$$E(S) \;=\; \sum_{k=1}^{K} \sum_{pq \in S^k} \frac{\| f_p - f_q \|^2}{2|S^k|}$$

equivalent criterion without parameters μ_k

sample variance: $\mathrm{var}(S^k) = \frac{1}{|S^k|} \sum_{p \in S^k} \| f_p - \mu_k \|^2 = \frac{1}{2|S^k|^2} \sum_{pq \in S^k} \| f_p - f_q \|^2$

K-means as variance clustering criteria





both formulas can be written as

$$E(S) = \sum_{k=1}^{K} |S^k| \, \mathbf{var}(S^k)$$

sample variance: $\operatorname{var}(S^k) = \frac{1}{|S^k|} \sum_{p \in S^k} \| f_p - \mu_k \|^2 = \frac{1}{2|S^k|^2} \sum_{pq \in S^k} \| f_p - f_q \|^2$

K-means Summary

Good

- Principled (objective function) approach to clustering
- Simple to implement (the approximate iterative optimization)
- Fast

Not so good

- Only a local minimum is found   (sensitive to initialization)
- May fail for non-blob like clusters
- Maybe sensitive to outliers
- How to choose ***K*** ?

Can add **sparsity/complexity** term making K an additional variable

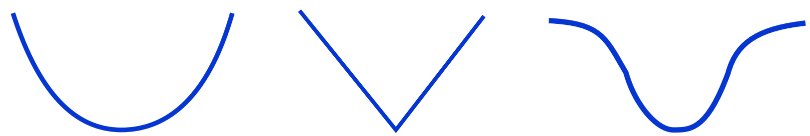
$$E(S,\mu,K) \;=\; \sum_{k=1}^{K} \sum_{p \in S^k} \|f_p - \mu_k\|^2 \;+\; \gamma\,|K|$$

Akaike Information Criterion (AIC) or
Bayesian Information Criterion (BIC)

(summary of)
Standard extensions of K-means:

- **Parametric:** with arbitrary *distortion* measure $\|\cdot\|_d$ (***distortion clustering***)

$$\sum_{k=1}^{K}\sum_{p\in S^k}\|f_p-\mu_k\|_d$$



Examples of $\|\cdot\|_d$: *quadratic* (*K-means*) *absolute* (*K-medians*) *bounded* (*K-modes*)

$-\log P(x\,|\,\theta)\sim\|x-\theta\|$

- **Parametric:** with arbitrary likelihoods $P(\cdot|\theta)$ (***probabilistic K-means***) *[Kearns, Mansour & Ng, UAI'97]*

$$-\sum_{k=1}^{K}\sum_{p\in S^k}\log P(f_p\,|\,\theta_k)$$

Examples of $P(\cdot|\theta)$: *Normal, gamma, exponential, Gibbs, etc.*

Normal → basic or elliptic K-means

- **Non-parametric:** with any *affinity or similarity measure, a.k.a. kernel* $k(x,y)$ (***kernel K-means***, *average association, average distortion, normalized cut*)

$$\sum_{k=1}^{K}\frac{\sum_{pq\in S^k}\|f_p-f_q\|^2}{2|S^k|}=const-\sum_{k=1}^{K}\frac{\sum_{pq\in S^k}\langle f_p,f_q\rangle}{|S^k|}\;\Rightarrow\;-\sum_{k=1}^{K}\frac{\sum_{pq\in S^k}k(f_p,f_q)}{|S^k|}$$

generalize from dot-product to "**inner product**" or arbitrary **affinity measure** or **kernel** k

From basic K-means to *kernel K-means*

(example: **Gaussian kernel** and its **robust metric** story)

easy to show

same as a Problem in
K-means part of HW4

minimize distances within clusters

maximize affinities within clusters

$$\sum_{k=1}^{K} \frac{\sum_{pq \in S^k} ||f_p - f_q||_K^2}{2|S^k|} = const - \sum_{k=1}^{K} \frac{\sum_{pq \in S^k} k(f_p, f_q)}{|S^k|}$$

for any kernel-induced metric:

$$||f_p - f_q||_K^2 := k(f_p, f_p) + k(f_q, f_q) - 2k(f_p, f_q)$$

NOTE: this is proper *metric* for any pos. def. kernels
e.g. works for *inner products*

Examples:

$$k(f_p, f_q) = \langle f_p, f_q \rangle$$

basic (linear) kernel

$\Longrightarrow$

squared L2 distance in standard K-means

$$||f_p - f_q||^2 = \langle f_p, f_p \rangle + \langle f_q, f_q \rangle - 2\langle f_p, f_q \rangle = \langle f_p - f_q, f_p - f_q \rangle$$

squared Euclidean distance

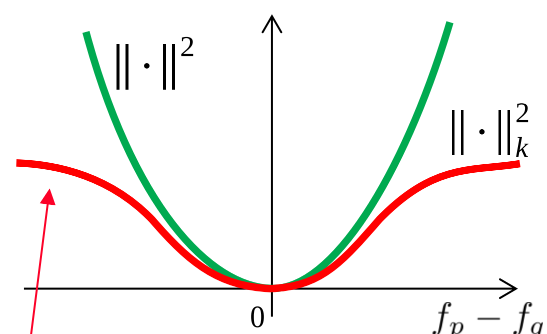
$$k(f_p, f_q) = e^{-\frac{\|f_p - f_q\|^2}{2\sigma^2}}$$

Gaussian kernel

$\Longrightarrow$

distance in Gaussian kernel K-means

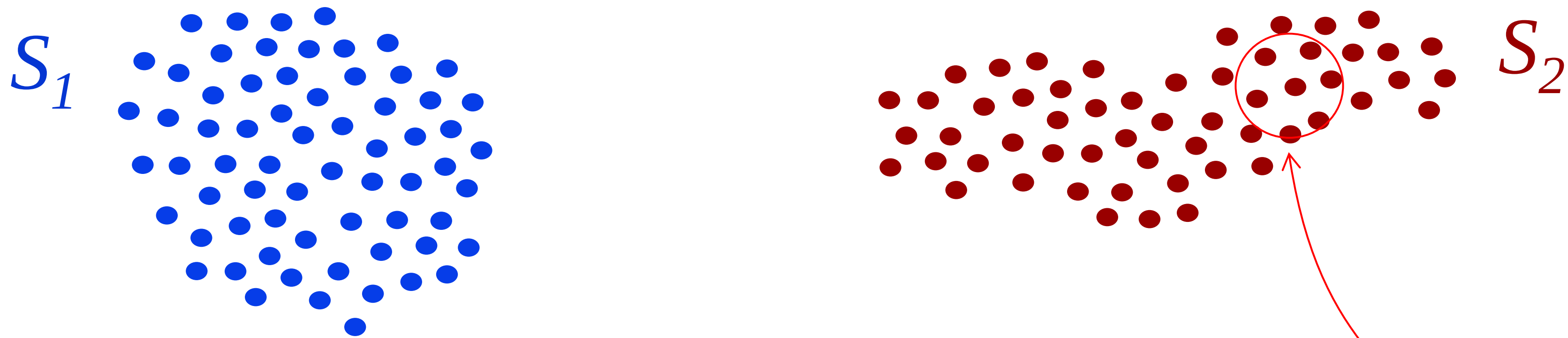
$$||f_p - f_q||_K^2 = 2\left(1 - e^{-\frac{\|f_p - f_q\|^2}{2\sigma^2}}\right)$$



robust metric focuses on **local distortion** (deemphasizes larger distances)

From basic K-means to *kernel K-means*

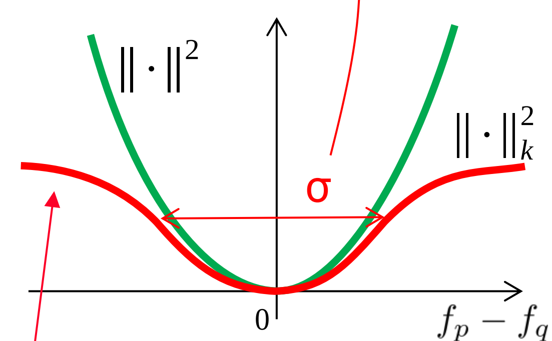
(example: **Gaussian kernel** and its **robust metric** story)



$$\sum_{pq\in S_1} ||f_p - f_q||^2 \ll \sum_{pq\in S_2} ||f_p - f_q||^2$$

$$\sum_{pq\in S_1} ||f_p - f_q||_K^2 \approx \sum_{pq\in S_2} ||f_p - f_q||_K^2$$

For Gaussian kernel both clusters look **equally tight/compact** since it "inspects" only their **neighborhoods of size σ**.



robust metric focuses on **local distortion** (deemphasizes larger distances)

A: local minima C: circular cluster

B: arc-shape gap D: thin/long cluster

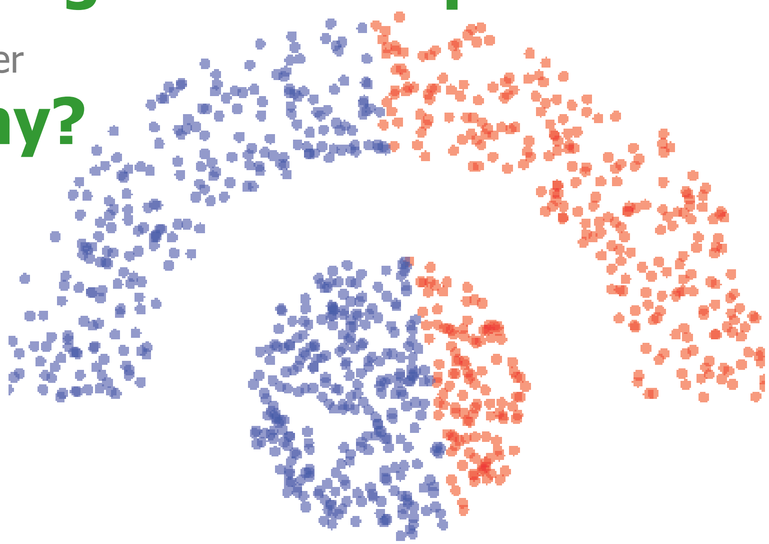
Basic K-means vs kernel K-means

$$\sum_{k=1}^{K} \frac{\sum_{pq \in S^k} ||f_p - f_q||_K^2}{2|S^k|} = const - \sum_{k=1}^{K} \frac{\sum_{pq \in S^k} k(f_p, f_q)}{|S^k|}$$

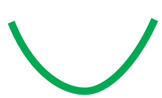
iClicker

Why?

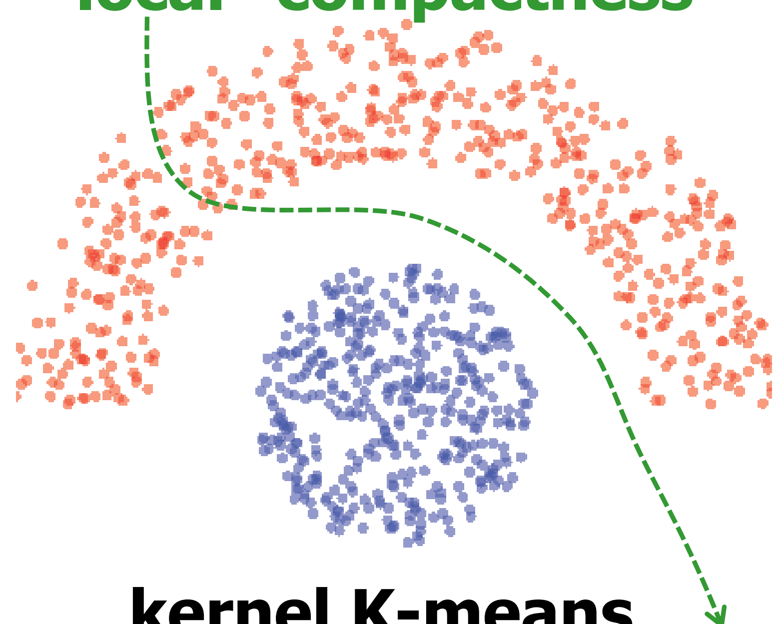
global "compactness"



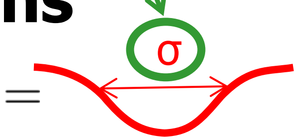
basic K-means

$||f_p - f_q||_K =$ 

local "compactness"



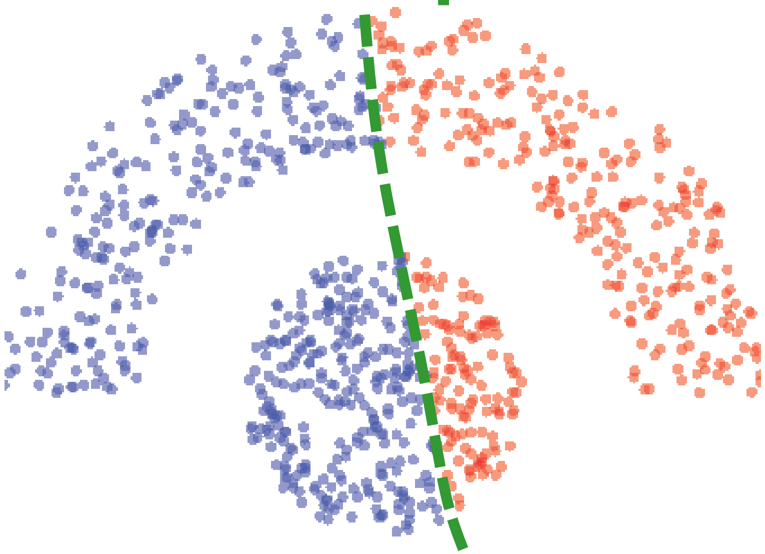
kernel K-means

e.g. Gaussian kernel => $||f_p - f_q||_K =$ 

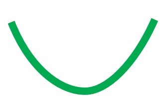
Basic K-means vs kernel K-means

$$\sum_{k=1}^{K} \frac{\sum_{pq\in S^k} ||f_p - f_q||_K^2}{2|S^k|} \quad = \quad const - \sum_{k=1}^{K} \frac{\sum_{pq\in S^k} k(f_p, f_q)}{|S^k|}$$

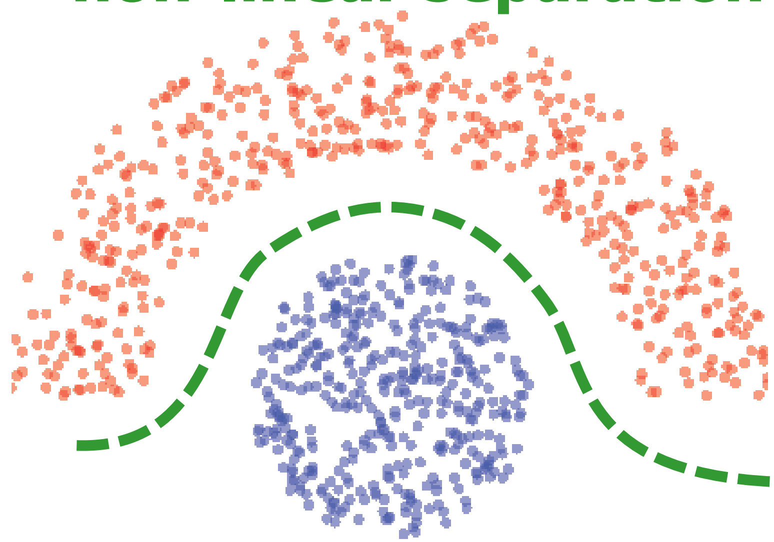
linear separation



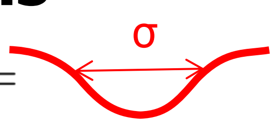
basic K-means

$||f_p - f_q||_K =$ 

non-linear separation



kernel K-means

e.g. Gaussian kernel => $||f_p - f_q||_K =$ 

kernel K-means

non-parametric (kernel) clustering

$$-\sum_{k=1}^{K} \frac{\sum_{pq \in S^k} k(f_p, f_q)}{|S^k|}$$

- objective

Kernel-based clustering (a.k.a. pairwise clustering):

- robustness to outliers

- non-parametric approach, arbitrary separation boundary, assumes only "local compactness" instead of fitting parameters of distributions (of known class) to clusters

- there are known biases, **many variants** addressing them

- **optimization?** (no block-coordinate descent as we dropped cluster parameters)

kernel K-means

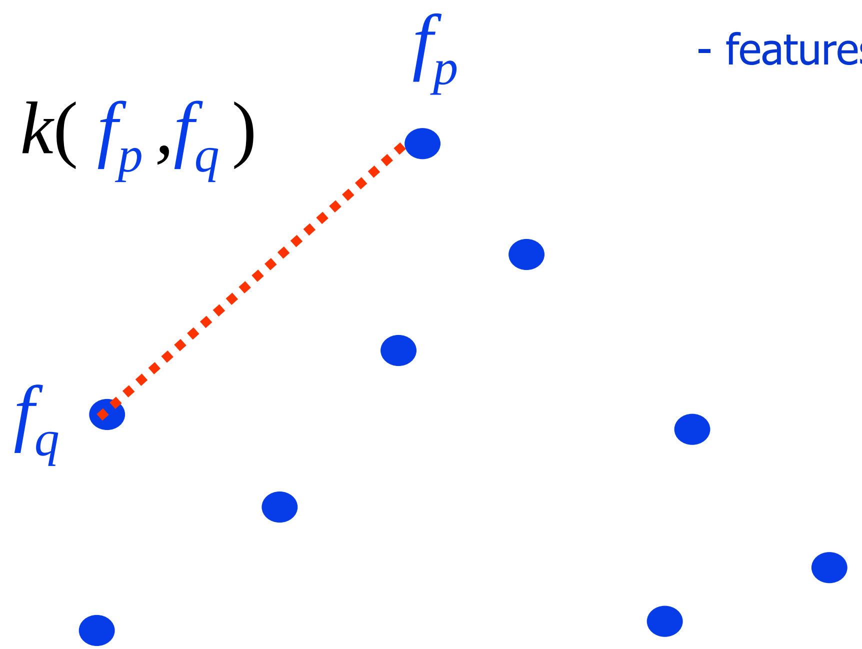
non-parametric (kernel) clustering

$$-\sum_{k=1}^{K} \frac{\sum_{pq \in S^k} k(f_p, f_q)}{|S^k|}$$

- objective

- features

$$A_{pq} = k(f_p, f_q)$$



kernel K-means

non-parametric (kernel) clustering

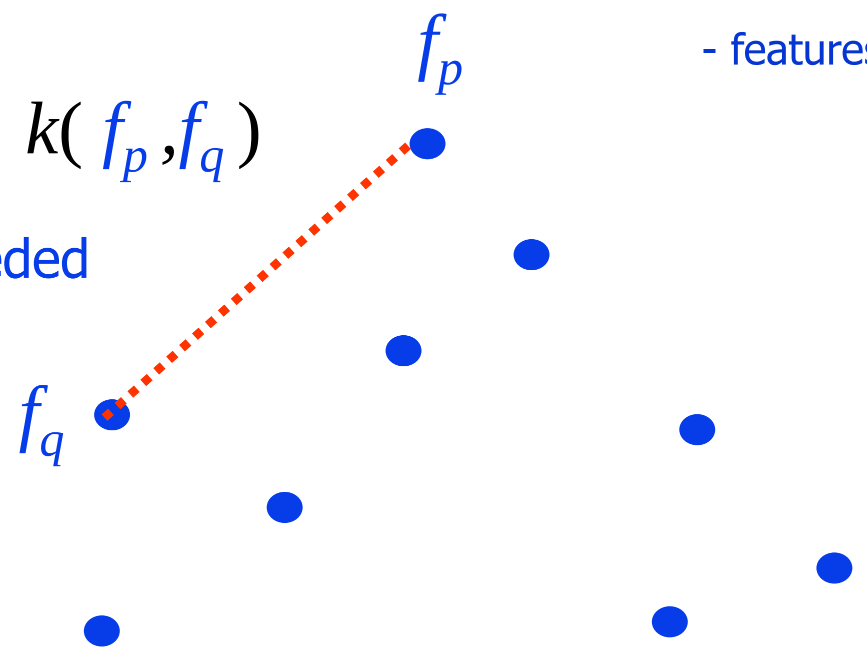
$$- \sum_{k=1}^{K} \frac{\sum_{pq \in S^k} A_{pq}}{|S^k|}$$

- objective

- features

$$A_{pq} = k(f_p, f_q)$$

<u>explicit</u> features f_p are no longer needed



kernel K-means

non-parametric (kernel) clustering

$$-\sum_{k=1}^{K} \frac{\sum_{pq \in S^k} A_{pq}}{|S^k|}$$

- objective

only need
affinity (or kernel) matrix

$A = [A_{pq}]$

(finite dimensional version of) **MERCER THEOREM**

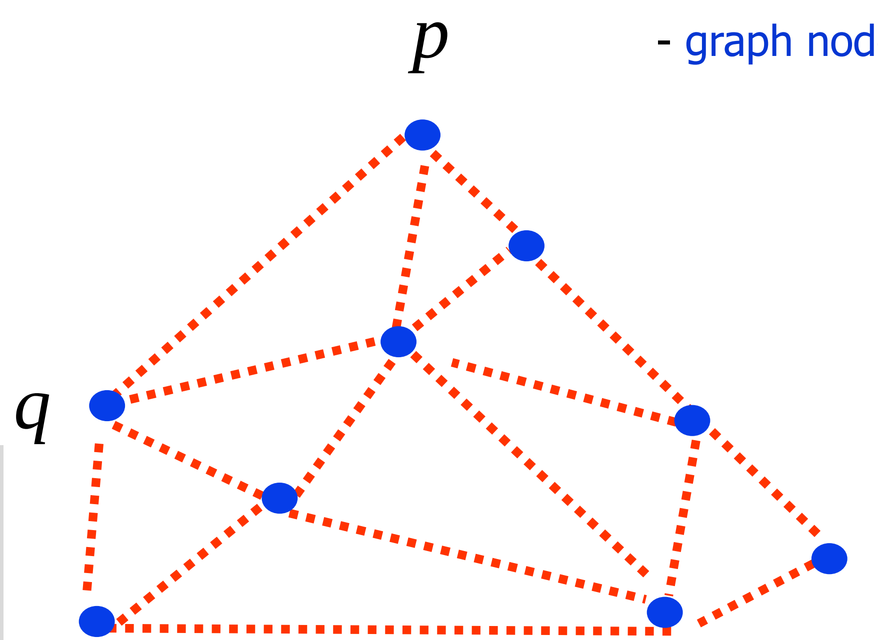
if needed, can find "embedding" $\{\phi_p\}$

s.t. $A_{pq} = \langle \phi_p, \phi_q \rangle$

using eigen decomposition for p.s.d. A

(problem from HW4)

- graph nodes



Essentially, we formulated a
graph clustering problem

kernel K-means

non-parametric (kernel) clustering

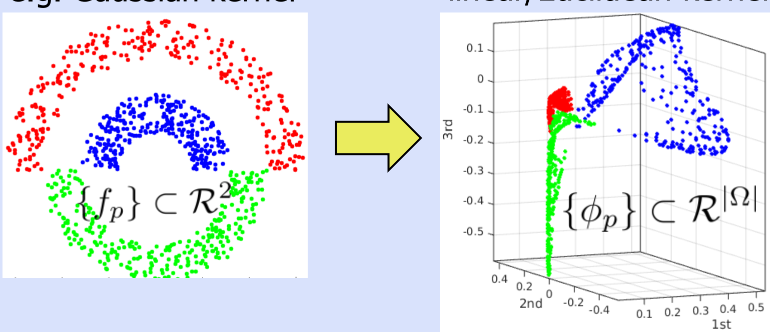
$$-\sum_{k=1}^{K} \frac{\sum_{pq \in S^k} A_{pq}}{|S^k|}$$

- objective

<u>high-dimensional isometric *Euclidean* embedding "story"</u>

$k(f_p, f_q) \;=\; A_{pq}$ = $A_{pq} \;=\; \langle \phi_p, \phi_q \rangle$

e.g. Gaussian kernel — linear/Euclidean kernel

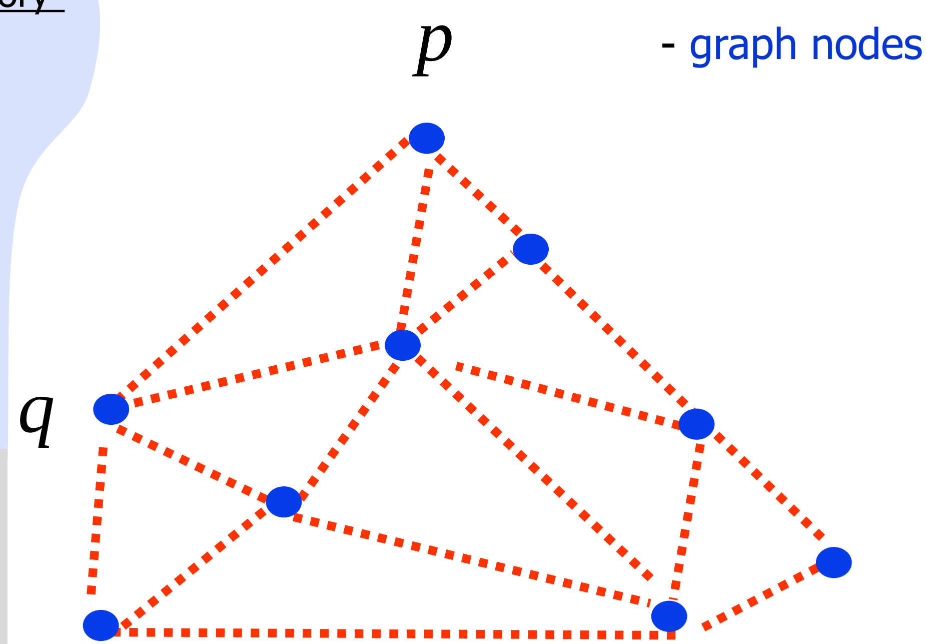


if needed, can find "embedding" $\{\phi_p\}$

s.t. $A_{pq} \;=\; \langle \phi_p, \phi_q \rangle$

using eigen decomposition for p.s.d. A

(problem from HW4)



Essentially, we formulated a **graph clustering problem**

Optimization for kernel clustering
(brief overview, details are left for homework 4)

- Idea 1: find (Euclidean) embedding $\{\phi_p\}$ s.t.

$$\langle \phi_p, \phi_q \rangle = A_{pq}$$

eigen decomposition of A (p.s.d)
(HW4 problem)

and use basic K-means (Lloyd's algorithm) over points $\{\phi_p\}$.

Problem: in general $\{\phi_p\} \subset \mathcal{R}^{|\Omega|}$ where $|\Omega|$ is the size of the data set

- Idea 2 **[spectral clustering]**: find embedding $\{\tilde{\phi}_p\}$ s.t.

$$\langle \tilde{\phi}_p, \tilde{\phi}_q \rangle = \tilde{A}_{pq}$$

(HW4 problem)

where $\tilde{A}$ is a <u>*low rank approximation*</u> of A (of any rank m).

[*a la* **Eckart-Young-Mirsky theorem** in Topic 7].

In this case can check $\{\tilde{\phi}_p\} \subset \mathcal{R}^m$ and K-means over $\{\tilde{\phi}_p\}$ is practical (for smaller m).

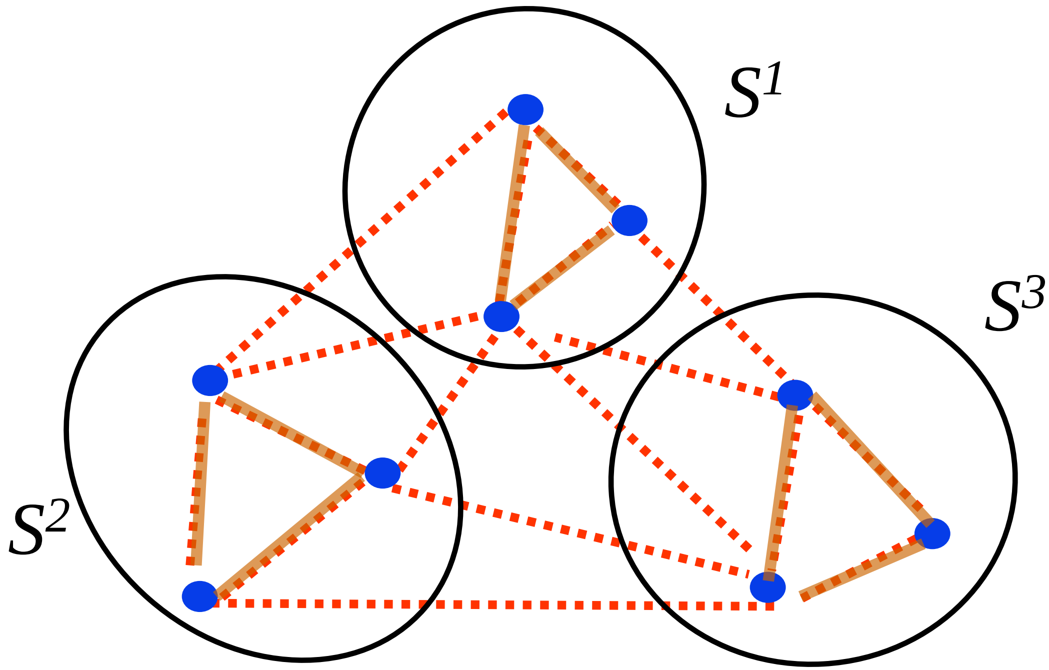
NOTE (relation to PCA): low-rank approximation $\tilde{A}$ keeps the most significant (m largest) eigen values of A.

kernel K-means or ***average association***

non-parametric (kernel) clustering

"self-association" of cluster S^k

$$E(S) = -\sum_{k=1}^{K} \frac{\sum_{pq \in S^k} A_{pq}}{|S^k|}$$



kernel K-means or ***average association***

non-parametric (kernel) clustering

$$E(S) = -\sum_{k=1}^{K} \frac{\sum_{pq \in S^k} A_{pq}}{|S^k|} \qquad \sum_{pq \in S^k} A_{pq} \equiv S^{k\prime} A S^k$$

in matrix notation:

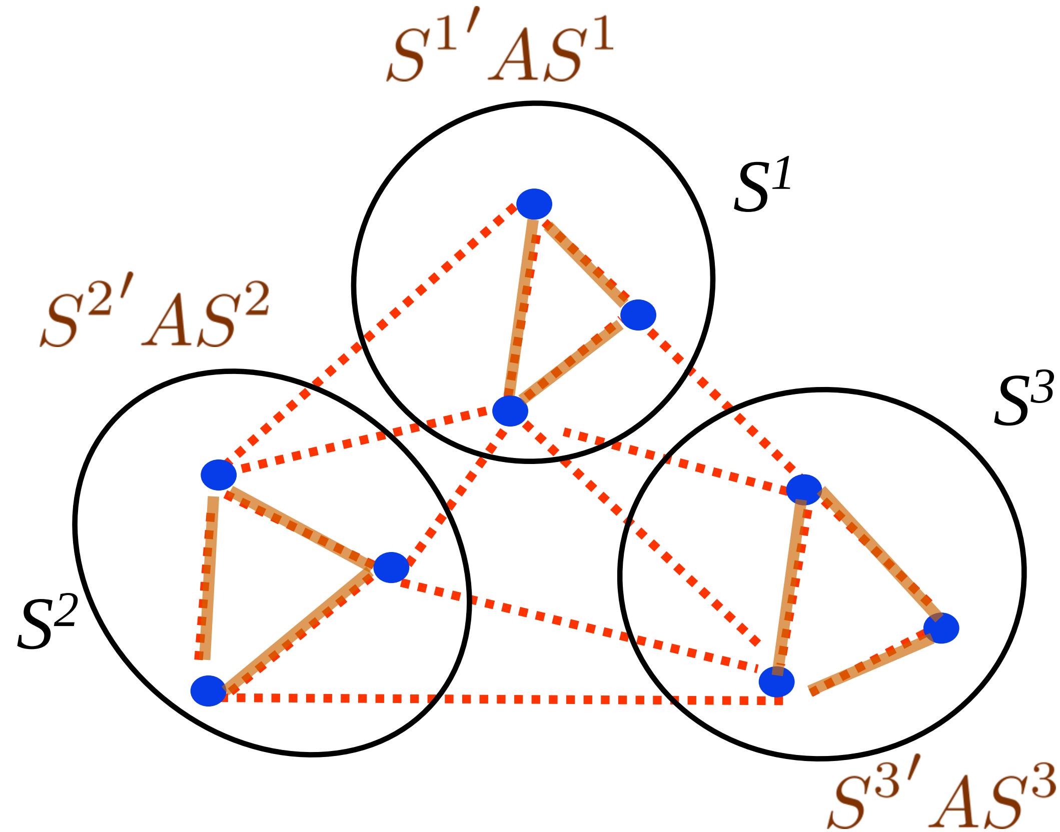
S^k - indicator vector

' means *transpose*

node indices

	1	2	3	4	5	6	7	8	9
S^1 =	[1	1	1	0	0	0	0	0	0]
S^2 =	[0	0	0	1	1	1	0	0	0]
S^3 =	[0	0	0	0	0	0	1	1	1]

assume clusters are represented by *indicator vectors S^k*



kernel K-means or ***average association***

non-parametric (kernel) clustering

$$E(S) = -\sum_{k=1}^{K} \frac{\qquad\qquad}{|S^k|}$$

$S^{k'} A S^k$

in matrix notation:

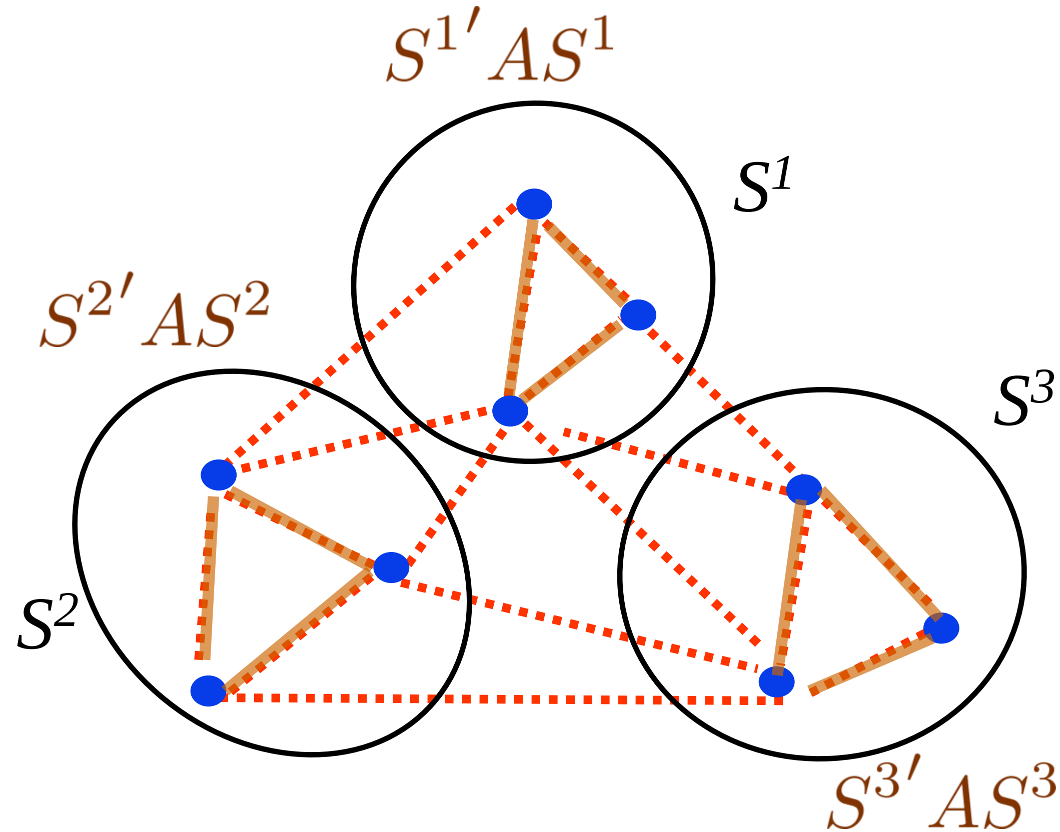
S^k - indicator vector

' means *transpose*

node indices

	1	2	3	4	5	6	7	8	9
S^1 =	[1	1	1	0	0	0	0	0	0]
S^2 =	[0	0	0	1	1	1	0	0	0]
S^3 =	[0	0	0	0	0	0	1	1	1]

assume clusters are represented by *indicator vectors* S^k



kernel K-means or ***average association***

non-parametric (kernel) clustering

$$E(S) = -\sum_{k=1}^{K} \frac{S^{k\prime} A S^k}{|S^k|}$$

in matrix notation:

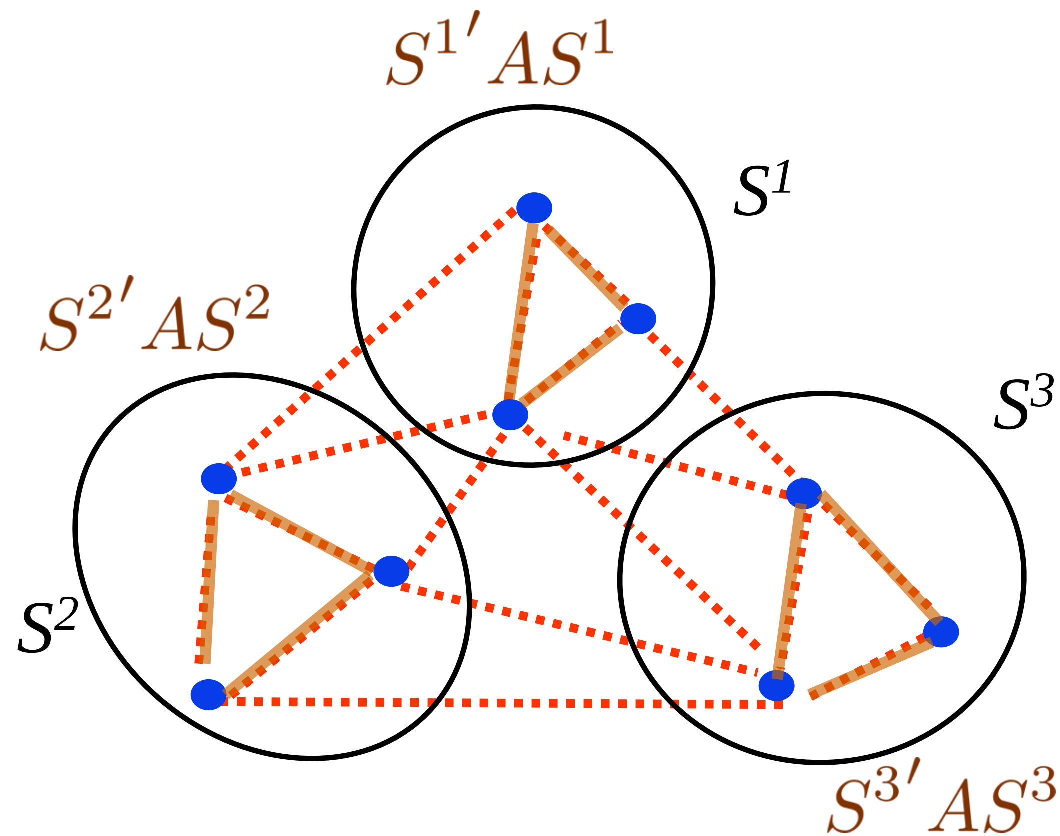
S^k - indicator vector

' means *transpose*

node indices

	1	2	3	4	5	6	7	8	9
S^1 =	[1	1	1	0	0	0	0	0	0]
S^2 =	[0	0	0	1	1	1	0	0	0]
S^3 =	[0	0	0	0	0	0	1	1	1]

assume clusters are represented by *indicator vectors S^k*



Convenient general notation

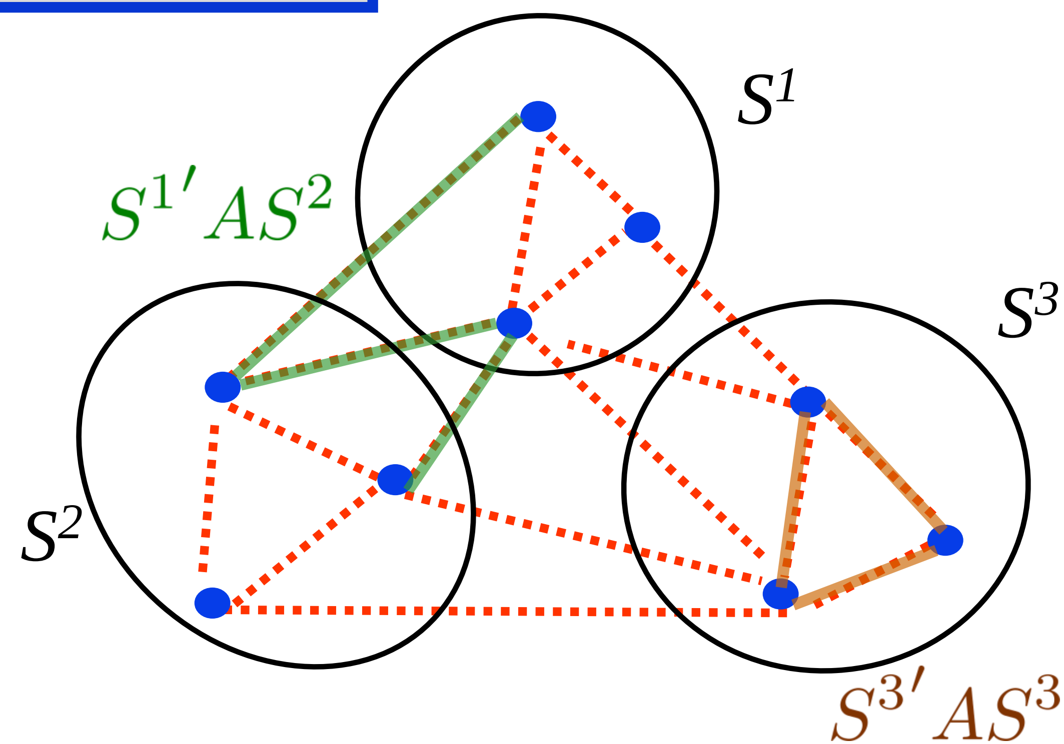
$$S^{i\prime} A S^j \equiv \sum_{p \in S^i, q \in S^j} A_{pq}$$

sum of all graph edge weights A_{pq} from set S^i to set S^j

node indices

	1	2	3	4	5	6	7	8	9
$S^1 =$	[1	1	1	0	0	0	0	0	0]
$S^2 =$	[0	0	0	1	1	1	0	0	0]
$S^3 =$	[0	0	0	0	0	0	1	1	1]

assume clusters are represented by *indicator vectors* S^k



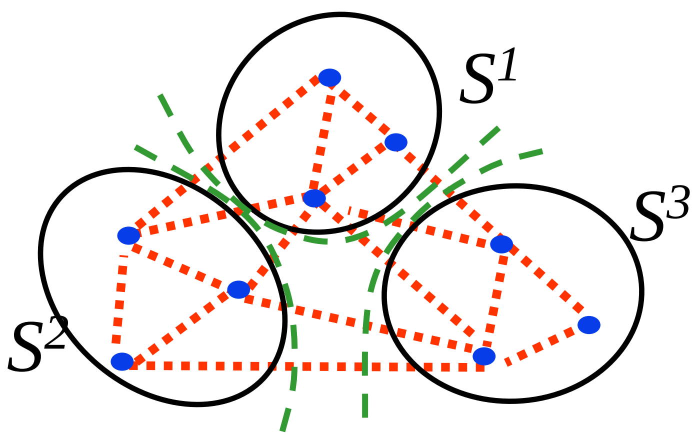
Other graph clustering objectives

related to Cheeger cut,
spectral graph theory,
isoperimetic constant, etc

Average Cut

"cut" for S^k

$$\sum_{k=1}^{K} \frac{S^{k\prime} A\,(\mathbf{1} - S^k)}{|S^k|}$$

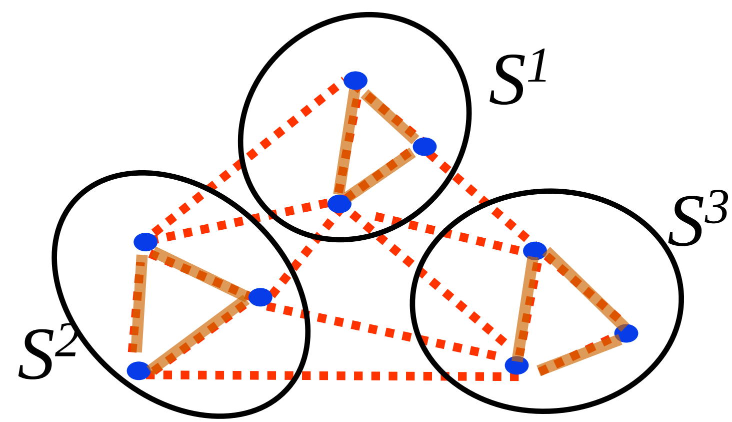


so far only looked at

Average Association

"self-association" for S^k

$$-\sum_{k=1}^{K} \frac{S^{k\prime} A\, S^k}{|S^k|}$$



Typically, approximately optimized via spectral methods using strongest *eigenvalues / eigenvectors of A*

Other graph clustering objectives

- **Average Association**

$$-\sum_{k=1}^{K} \frac{S^{k\prime} A\, S^k}{|S^k|}$$

- **Average Cut**

$$\sum_{k=1}^{K} \frac{S^{k\prime} A\,(\mathbf{1} - S^k)}{|S^k|}$$

$$|S^k| = \mathbf{1}' S^k$$

- **Normalized Cut**

[Shi & Malik, 2000]

$$\sum_{k=1}^{K} \frac{S^{k\prime} A\,(\mathbf{1} - S^k)}{d' S^k} \equiv K - \sum_{k=1}^{K} \frac{S^{k\prime} A\, S^k}{d' S^k}$$

for $d := A\mathbf{1}$

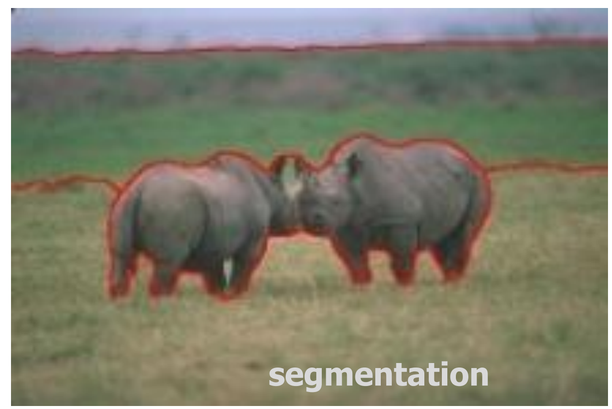
vector of node **degrees** (connectivity)

Basic K-means vs Kernel Clustering

$$E(S) = -\sum_{k=1}^{K} \frac{S^{k'} A S^k}{|S^k|}$$

compact "blobs" in RGB**XY** space

[Achanta et al., PAMI 2011]



[Shi&Malik 2000]

"segments" in RGB**XY** space

not just super-pixels

basic K-means

for $A_{pq} = \langle f_p, f_q \rangle$

kernel K-means

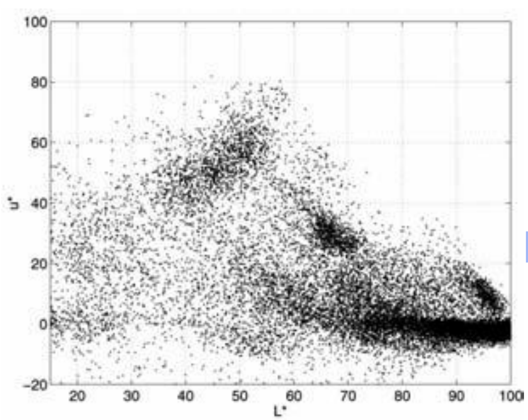
e.g. for Gaussian kernel $A_{pq} = \exp - \frac{\|f_p - f_q\|^2}{2\sigma^2}$

From “**means**” towards “**modes**” clustering:

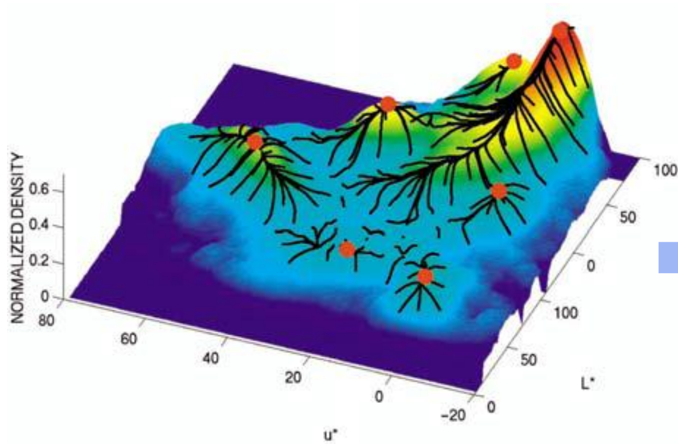
Kernel-based ***mode clustering***

Optional Material

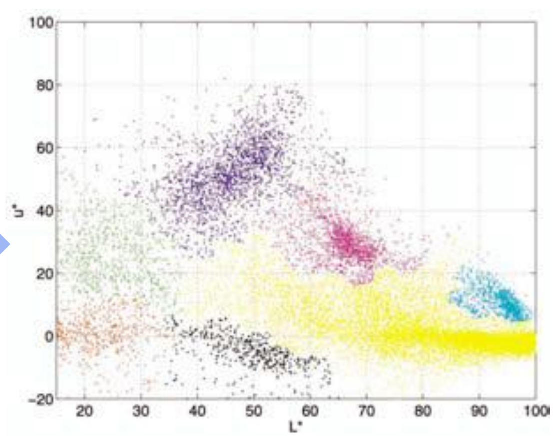
- Formulate clustering as *histogram partitioning*
  - look for **modes** in data histograms
  - assign points to modes



data points



data histogram and its modes

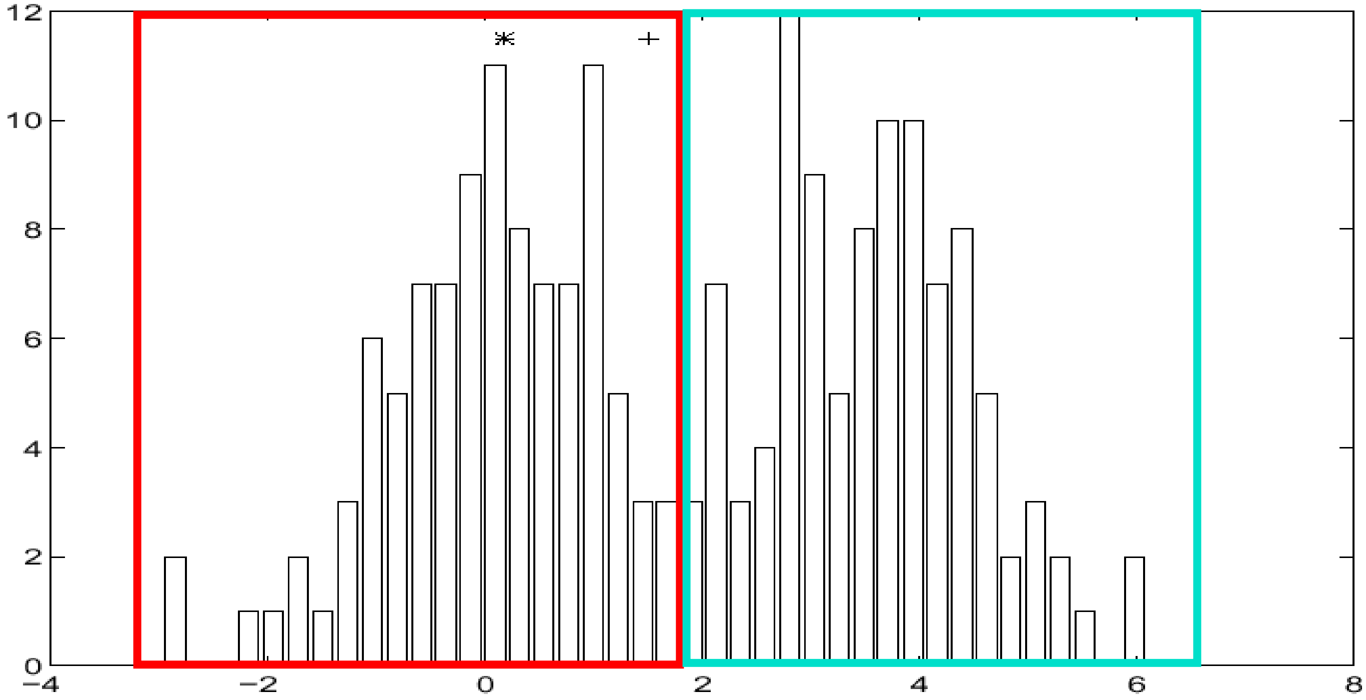


clustering

UNIVERSITY OF WATERLOO

Optional Material

Finding Modes in a Histogram



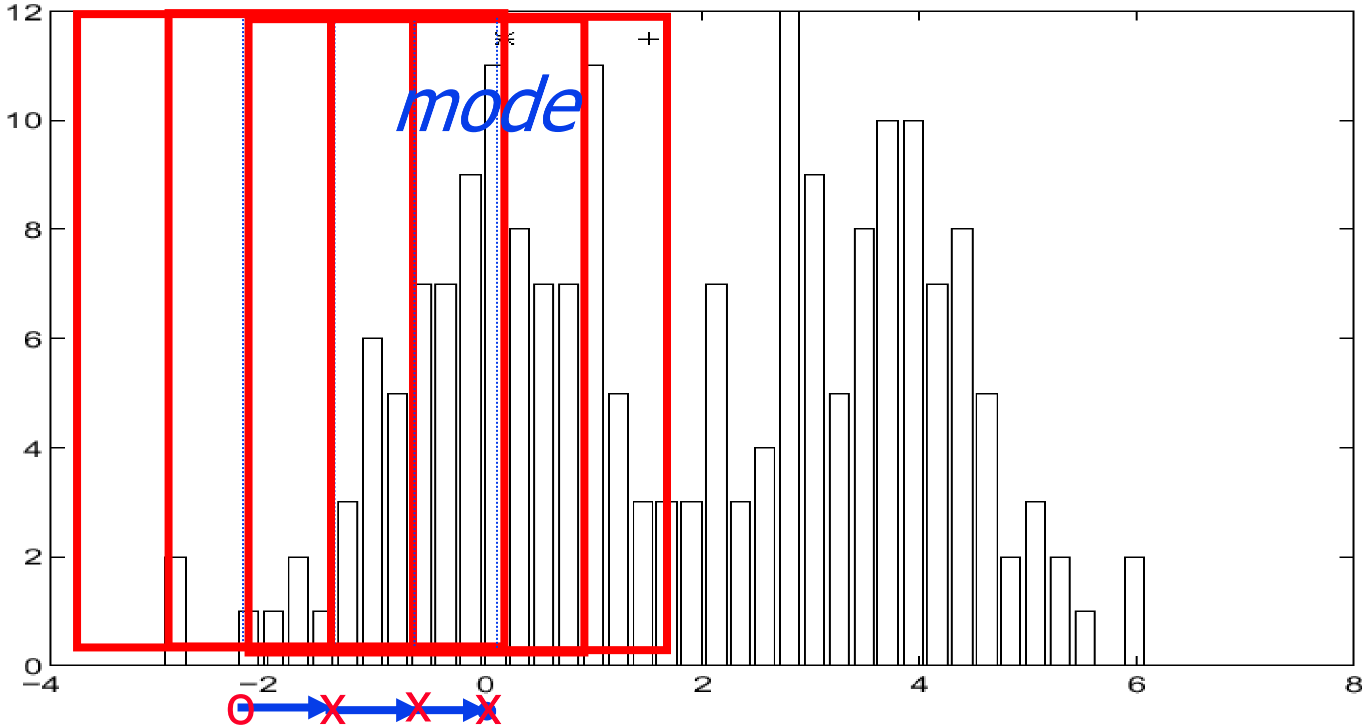
- How Many Modes Are There?
  - Easy to see, not too obvious how to compute

UNIVERSITY OF WATERLOO

Optional Material

Mean Shift

[Fukunaga and Hostetler 1975, Cheng 1995, Comaniciu & Meer 2002]



Iterative Mode Search

1. Initialize random seed, and fixed window
2. Calculate center of gravity ‘x’ of the window (the“mean”)
3. Translate the search window to the mean
4. Repeat Step 2 until convergence

UNIVERSITY OF WATERLOO

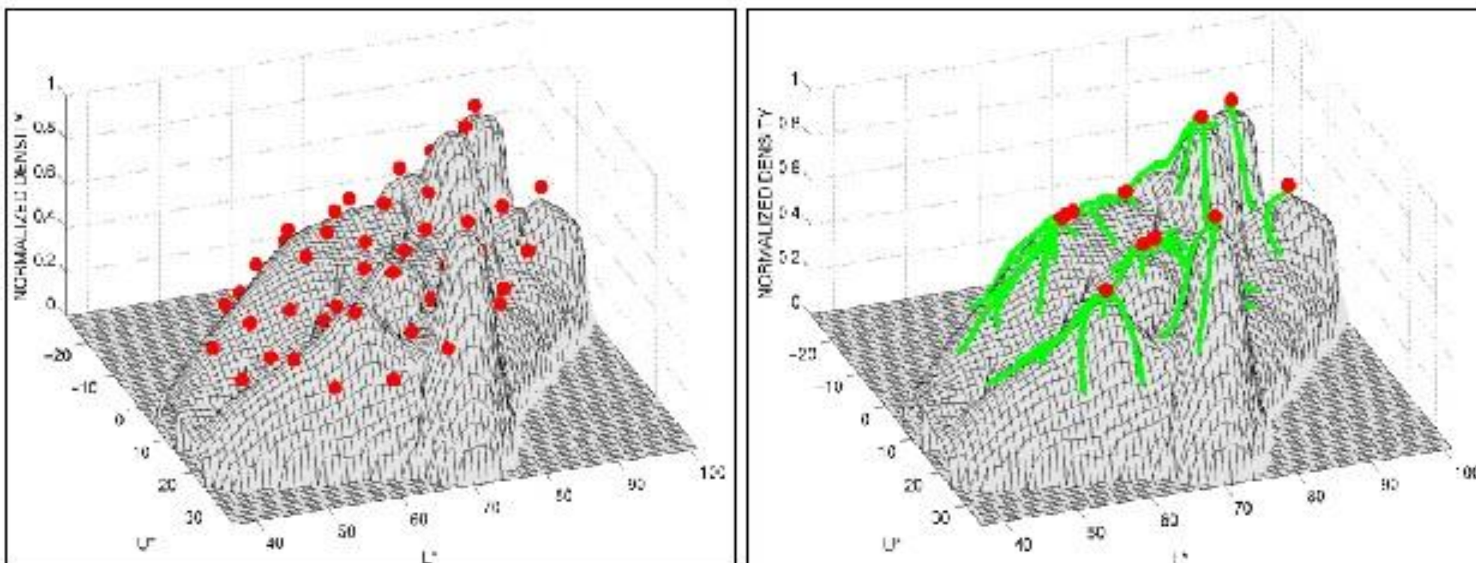
Optional Material

Mean Shift

[Fukunaga and Hostetler 1975, Cheng 1995, Comaniciu & Meer 2002]

Multimodal Distributions

- Parallel processing of an initial tessellation.
- Pruning of mode candidates.
- Classification based on the basin of attraction.



Mean shift trajectories

UNIVERSITY OF WATERLOO

Optional Material

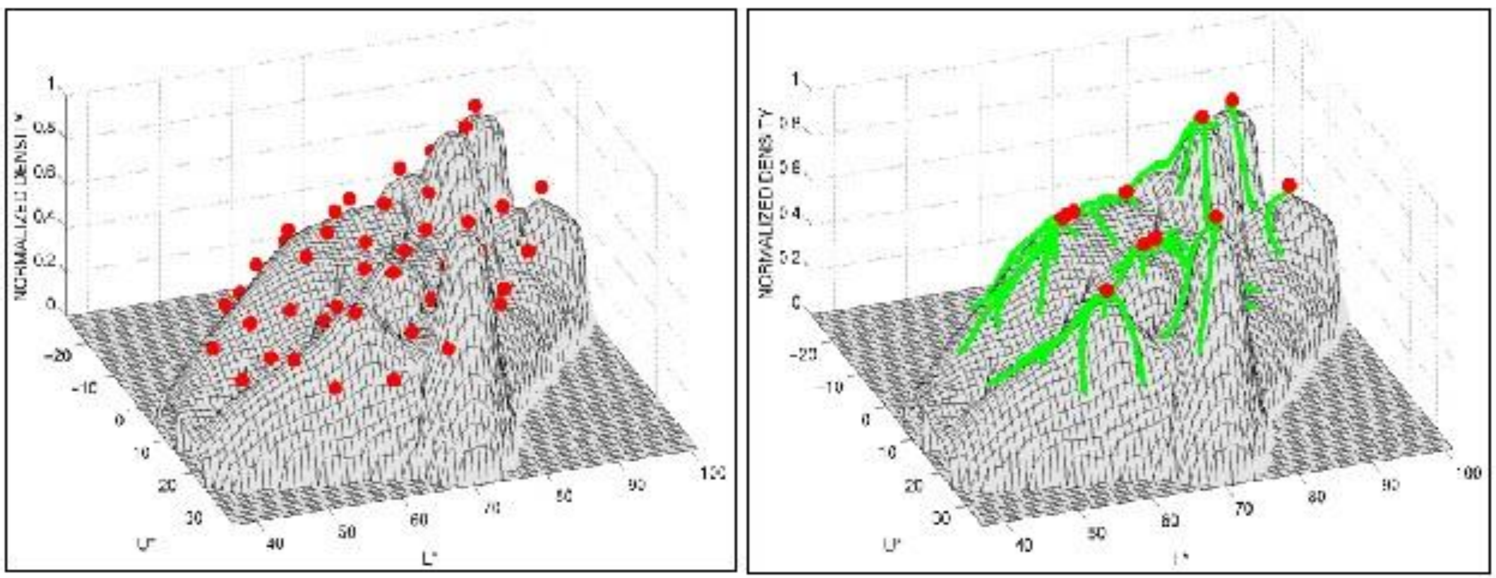
Mean Shift as K-modes

[Salah, Mitche, Ben-Ayed 2010]

$$\sum_{k=1}^{K}\sum_{p\in S^k}\left\| f_p - \mu_k \right\|_d \qquad \|\cdot\|_d \; :$$

quadratic	absolute	bounded
(K-means)	(K-medians)	(K-modes)

Mean-shift segmentation relates to distortion clustering with a bounded loss (**K-modes**)



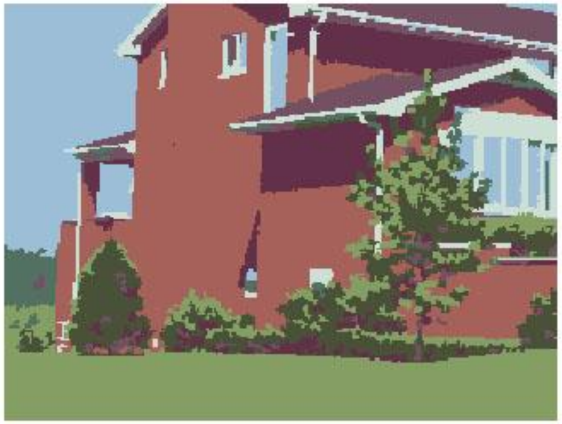
Mean shift trajectories

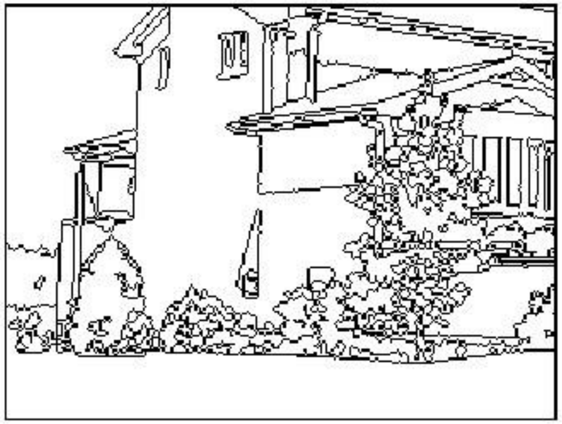
Mean-shift results for segmentation

Optional Material

RGB+XY clustering
[Comaniciu & Meer 2002]

Figure 2: The *house* image, 255 × 192 pixels, 9603 colors.

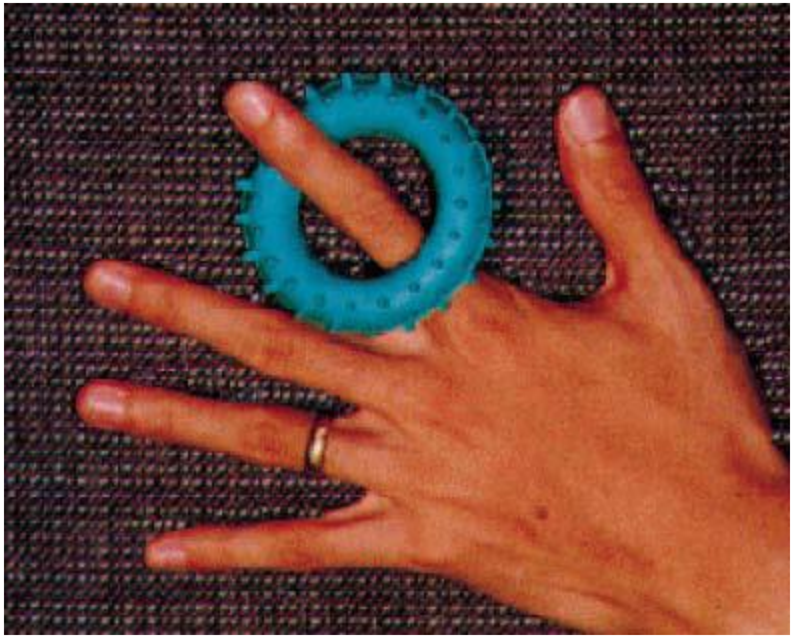


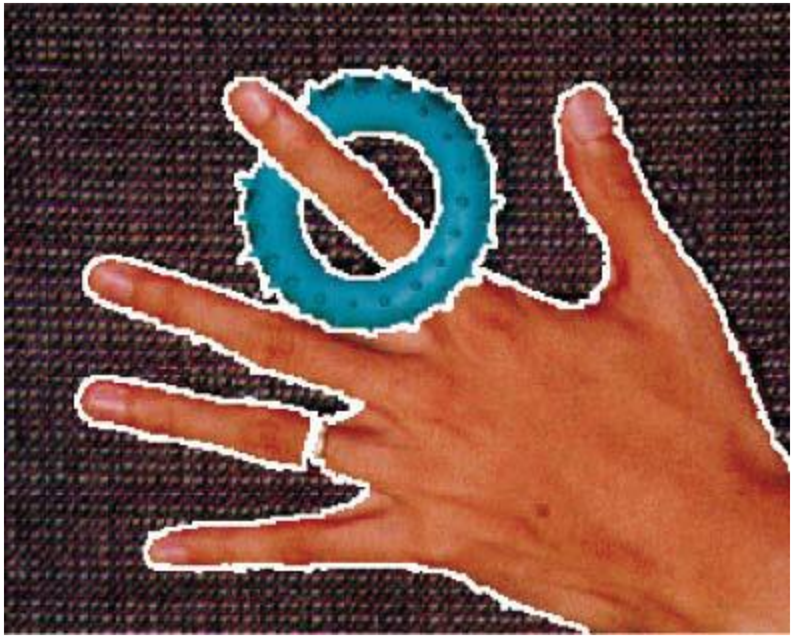


Mean-shift results for segmentation

Optional Material

RGB+XY clustering
[Comaniciu & Meer 2002]

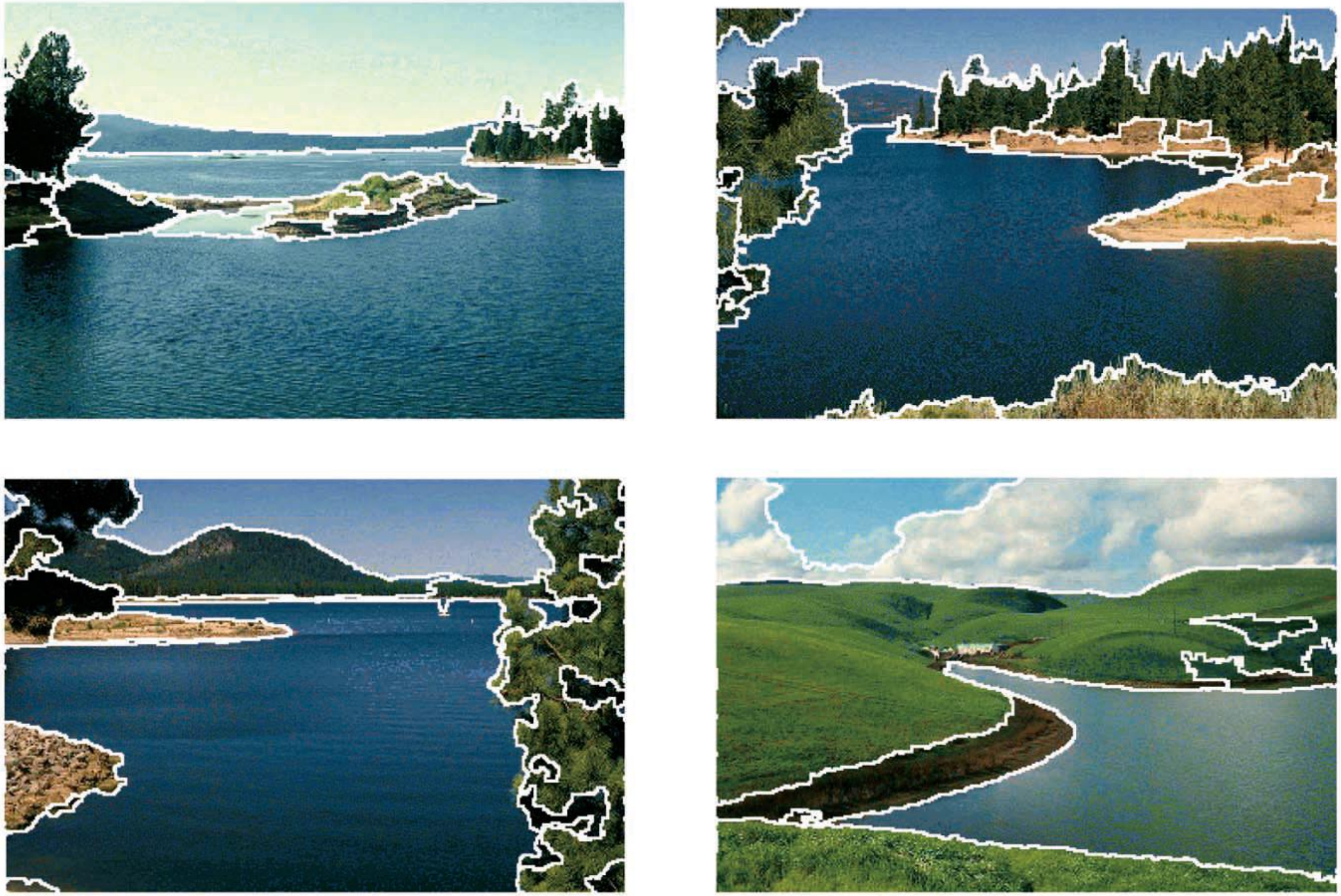




Mean-shift results for segmentation

Optional Material

RGB+XY clustering
[Comaniciu & Meer 2002]



works well for segments with near-consistent color

Issues for kernel clustering methods:

- *kernel bandwidth* selection
  - can not be too small or too large
  - indirectly controls the number of clusters (in *mean-shift*)
  - different width in RGB and XY parts of the space
- Biases (e.g. to equal size, to dense clumps, to sparse points, etc)
  - can use adaptive bandwidths or weighted points, e.g. [Marin *et al*. TPAMI 2017]
- Color features may not be discriminant enough

  (e.g. color overlap between different objects)
- Boundary properties (geometry) are missing
  - contrast edge alignment could be a problem
  - smoothness or other shape priors