DIGITAL IMAGE PROCESSING

Lecture 12

Image Segmentation/ Unsupervised Learning

Tammy Riklin Raviv
Electrical and Computer Engineering
Ben-Gurion University of the Negev

Machine Learning Problems

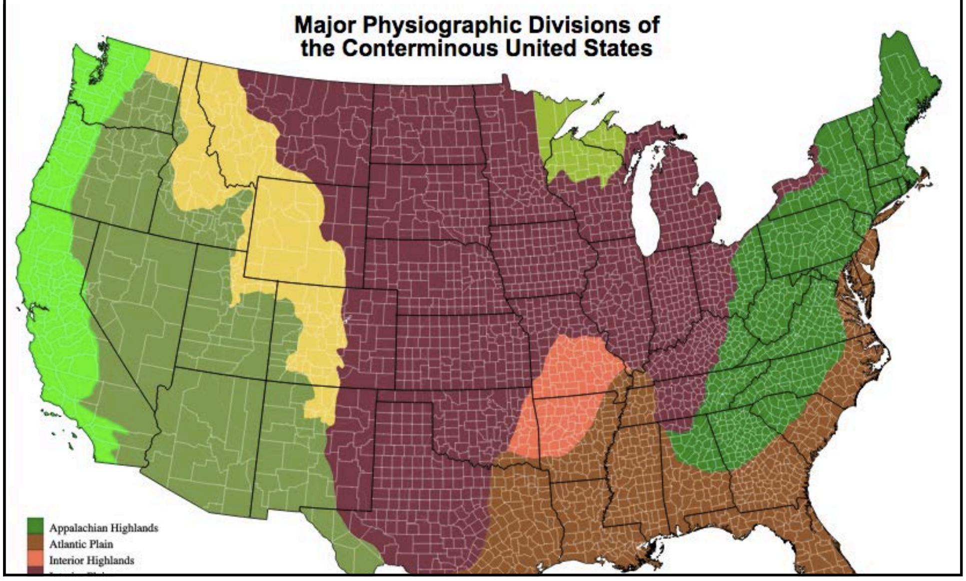
	Supervised Learning	*Unsupervised Learning*
Discrete	classification or categorization	clustering
Continuous	regression	dimensionality reduction

Machine Learning Problems

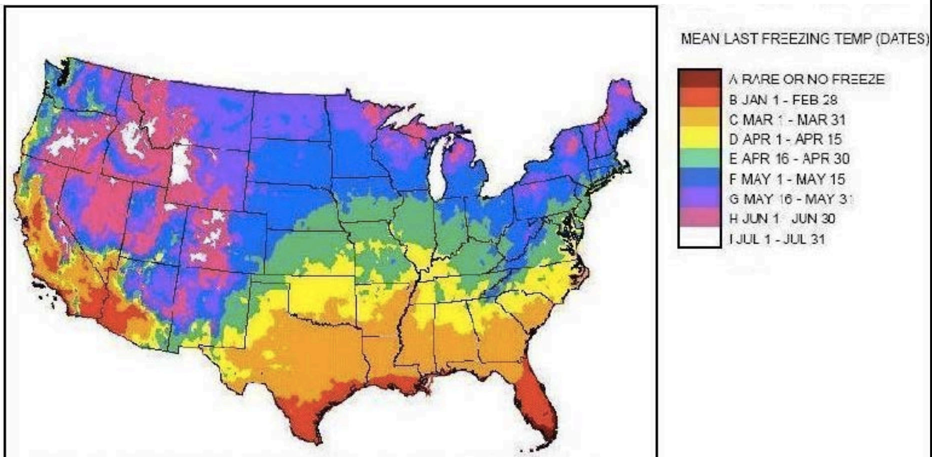
	Supervised Learning	Unsupervised Learning
Discrete	classification or categorization	clustering
Continuous	regression	dimensionality reduction

Cluster The United States of America

Cluster The United States of America



Cluster The United States of America



Mean date of last 32 degree temperature for Spring planting - Internet Accuracy Project

Cluster The United States of America

http://fakeisthenewreal.org/reform/

Clustering

Group together similar 'points' and represent them with a single token.

Key Challenges:

1) Which features to select for meaningful clustering?

2) What makes two points/images/patches similar? - define a metric

3) How do we compute an overall grouping from pairwise similarities?

4) Hard or Soft Clustering?

Adapted from Derek Hoiem

Why do we cluster?

- **Summarizing data**
  - Look at large amounts of data
  - Patch-based compression or denoising
  - Represent a large continuous vector with the cluster number
- **Counting**
  - Histograms of texture, color, SIFT vectors
- **Segmentation**
  - Separate the image into different regions
- **Prediction**
  - Images in the same cluster may have the same labels

Derek Hoiem

How do we cluster?

- K-means
  - Iteratively re-assign points to the nearest cluster center
- Agglomerative clustering
  - Start with each point as its own cluster and iteratively merge the closest clusters
- Mean-shift clustering
  - Estimate modes of probability density function (pdf)
- Spectral clustering
  - Split the nodes in a graph based on assigned links with similarity weights

K-means algorithm

1. Randomly select K centers
2. Assign each point to nearest center
3. Compute new center (mean) for each cluster

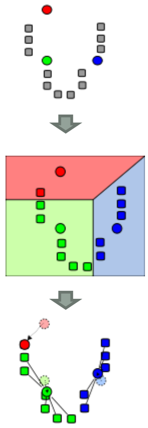
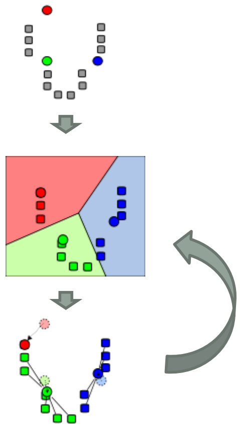


Illustration: http://en.wikipedia.org/wiki/K-means_clustering

K-means algorithm

1. Randomly select K centers
2. Assign each point to nearest center
3. Compute new center (mean) for each cluster



Back to 2

Illustration: http://en.wikipedia.org/wiki/K-means_clustering

K-means: design choices

- Initialization
  - Randomly select K points as initial cluster center
  - Or greedily choose K points to minimize residual
- Distance measures
  - Traditionally Euclidean, could be others
- Optimization
  - Will converge to a *local minimum*
  - May want to perform multiple restarts

K-means

1. Initialize cluster centers: $\mathbf{c}^0$; t=0

2. Assign each point to the closest center

$$\boldsymbol{\delta}^t = \underset{\boldsymbol{\delta}}{\operatorname{argmin}} \tfrac{1}{N} \sum_j^N \sum_i^K \delta_{ij} \left(\mathbf{c}_i^{t-1} - \mathbf{x}_j\right)^2$$

3. Update cluster centers as the mean of the points

$$\mathbf{c}^t = \underset{\mathbf{c}}{\operatorname{argmin}} \tfrac{1}{N} \sum_j^N \sum_i^K \delta_{ij}^t \left(\mathbf{c}_i - \mathbf{x}_j\right)^2$$

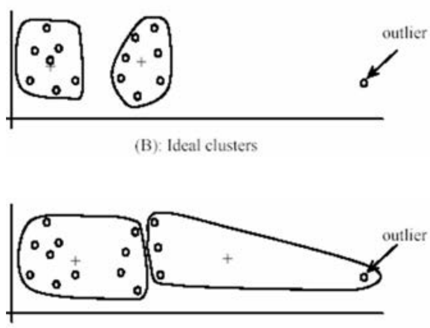
4. Repeat 2-3 until no points are re-assigned (t=t+1)

Slide: Derek Hoiem

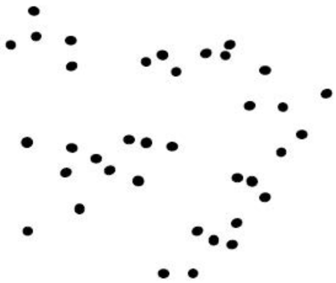
K-means Clustering Example

K-Means pros and cons

- Pros
  - Finds cluster centers that minimize conditional variance (good representation of data)
  - Simple and fast*
  - Easy to implement
- Cons
  - Need to choose K
  - Sensitive to outliers
  - Prone to local minima
  - All clusters have the same parameters (e.g., distance measure is non-adaptive)
  - *Can be slow: each iteration is O(KNd) for N d-dimensional points
- Usage
  - Rarely used for pixel segmentation

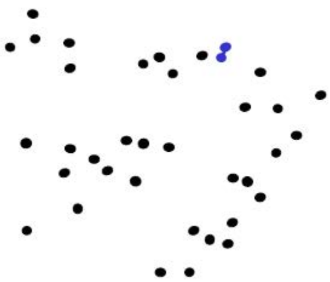


Agglomerative clustering



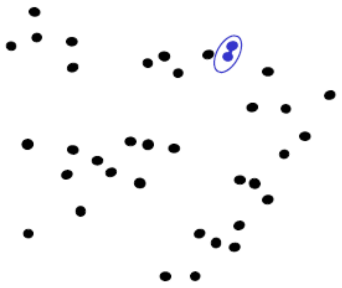
1. Say "Every point is its own cluster"

Agglomerative clustering



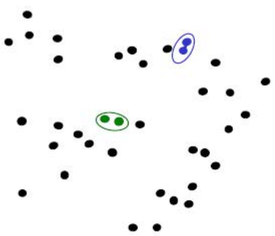
1. Say "Every point is its own cluster"
2. Find "most similar" pair of clusters

Agglomerative clustering



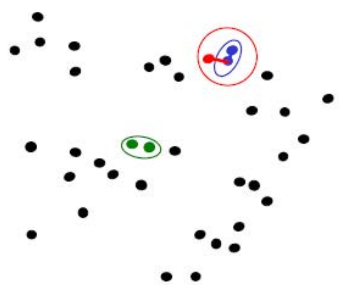
1. Say "Every point is its own cluster"
2. Find "most similar" pair of clusters
3. Merge it into a parent cluster

Agglomerative clustering

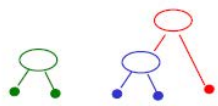


1. Say "Every point is its own cluster"
2. Find "most similar" pair of clusters
3. Merge it into a parent cluster
4. Repeat

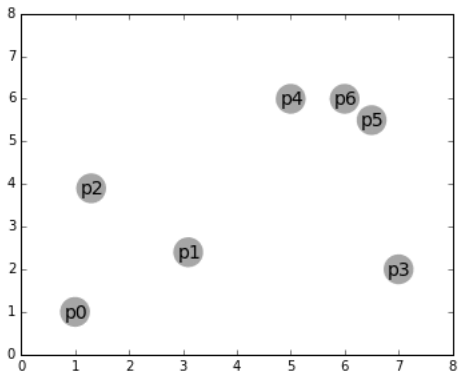
Agglomerative clustering

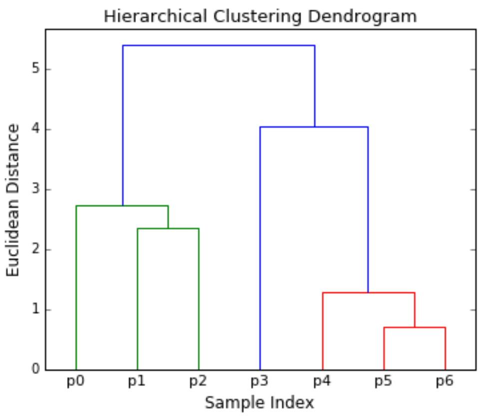


1. Say "Every point is its own cluster"
2. Find "most similar" pair of clusters
3. Merge it into a parent cluster
4. Repeat

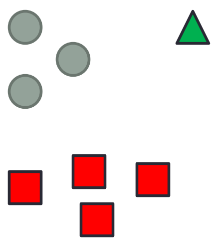


Hierarchical Clustering Example





Agglomerative clustering

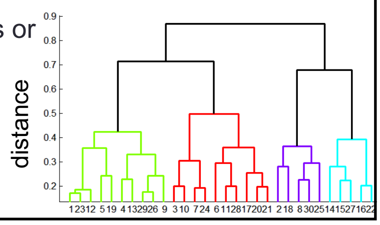


How to define cluster similarity?

- Average distance between points, maximum distance, minimum distance
- Distance between means or medoids
- (medoids like means but restricted to be in the dataset)

How many clusters?

- Clustering creates a dendrogram (a tree)
- Threshold based on max number of clusters or based on distance between merges



Conclusions: Agglomerative Clustering

Good

- Simple to implement, widespread application
- Clusters have adaptive shapes
- Provides a hierarchy of clusters

Bad

- May have imbalanced clusters
- Still have to choose number of clusters or threshold
- Need to use an “ultrametric” to get a meaningful hierarchy

Let's return to K-means…

Expectation-Maximization Algorithm

K-Means – the Soft Version

K-means algorithm is a hard clustering algorithm: every point is assigned to a single cluster.

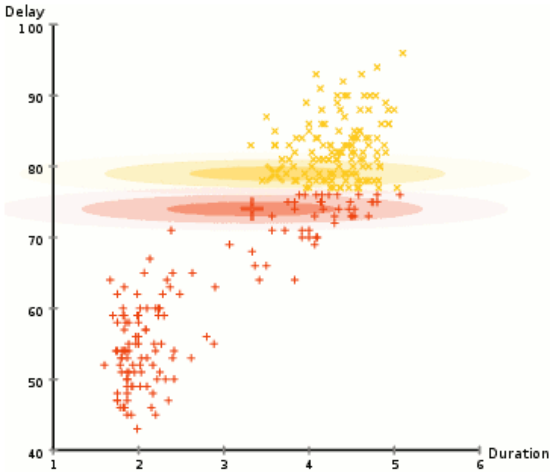
It is an iterative algorithm with two step: assign and update.

In soft clustering algorithm all data points are assigned to all clusters with a certain degree (or weight).

The **EM** algorithm is a soft clustering algorithm (analogous to K-means) where **E** stands for **Expectation** and **M** for **Maximization**.

(Dempster, Laird, and Rubin 1977)

Expectation-Maximization Algorithm

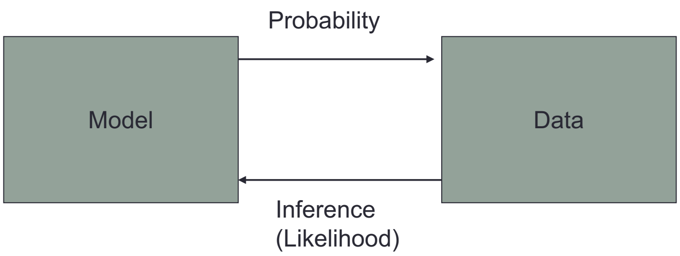


https://lizluvsanime2.deviantart.com/art/Old-Faithful-129421239

Old Faithful data

From Wikipedia

Some Background before we go deeper



A model of the data generating process gives rise to data.
Model estimation from data is most commonly done through Likelihood estimation

sydney.edu.au/engineering/it/~comp5318/lectures/EMAlgorithm.ppt

Likelihood Function

$$P(Model \mid Data) = \frac{P(Data \mid Model)P(Model)}{P(Data)}$$

Likelihood Function

Find the “best” model which has generated the data. In a likelihood function the data is considered fixed and one searches for the best model over the different choices available.

sydney.edu.au/engineering/it/~comp5318/lectures/EMAlgorithm.ppt

Model Space

- The choice of the model space is plentiful but not unlimited.
- There is a bit of “art” in selecting the appropriate model space.
- Typically the model space is assumed to be a linear combination of known probability distribution functions.

sydney.edu.au/engineering/it/~comp5318/lectures/EMAlgorithm.ppt

Examples

- Suppose we have the following data
  - 0,1,1,0,0,1,1,0
- In this case it is sensible to choose the Bernoulli distribution (B(p)) as the model space.

$$P(X = x) = p^x(1-p)^{1-x}$$

- Now we want to choose the best p, i.e.,

$$\text{argmax}_p P(Data|B(p))$$

sydney.edu.au/engineering/it/~comp5318/lectures/EMAlgorithm.ppt

Examples

Suppose the following are marks in a course

55.5, 67, 87, 48, 63

Marks typically follow a Normal distribution whose density function is

$$N(\mu, \sigma) = \frac{1}{\sqrt{2\pi\sigma}} e^{-\frac{1}{2\sigma}(x-\mu)^2}$$

Now, we want to find the best μ,σ such that

$$argmax_{\mu,\sigma} p(Data|\mu, \sigma)$$

sydney.edu.au/engineering/it/~comp5318/lectures/EMAlgorithm.ppt

Examples

- Suppose we have data about heights of people (in cm)
  - 185,140,134,150,170
- Heights follow a normal (log normal) distribution but men on average are taller than women. This suggests a **mixture** of two distributions

$$\pi_1 N(\mu_1, \sigma_1) + \pi_2 N(\mu_2, \sigma_2)$$

sydney.edu.au/engineering/it/~comp5318/lectures/EMAlgorithm.ppt

Maximum Likelihood Estimation (MLE)

- We have reduced the problem of selecting the best model to that of selecting the best parameter.
- We want to select a parameter p which will **maximize** the probability that the data was generated from the model with the parameter p plugged-in.
- The parameter **p** is called the maximum likelihood estimator.

sydney.edu.au/engineering/it/~comp5318/lectures/EMAlgorithm.ppt

MLE for Mixture Distributions

- When we proceed to calculate the MLE for a mixture, the presence of the sum of the distributions prevents a "neat" factorization using the log function.
- A completely new rethink is required to estimate the parameter.
- The new rethink also provides a solution to the clustering problem.

sydney.edu.au/engineering/it/~comp5318/lectures/EMAlgorithm.ppt

Expectation-Maximization Algorithm

An **expectation–maximization** (**EM**) **algorithm** is an iterative method to find maximum likelihood or maximum a posteriori (MAP) estimates of parameters in statistical models, where the model depends on unobserved latent variables. The EM iteration alternates between

1. Expectation (E) step: expectation of the log-likelihood evaluated using the current estimate for the parameters
2. Maximization (M) step: which computes parameters maximizing the expected log-likelihood found on the *E* step.

These parameter-estimates are then used to determine the distribution of the latent variables in the next E step.

Wikipedia: EM

EM Algorithm for Mixture of Normals

$$f(x) = \sum_{k=1}^{K} \pi_k f_k(x, \mu_k, \sigma_k)$$

Mixture of Normals

$$P(k|x) = \frac{\pi_k f_k(x, \mu_k, \sigma_k)}{f(x)}$$

E Step

$$\pi_k = \frac{1}{n} \sum_{i=1}^{n} P(k|x(i))$$

$$\mu_k = \frac{1}{n\pi_k} \sum_{i=1}^{n} P(k|x(i))x(i)$$

$$\sigma_k = \frac{1}{n\pi_k} \sum_{i=1}^{n} P(k|x(i))(x(i) - \mu_k)^2$$

M-Step

sydney.edu.au/engineering/it/~comp5318/lectures/EMAlgorithm.ppt

EM and K-means

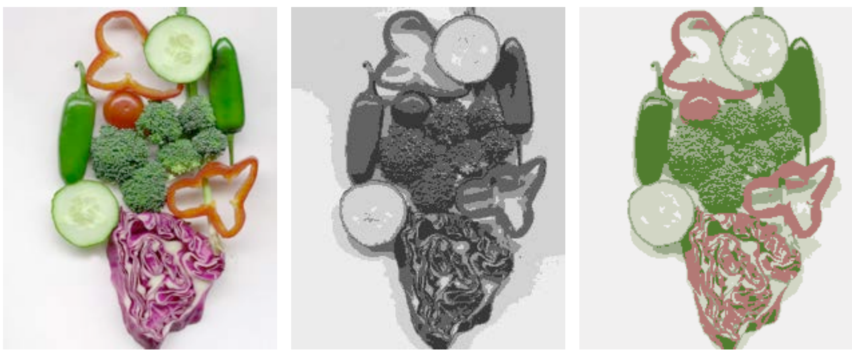
- Notice the similarity between EM for Normal mixtures and K-means.
- The expectation step is the assignment.
- The maximization step is the update of centers.

Clustering for Image Processing: Image Segmentation

Goal: Break up the image into meaningful or perceptually similar regions

K-means clustering using intensity or color

Image | Clusters on intensity | Clusters on color



Segmentation by K-Means Clustering

Matlab Command:

idx = kmeans(X,k)

Input: X – n-by-p observation matrix

for Images: n is the number of pixels,

p is the number of features:

RGB – channels; or RGB+ image coordinates (x,y)

Output: vector idx containing cluster indices

Features Space

$$\begin{vmatrix} 49 & 37 & 47 \\ 51 & 41 & 50 \\ 49 & 44 & 51 \\ 46 & 45 & 53 \\ 41 & 45 & 54 \\ 38 & 47 & 56 \\ & \vdots & \end{vmatrix}$$

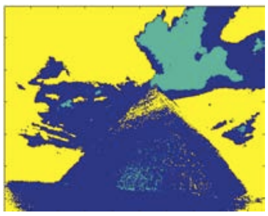
Segmentation by K-Means Clustering

$$X = \begin{vmatrix} 49 & 37 & 47 \\ 51 & 41 & 50 \\ 49 & 44 & 51 \\ 46 & 45 & 53 \\ 41 & 45 & 54 \\ 38 & 47 & 56 \\ & \vdots & \end{vmatrix}$$

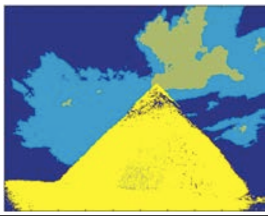
What is K?

3 ? 4? ….

K=3

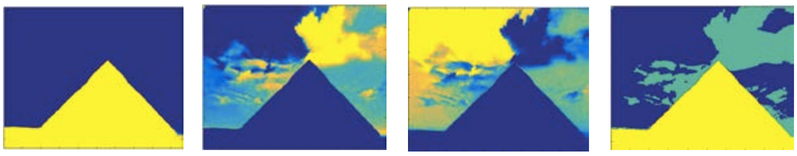


K=4

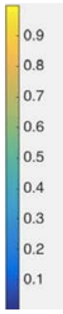


GMM- EM-based Segmentation

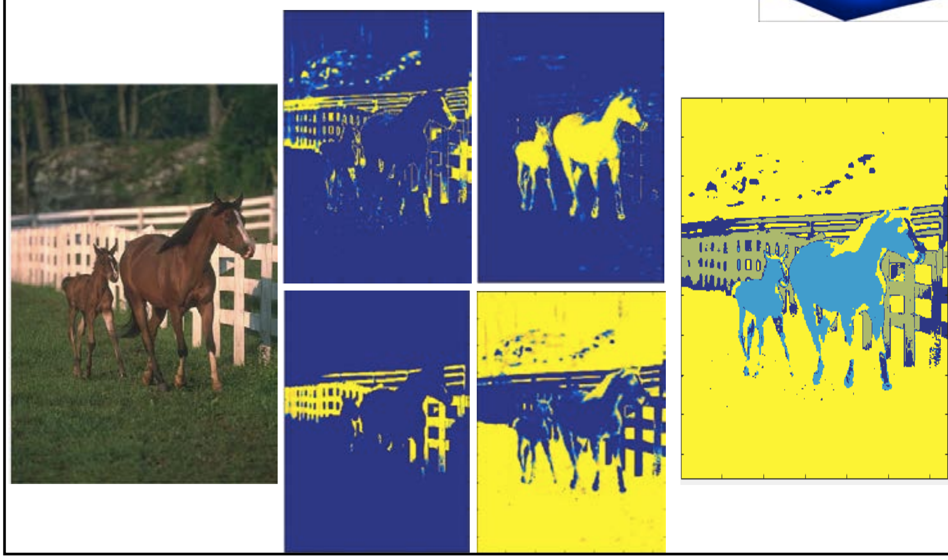
$$X = \begin{vmatrix} 49 & 37 & 47 \\ 51 & 41 & 50 \\ 49 & 44 & 51 \\ 46 & 45 & 53 \\ 41 & 45 & 54 \\ 38 & 47 & 56 \\ & \vdots & \end{vmatrix}$$



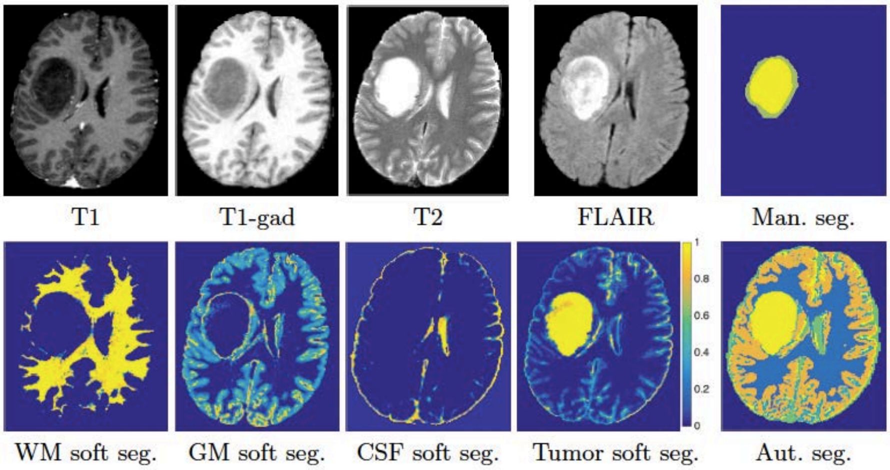
$P(\mathbf{x} \in \omega_1)$ $P(\mathbf{x} \in \omega_2)$ $P(\mathbf{x} \in \omega_3)$ Label Map



EM-Based Segmentation



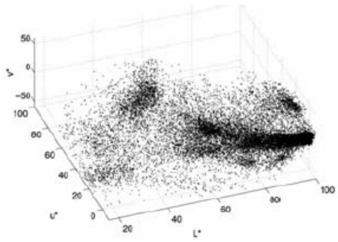
Brain Tumor Segmentation



Tammy Riklin Raviv, Multinomial Level-Set Framework for Multi-Region Image Segmentation, SSVM 2017

Mean shift algorithm

Try to find *modes* of a non-parametric density.



Original Image

L*U*V* color space

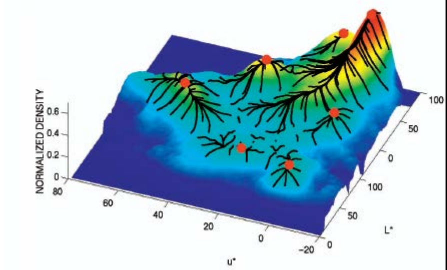
Find smooth continuous non-parametric model of the intensity distribution

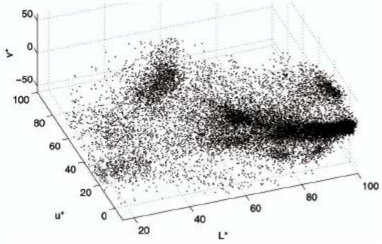
Efficiently search for peaks in this high-dimensional data distribution without ever computing the complete function explicitly

(Fukunaga and Hostetler 1975; Cheng 1995; Comaniciu and Meer 2002).

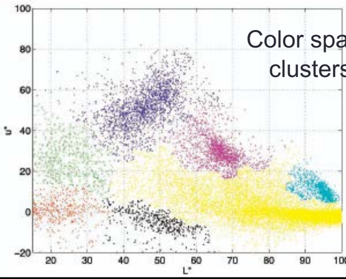
Mean shift algorithm

Try to find *modes* of a non-parametric density.





Color space

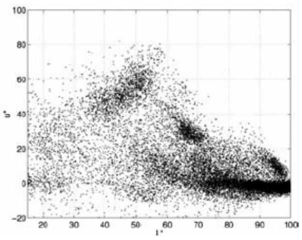


Mean Shift Algorithm

How to estimate the density function given a sparse set of samples?

smooth the data, e.g., by convolving it with a fixed kernel of width h:

$$f(x) = \sum_i K(x - x_i) = \sum_i k\left(\frac{\|x - x_i\|^2}{h^2}\right)$$



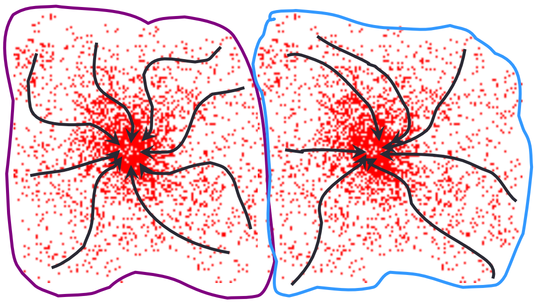
L*U* color space

where x_i are the input samples and $k(r)$ is the kernel function (or *Parzen window*).

Once we have computed f(x), as we can find its local maxima using gradient ascent or some other optimization technique.

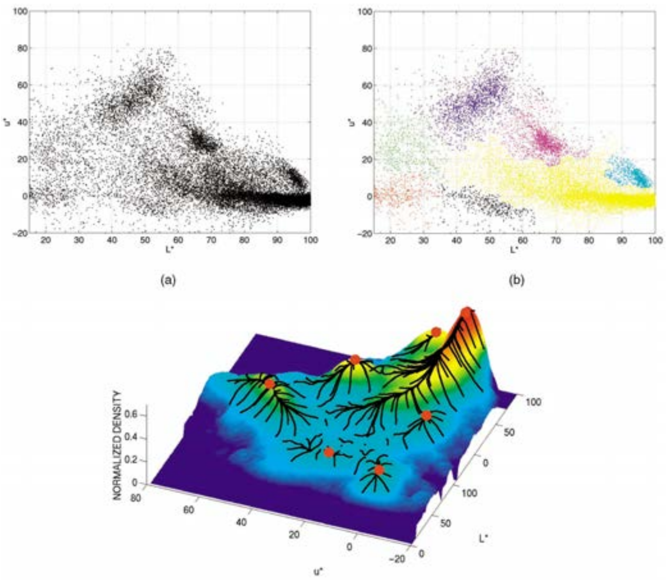
Attraction basin

- Attraction basin: the region for which all trajectories lead to the same mode
- Cluster: all data points in the attraction basin of a mode

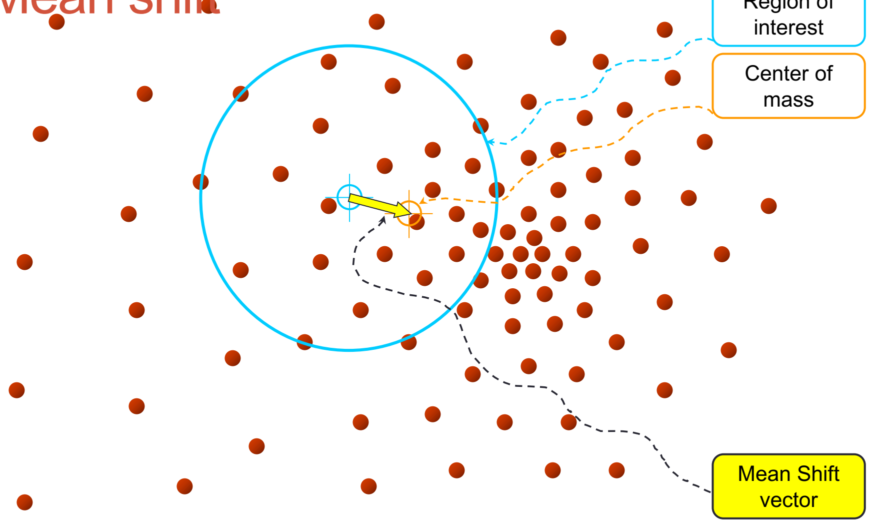


Slide by Y. Ukrainitz & B. Sarel

Attraction basin

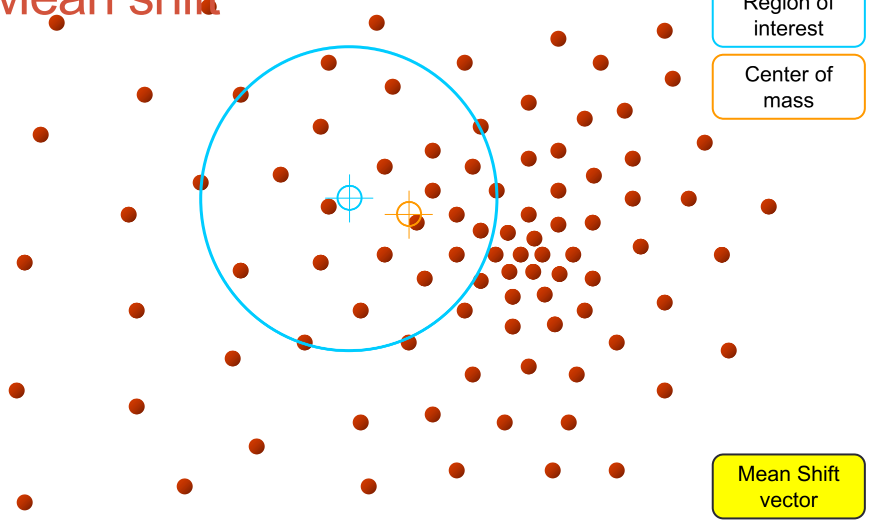


Mean shift



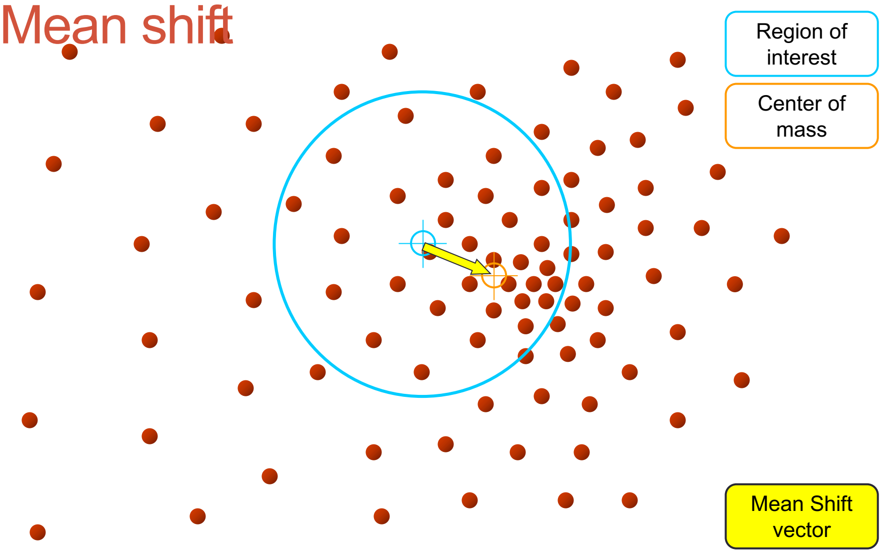
Slide by Y. Ukrainitz & B. Sarel

Mean shift



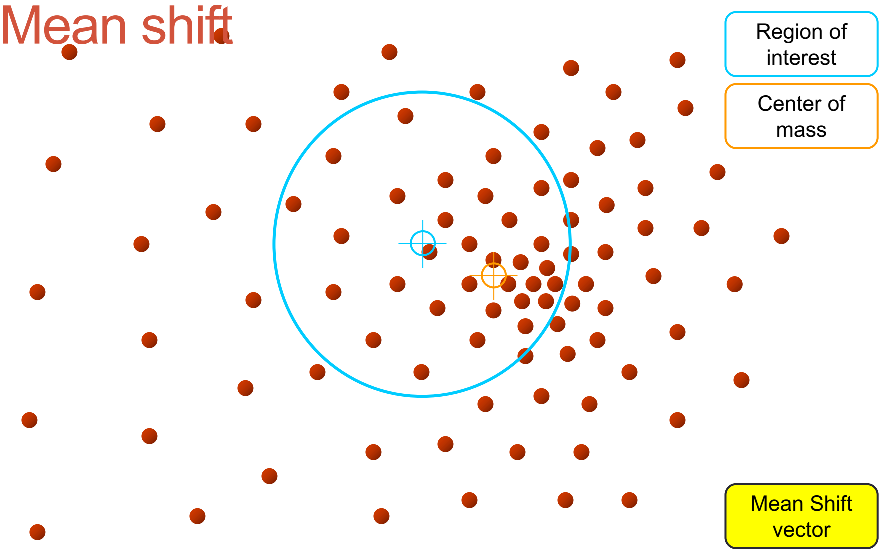
Slide by Y. Ukrainitz & B. Sarel

Mean shift

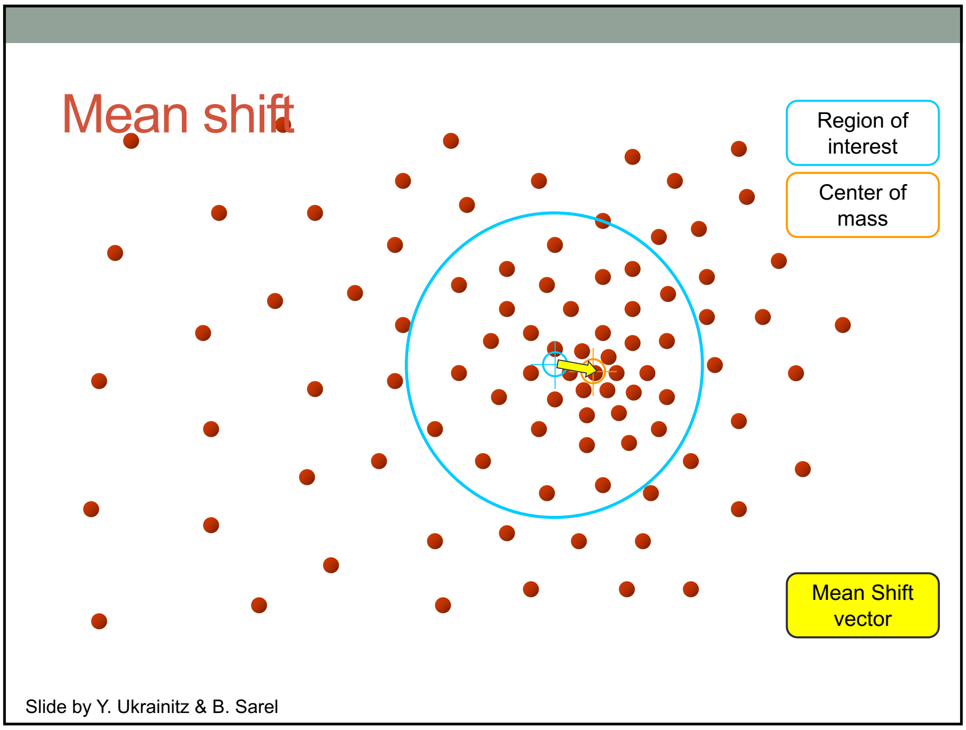


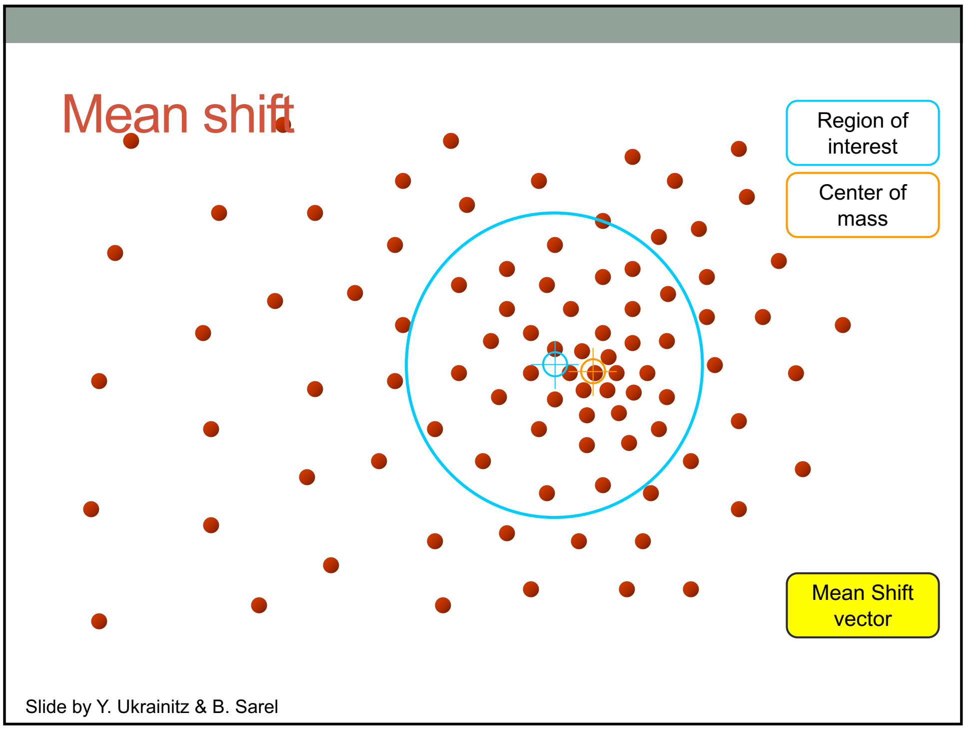
Slide by Y. Ukrainitz & B. Sarel

Mean shift

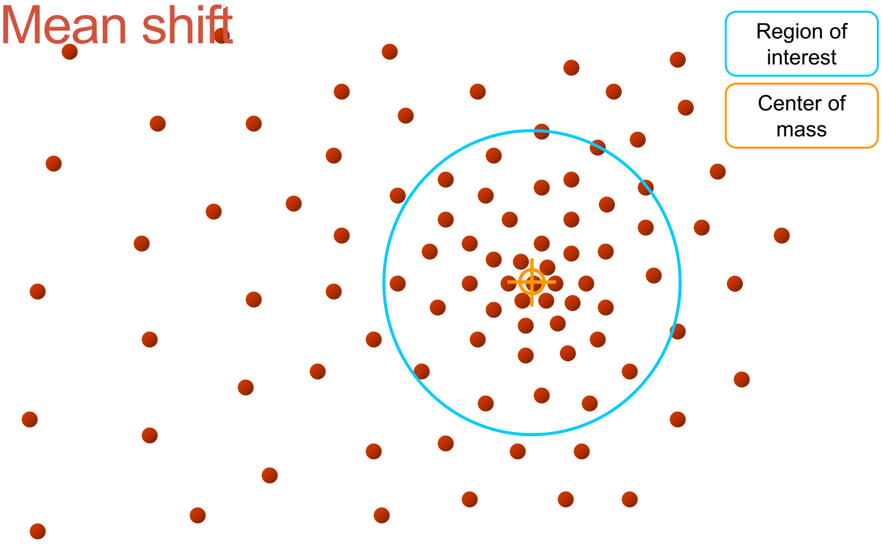


Slide by Y. Ukrainitz & B. Sarel


Mean shift
Region of interest
Center of mass
Mean Shift vector
Slide by Y. Ukrainitz & B. Sarel


Mean shift
Region of interest
Center of mass
Mean Shift vector
Slide by Y. Ukrainitz & B. Sarel

Mean shift



Slide by Y. Ukrainitz & B. Sarel

Mean-shift algorithm

- Mean shift is a procedure for locating the maxima—the modes—of a density function given discrete data sampled from that function.
- Let a kernel function $K(\mathbf{x} - \mathbf{x}_i)$ be given.
- Typical kernels :
- Gaussian: $K(\mathbf{x} - \mathbf{x}_i) = k\left(\frac{||\mathbf{x} - \mathbf{x}_i||^2}{h^2}\right)$
- Flat kernel:

$$K(\mathbf{x} - \mathbf{x}_i) = \begin{cases} 1 & \text{if} \quad ||\mathbf{x} - \mathbf{x}_i|| \leq \lambda \\ 0 & \text{if} \quad ||\mathbf{x} - \mathbf{x}_i|| > \lambda \end{cases}$$

Mean-shift algorithm

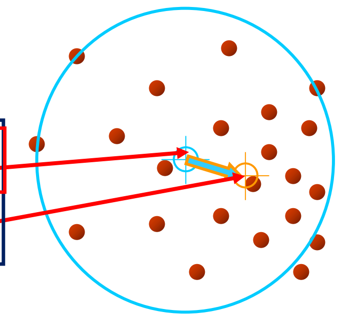
- Mean shift is a procedure for locating the maxima—the modes—of a density function given discrete data sampled from that function.
- Let a kernel function $K(\mathbf{x}-\mathbf{x}_i)$ be given.
- The weighted mean of the density in the window determined by K is
$$m(\mathbf{x}) = \frac{\sum_{\mathbf{x}_i \in N(\mathbf{x})} K(\mathbf{x}_i - \mathbf{x})\mathbf{x}_i}{\sum_{\mathbf{x}_i \in N(\mathbf{x})} K(\mathbf{x}_i - \mathbf{x})}$$
- $N(\mathbf{x})$ is the neighborhood of $\mathbf{x}$. A set of points of which $K(\mathbf{x}, \mathbf{x}_i) \neq 0$.

Computing the Mean Shift

Simple Mean Shift procedure:
- Compute mean shift vector

- Translate the Kernel window by **m(x)**

$$m(\mathbf{x}) = \frac{\sum_{\mathbf{x}_i \in N(\mathbf{x})} K(\mathbf{x}_i - \mathbf{x})\boxed{\mathbf{x}_i}}{\sum_{\mathbf{x}_i \in N(\mathbf{x})} K(\mathbf{x}_i - \mathbf{x})}$$



Slide by Y. Ukrainitz & B. Sarel

$g(\mathbf{x}) = -k'(\mathbf{x})$

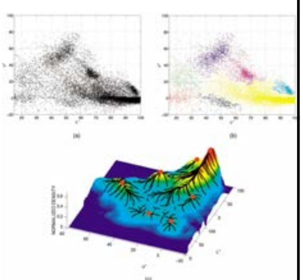
Mean shift clustering

- The mean shift algorithm seeks *modes* of the given set of points
  1. Choose kernel and bandwidth
  2. For each point:
     a) Center a window on that point
     b) Compute the mean of the data in the search window
     c) Center the search window at the new mean location
     d) Repeat (b,c) until convergence
  3. Assign points that lead to nearby modes to the same cluster

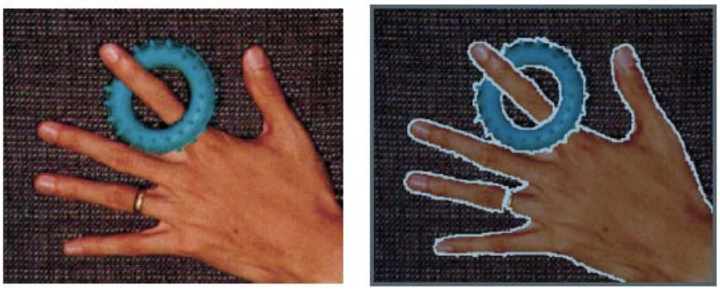
Segmentation by Mean Shift

- Compute features for each pixel (color, gradients, texture, etc.).
- Set kernel size for features K_f and position K_s.
- Initialize windows at individual pixel locations.
- Perform mean shift for each window until convergence.
- Merge windows that are within width of K_f and K_s.

$$K(x_j) = k\left(\frac{\|x_r\|^2}{h_r^2}\right) k\left(\frac{\|x_s\|^2}{h_s^2}\right)$$



Mean Shift Algorithm

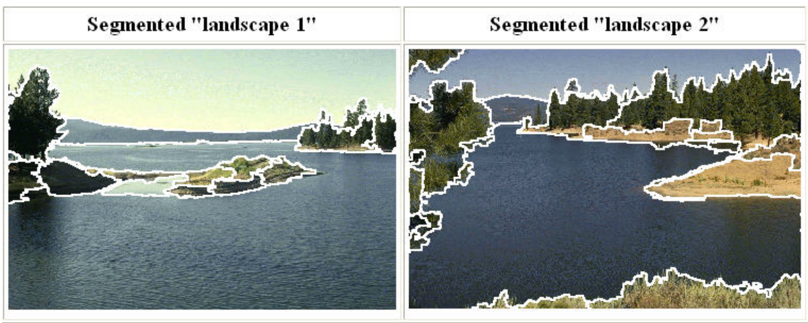


(Comaniciu and Meer 2002) © 2002 IEEE.

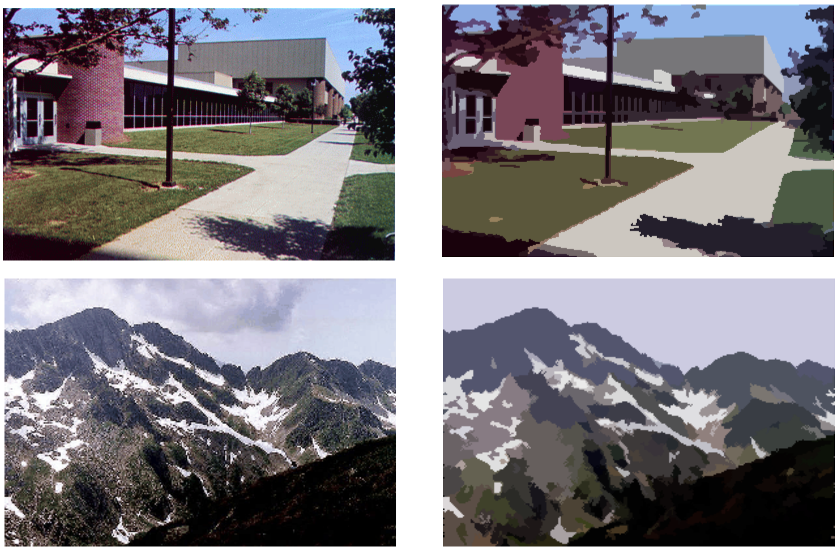
Mean shift segmentation

D. Comaniciu and P. Meer, Mean Shift: A Robust Approach toward Feature Space Analysis, PAMI 2002.

- Versatile technique for clustering-based segmentation



Mean shift segmentation results



Comaniciu and Meer 2002

Mean shift segmentation results

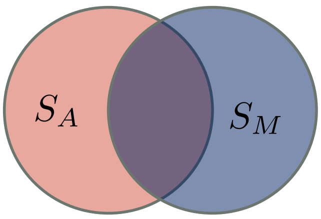
Comaniciu and Meer 2002

Mean shift pros and cons

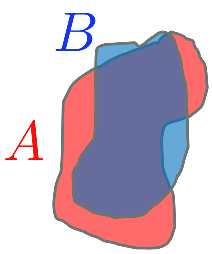
- Pros
  - Good general-practice segmentation
  - Flexible in number and shape of regions
  - Robust to outliers
- Cons
  - Have to choose kernel size in advance
  - Not suitable for high-dimensional features
- When to use it
  - Oversegmentation
  - Multiple segmentations
  - Tracking, clustering, filtering applications

Quantitative Evaluation

$$\mathrm{IoU}(S_A, S_M) = \frac{S_A \cap S_M}{S_A \cup S_M}$$



QUANTITATIVE Similarity Measures

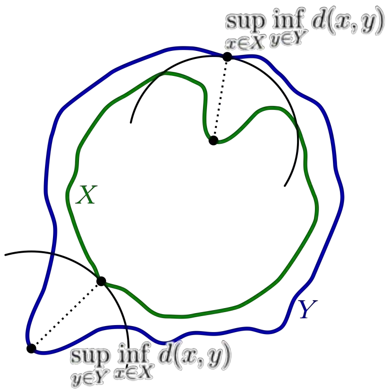


$$DSC = 2\frac{A \cap B}{A + B}$$

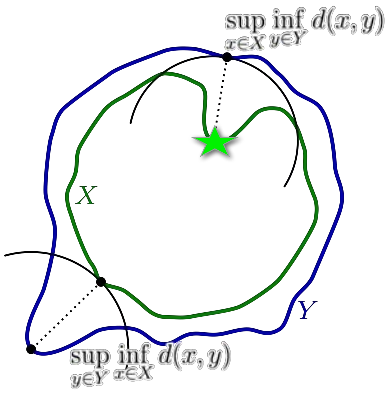
$$IoU = \frac{A \cap B}{A \cup B}$$

$$\mathrm{TS} = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FN} + \mathrm{FP}}$$

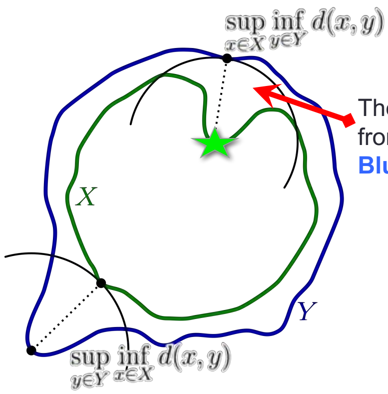
Hausdorff distance



Hausdorff distance

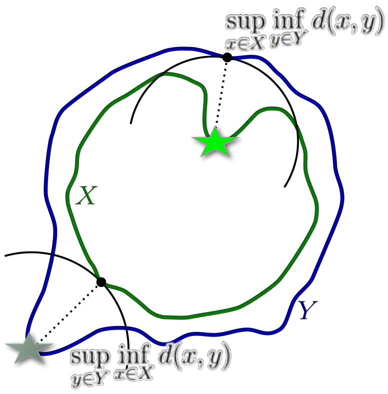


Hausdorff distance

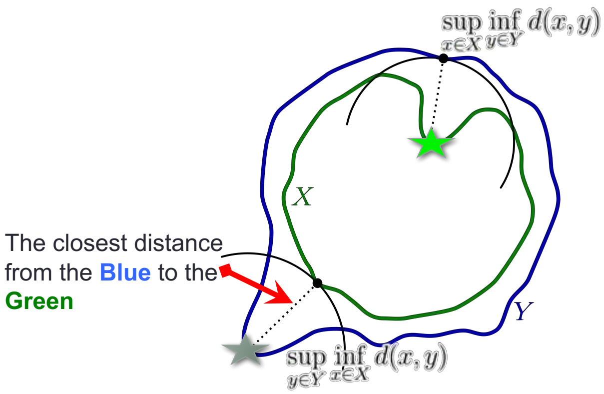


The closest distance from the **Green** to the **Blue.**

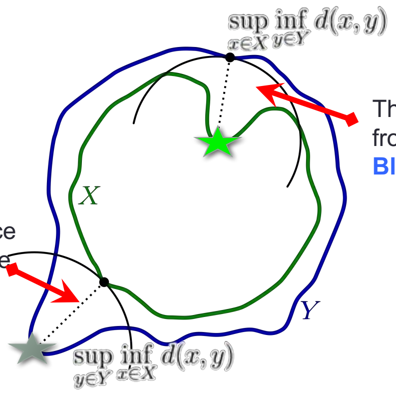
Hausdorff distance



Hausdorff distance



Hausdorff distance



The closest distance from the **Green** to the **Blue.**

The closest distance from the **Blue** to the **Green**

Hausdorff distance

- A shape could be represented by a point cloud S_i
- We could use the **Hausdorff metric** to define the distance between S_i and S_j:

$$D_H(S_i, S_j) =$$

$$\max\{ \sup_{\mathbf{x}_i \in S_i} \inf_{\mathbf{x}_j \in S_j} d(\mathbf{x}_i, \mathbf{x}_j), \sup_{\mathbf{x}_j \in S_j} \inf_{\mathbf{x}_i \in S_i} d(\mathbf{x}_i, \mathbf{x}_j)\},$$

Modified Hausdorff distance

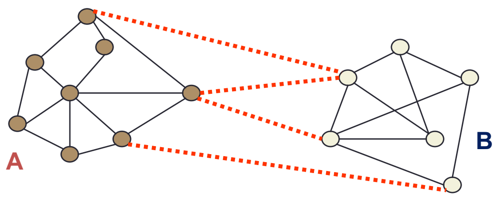
$$D_{MH}(S_i, S_j) =$$

$$\sum_{\mathbf{x}_i \in S_i} \inf_{\mathbf{x}_j \in S_j} d(\mathbf{x}_i, \mathbf{x}_j) + \sum_{\mathbf{x}_j \in S_j} \inf_{\mathbf{x}_i \in S_i} d(\mathbf{x}_i, \mathbf{x}_j).$$

Not a metric yet More robust

Spectral clustering

Group points based on graph structure & edge costs.
Captures "neighborhood-ness" or local smoothness.



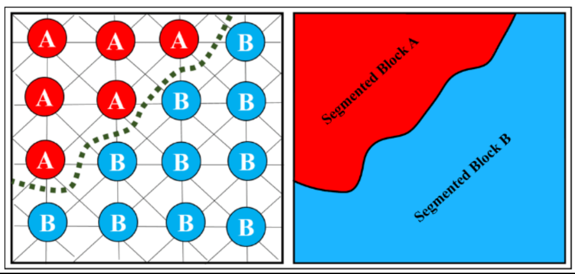


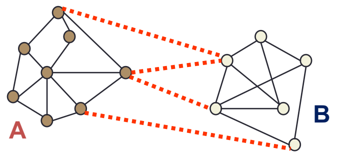
Image:
Hassan et al.

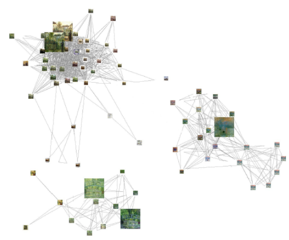
Spectral clustering

Main idea: Group points based on links in a graph

Construct a symmetric matrix W

$w_{i,j}$ is the affinity between points i and j.

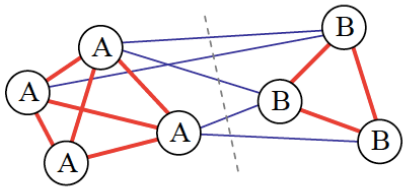




$$\mathrm{cut}(A,B) = \sum_{i \in A, j \in B} w_{i,j}$$

Cuts in a graph

$$\mathrm{cut}(A,B) = \sum_{i \in A, j \in B} w_{i,j}$$



	A	B	sum
A	$assoc(A,A)$	$cut(A,B)$	$assoc(A,V)$
B	$cut(B,A)$	$assoc(B,B)$	$assoc(B,V)$
sum	$assoc(A,V)$	$assoc(B,V)$	

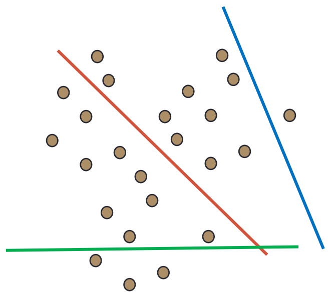
$$\mathrm{assoc}(A,A) = \sum_{i \in A, j \in A} w_{i,j}$$

$$\mathrm{assoc}(B,B) = \sum_{i \in B, j \in B} w_{i,j}$$

$$\mathrm{assoc}(A,V) = \mathrm{assoc}(A,A) + \mathrm{cut}(A,B)$$

sum of all weights associated with A

Normalized Cut (Shi and Malik)



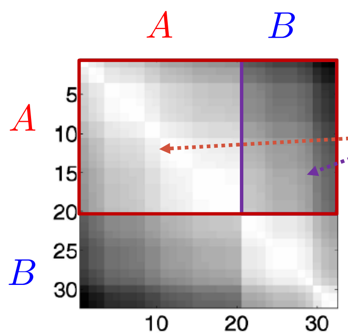
$$\min_{A,B} \text{cut}(A,B) = \sum_{i \in A, j \in B} w_{i,j}$$

imbalance clustering

Normalized cut

$$\text{Ncut}(A,B) = \frac{\text{cut}(A,B)}{\text{assoc}(A,V)} + \frac{\text{cut}(A,B)}{\text{assoc}(B,V)}$$

Normalized Cut (Shi & Malik)



$$\text{Ncut}(A,B) = \frac{\text{cut}(A,B)}{\text{assoc}(A,V)} + \frac{\text{cut}(A,B)}{\text{assoc}(B,V)}$$

$$\mathbf{W} = [w_{i,j}]$$

Unfortunately, computing the optimal normalized cut is NP-complete.

Normalized Cut (Shi & Malik)

Let x be the indicator vector where $x_i = +1$ iff $i \in A$ and $x_i = -1$ iff $i \in B$.

Let $d = W\mathbf{1}$ be the row sums of the symmetric matrix W and $D = \mathrm{diag}(d)$

Shi and Malik show that minimizing the normalized cut over all possible indicator vectors x is equivalent to minimizing

Rayleigh quotient

$$\min_y \frac{y^T(D-W)y}{y^T D y}$$

where, $y = ((1+x) - b(1-x))/2$ such that $y \cdot d = 0$.

a vector of all ones and b's

Normalized Cut (Shi & Malik)

$$\min_y \frac{y^T(D-W)y}{y^T D y},$$

Minimizing this Rayleigh quotient is equivalent to solving the generalized eigenvalue system

$$(D-W)y = \lambda D y,$$

which can be turned into a regular eigenvalue problem

$$(I-N)z = \lambda z,$$

where $N = D^{-1/2} W D^{-1/2}$ and $z = D^{1/2} y$.

Normalized Affinity Matrix (Weiss 1999)

Normalized Cut (Shi & Malik)

Pixel-wise affinities:

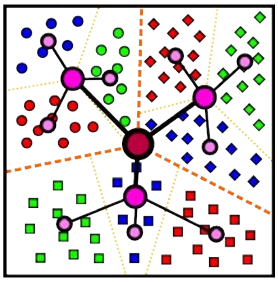
$$w_{ij} = \exp\left(-\frac{\|\boldsymbol{F}_i - \boldsymbol{F}_j\|^2}{\sigma_F^2} - \frac{\|\boldsymbol{x}_i - \boldsymbol{x}_j\|^2}{\sigma_s^2}\right)$$

F is a feature vector that consists of intensities, colors, or oriented filter histograms.

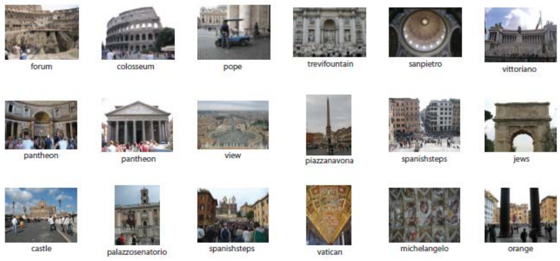
Normalized cuts for segmentation

Which algorithm to use?

- Quantization/Summarization: K-means
  - Aims to preserve variance of original data
  - Can easily assign new point to a cluster



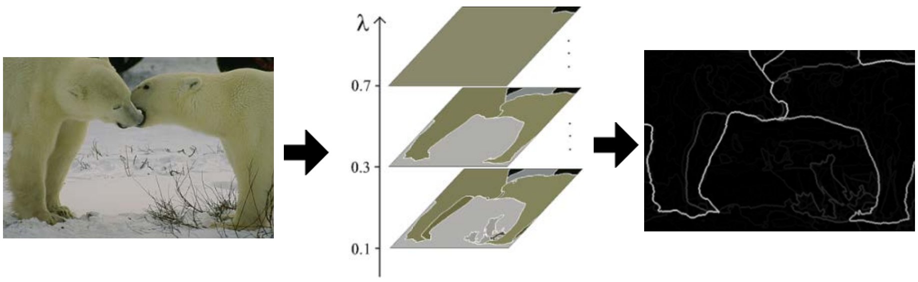
Quantization for computing histograms



Summary of 20,000 photos of Rome using "greedy k-means"
http://grail.cs.washington.edu/projects/canonview/

Which algorithm to use?

- Image segmentation: agglomerative clustering
  - More flexible with distance measures (e.g., can be based on boundary prediction)
  - Adapts better to specific data
  - Hierarchy can be useful

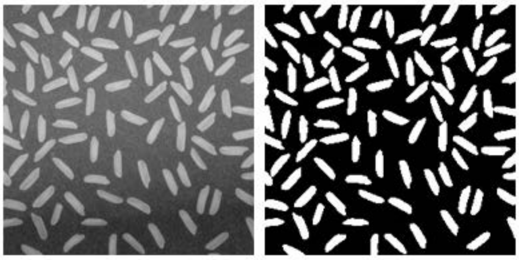


http://www.cs.berkeley.edu/~arbelaez/UCM.html

Segmentation in MATLAB

- https://www.mathworks.com/discovery/image-segmentation.html

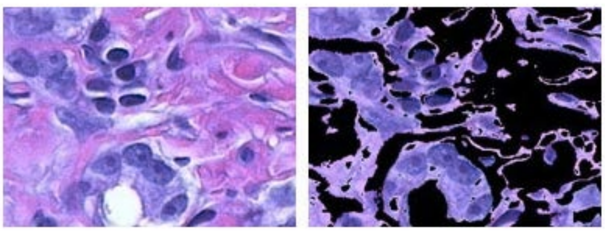
  - Thresholding methods such as Otsu's method



Segmentation in MATLAB

- https://www.mathworks.com/discovery/image-segmentation.html

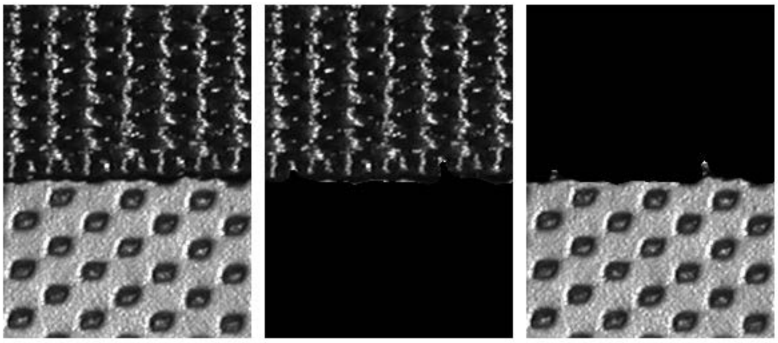
  - Color-based Segmentation such as K-means clustering



Segmentation in MATLAB

- https://www.mathworks.com/discovery/image-segmentation.html

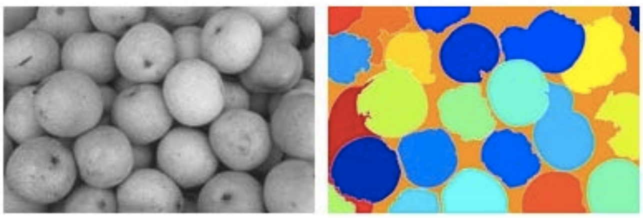
- Texture methods such as texture filters



Segmentation in MATLAB

- https://www.mathworks.com/discovery/image-segmentation.html

- Transform methods such as watershed segmentation



Prior based segmentation

supervised/unsupervised
top-down – bottom-up
segmentation

Riklin Raviv et al, ECCV 2004, ICCV 2005, IJCV 2007

Prior based segmentation

Co-segmentation/Mutual Segmentation

Riklin-Raviv et al CVPR workshop (POCV) 2006, IJCV 2008

Prior based segmentation

Symmetry based segmentation

Riklin Raviv et al, CVPR 2006, IEEE TPAMI 2009