

# Cluster analysis – advanced and special algorithms

---

**Jiří Kléma**

Department of Computer Science,  
Czech Technical University in Prague

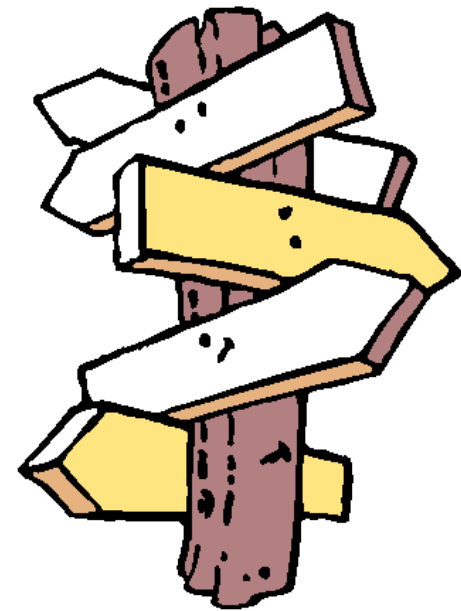


<http://cw.felk.cvut.cz/wiki/courses/b4m36san/start>

# Outline

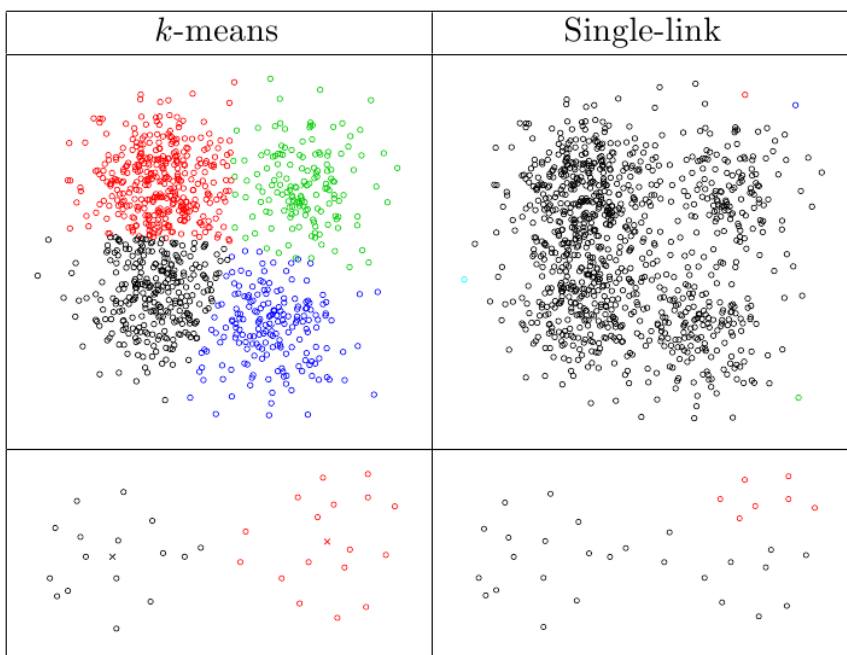
---

- robustness of the introduced methods – examples,
- k-means and hierarchical agglomerative clustering – complexity,
- clustering quality – evaluation
  - internal versus external partitioning evaluation,
- clustering definition?
- an advanced method
  - spectral clustering,

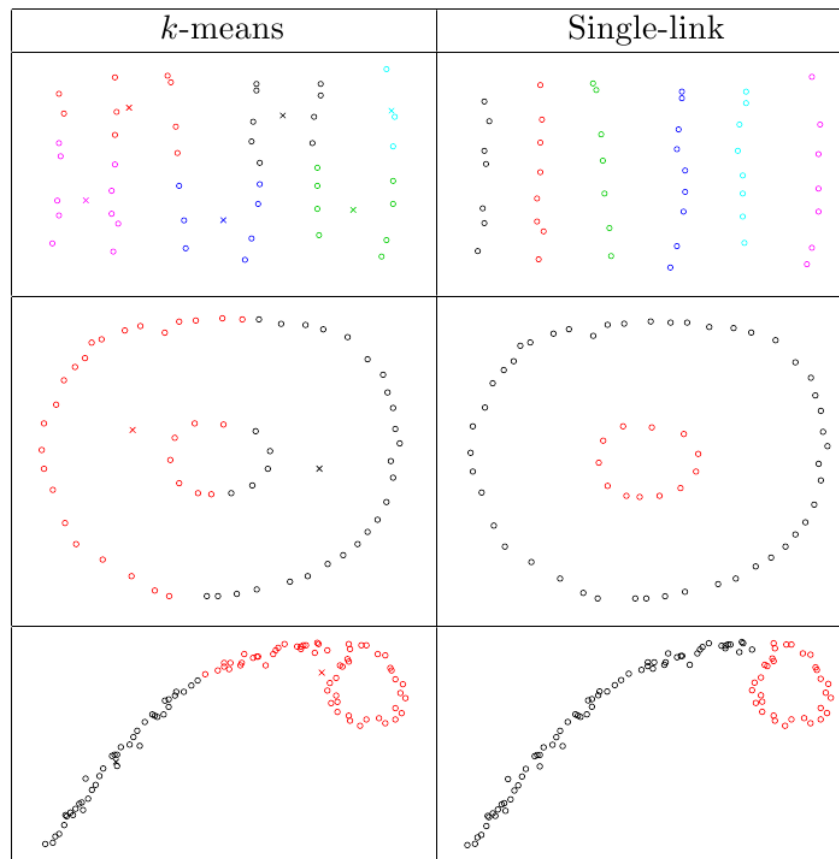


# Comparison: k-means and hierarchical single-link

- single linkage tends to generate longer non-compact clusters,
- k-means makes compact clusters, complete linkage is outlier sensitive,



k-means intuitively correct



single linkage intuitively correct

Carnegie Mellon University, course: Statistics 36-350: Data Mining

## Complexity – comparison

---

- assumed:  $d(x_i, x_j) \in \mathcal{O}(n)$ ,
- k-means algorithm
  - assign instances into clusters:  $\mathcal{O}(km)$  distance computations  $\rightarrow f_E \in \mathcal{O}(knm)$ ,
  - modify centroids:  $f_M \in \mathcal{O}(nm)$  (each instance used exactly once in one of the centroids),
  - unknown iteration number before stop:  $i$  (estimates vary from the constant with  $m$  up to  $\mathcal{O}(m^{kn})$ ),
  - summary:  $f = i(f_E + f_M) \in \mathcal{O}(iknm)$ ,
- hierarchical agglomerative clustering (single link)
  - initialize
    - \* compute distances among all instance pairs  $f_I \in \mathcal{O}(m^2n)$ ,
    - \* *next-best-merge* array – the nearest neighbor with its distance for each cluster (complexity hidden in the previous line),
  - $m - 1$  iterations to complete the dendrogram
    - \* find the smallest distance in the next-best-merge array  $f_{E_0} \in \mathcal{O}(m)$ ,
    - \* adjust the distance matrix  $f_{E_1} \in \mathcal{O}(m)$ ,
    - \* adjust the next-best-merge array  $f_{E_2} \in \mathcal{O}(m)$ ,
  - summary:  $f = f_I + (m - 1)(f_{E_0} + f_{E_1} + f_{E_2}) \in \mathcal{O}(m^2n)$ .

## Clustering quality – evaluation

---

- **internal:** quantifies three partitioning characteristics

- homogeneity – are instances within clusters similar?

$$hom = \frac{1}{m} \sum_{i=1}^k \sum_{x_j \in C_i} d(x_j, \mu_i),$$

- separability – are instances in different clusters dissimilar?

$$sep = \frac{1}{k(k-1)} \sum_{i=1}^k \sum_{j=1, j \neq i}^k \exp\left(-\frac{d^2(\mu_i, \mu_j)}{2\sigma^2}\right),$$

- stability – how many relations breaks up by adding noise or random instance sampling?

- internal evaluation criteria in the clustering algorithms

- intra-cluster variability: (see homogeneity for k-means),
- model likelihood: (for probabilistic models, see EM GMM),
- the size of the balanced cut of similarity graph: (see spectral clustering later)

$$\frac{1}{2} \sum_{ij} s(x_i, x_j) (1 - \delta(C_i - C_j)) \left(\frac{1}{|C_i|} + \frac{1}{|C_j|}\right),$$

- clustering is subjective, the “objective” internal measure can be improper.

## Clustering quality – evaluation

---

- **external:** match the partition  $\Omega$  with a known annotation  $G = \{G_1, \dots, G_c\}$  (gold standard),
- basic external evaluation criteria

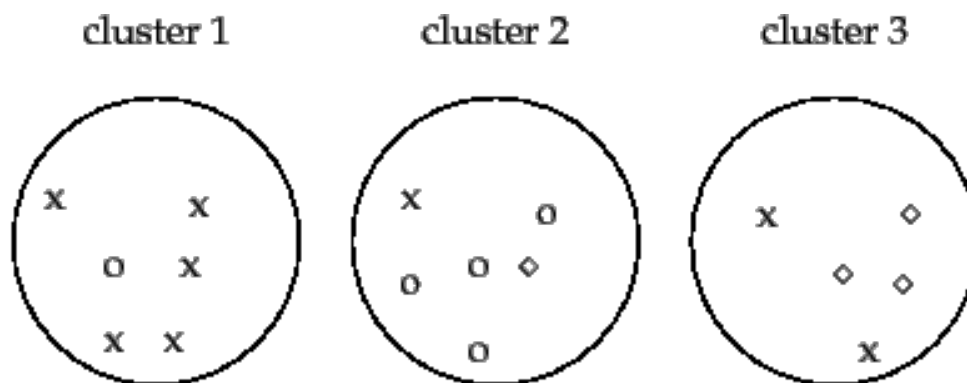
– purity

- \* the total major instance class ratio across clusters  
(each cluster has a major class, the higher its ratio in the cluster the better),

$$purity(\Omega, G) = \frac{1}{m} \sum_{i=1 \dots k} \max_{j=1 \dots c} |C_i \cap G_j|$$

- \* disadvantage: cannot compare partitions with different  $k$ ,

- \* example (figure):  $purity = \frac{5+4+3}{17} = 0.71$



Manning et al.: Introduction to Information Retrieval.  
<http://nlp.stanford.edu/IR-book/html/htmledition/irbook.html>

## Clustering quality – evaluation

---

- basic external evaluation criteria (continuation)

- normalized mutual information

- \* based on information entropy,

$$NMI(\Omega, G) = \frac{2I(\Omega, G)}{H(\Omega) + H(G)}$$

$$I(\Omega, G) = - \sum_{i=1 \dots k} \sum_{j=1 \dots c} P(C_i \cap G_j) \log_2 \left( \frac{P(C_i \cap G_j)}{P(C_i)P(G_j)} \right)$$

$$H(\Omega) = - \sum_{i=1 \dots k} P(C_i) \log_2(P(C_i)), \quad H(G) = - \sum_{j=1 \dots c} P(G_j) \log_2(P(G_j))$$

- \* example (figure):  $H(\Omega) = -2 \frac{6}{17} \log_2 \left( \frac{6}{17} \right) - \frac{5}{17} \log_2 \left( \frac{5}{17} \right) = 1.58$ ,  $H(G) = 1.52$

$$NMI = \frac{2 \times 0.96}{1.58 + 1.52} = 0.62$$

- rand index

- \* clustering is interpreted as a sequence of  $\binom{m}{2} = \frac{m(m-1)}{2}$  decisions,

- \* decision = the partitioning either puts a instance pair into the same cluster or not,

- \*  $TP$  ... an instance pair in the same cluster in  $\Omega$  and the same class in  $G$ ,

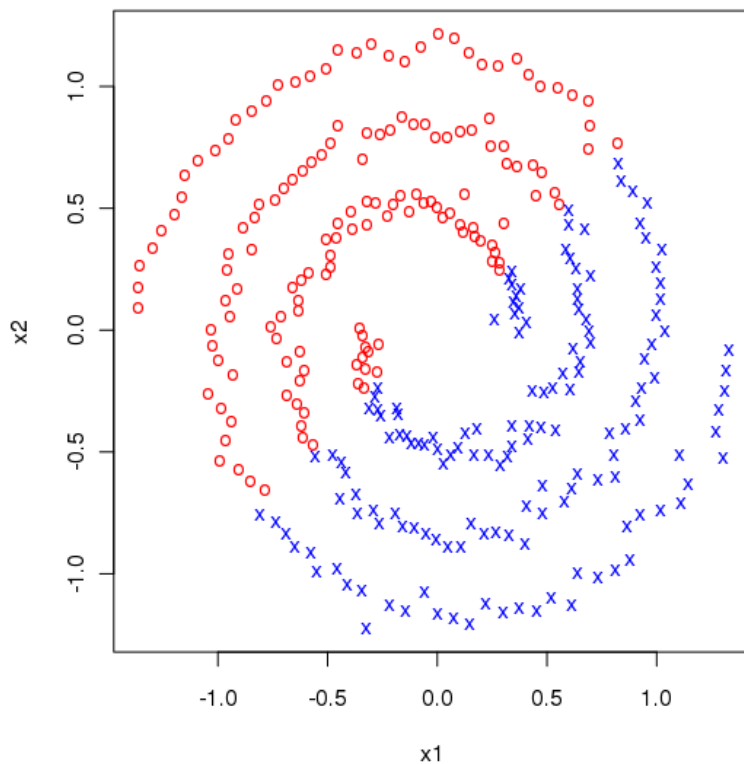
- \*  $TN$  ... an instance pair in different clusters in  $\Omega$  and different classes in  $G$ ,

- \*  $RI(\Omega, G) = \frac{TP + TN}{TP + TN + FP + FN} = \frac{2(TP + TN)}{m(m-1)}$

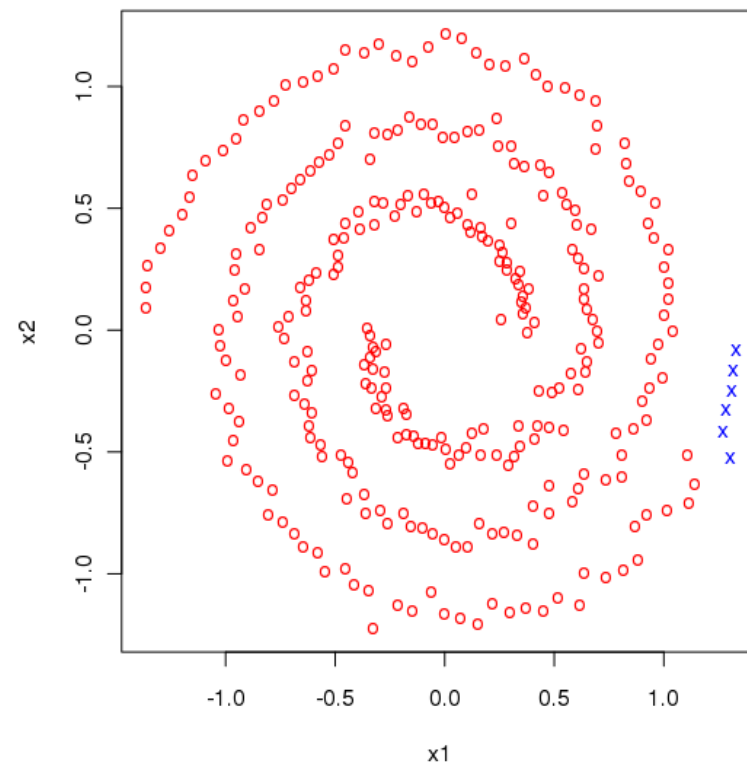
- \* example (figure):  $RI = \frac{2 \left( \binom{5}{2} + \binom{4}{2} + \binom{3}{2} + \binom{2}{2} + 5 \times 8 + 1 \times 7 + 4 \times 5 + 1 \times 2 + 1 \times 3 \right)}{17 \times 16} = 0.68$

# Spectral clustering – motivation

- clustering algorithms assume certain cluster shapes
  - unexpected shapes cause difficulties (eg. linearly non-separable clusters),
  - “classical pairwise similarity” can be insufficient.



K-means application



Single linkage application

R, kernlab package, specc function demo



## Spectral clustering – context

---

- frequent solution is a **feature space transformation**,
- a domain independent clustering algorithm, the transformation tuned for the domain
  - explicit transformation
    - \* get the object coordinates in the new feature space,
    - \* traditional clustering in the new space,
    - \* illustrative, but impractical,
  - implicit transformation
    - \* via similarity resp. kernel function  $K : \mathcal{X} \times \mathcal{X} \rightarrow \mathbb{R}$ ,
    - \* purely a function of object pairs, no object coordinates in the new space,
    - \* very natural for clustering, similarity/distance its essential part anyway,
  
    - \* kernel trick analogy (SVM classification),
      - kernel k-means (see the next slide),
    - \* an implicit high-dimensional space, clusters (classes) potentially easily separable,
  
    - \* kernel PCA – kernel matrix  $\rightarrow$  diagonalize  $\rightarrow$  a low-dimensional feature space.

## Kernel k-means

---

- apply k-means in the transformed feature space induced by a kernel function
  - the original objects:  $x_1, x_2, \dots, x_m$ ,
  - the transformed objects:  $\Phi(x_1), \Phi(x_2), \dots, \Phi(x_m)$  (not explicitly calculated),
  - the kernel function:  $k(x_i, x_j) = \langle \Phi(x_i), \Phi(x_j) \rangle$ ,
  - cluster centers in the transformed space:  $\mu_v = \frac{1}{|C_v|} \sum_{x_i \in C_v} \Phi(x_i)$  (not explicitly known),
  - only (squared) distances between objects and cluster centers need to be known:

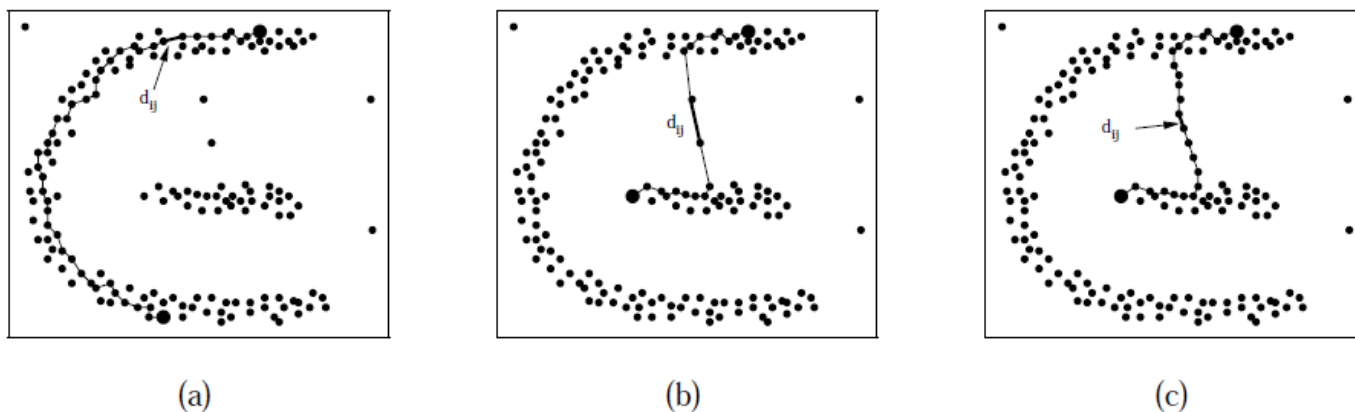
$$\begin{aligned} \|\Phi(x) - \mu_v\|^2 &= \left\| \Phi(x) - \frac{1}{|C_v|} \sum_{x_i \in C_v} \Phi(x_i) \right\|^2 = \\ &= \left\langle \Phi(x) - \frac{1}{|C_v|} \sum_{x_i \in C_v} \Phi(x_i), \Phi(x) - \frac{1}{|C_v|} \sum_{x_i \in C_v} \Phi(x_i) \right\rangle = \\ &= \langle \Phi(x), \Phi(x) \rangle - \frac{2}{|C_v|} \sum_{x_i \in C_v} \langle \Phi(x), \Phi(x_i) \rangle + \frac{1}{|C_v|^2} \sum_{x_i \in C_v, x_j \in C_v} \langle \Phi(x_i), \Phi(x_j) \rangle = \\ &= k(x, x) - \frac{2}{|C_v|} \sum_{x_i \in C_v} k(x, x_i) + \frac{1}{|C_v|^2} \sum_{x_i \in C_v, x_j \in C_v} k(x_i, x_j) \end{aligned}$$

## Example: spirals – connectivity kernel, Gaussian kernel

---

### ■ connectivity kernel

- the object pair distance given by the max edge on the path connecting the objects,
- if there are more paths, the one minimizing the criterion above is taken,
- e.g., this kernel makes k-means behave similar to single linkage hierarchical clustering,



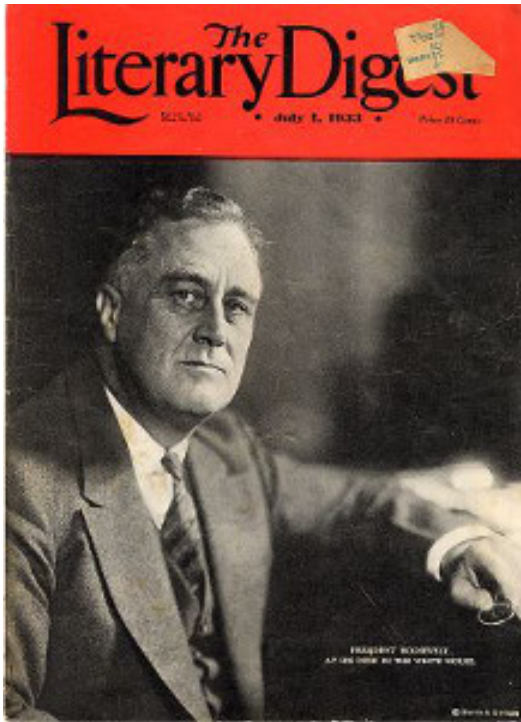
Fischer et al.: Clustering with the Connectivity Kernel

### ■ Gaussian (RBF) kernel

- $s(x_i, x_j) = \exp(-\|x_i - x_j\|/\sigma^2)$ ,
- $\sigma$  set to have a “tight” object neighborhood,
- an implicit feature space (infinite dimension).



## Famous statistical blunders ...



US presidential elections, 1936

FD Roosevelt - Alf Landon



Draft lottery, 1970

Vietnam war



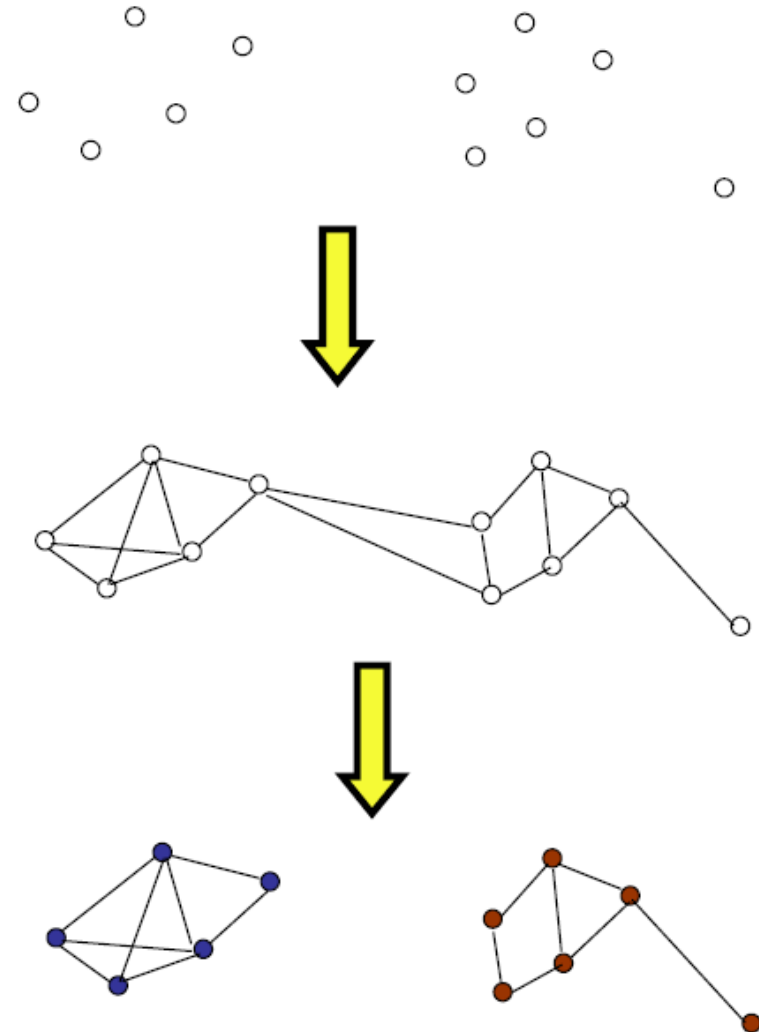
Financial crisis, 2008

Gaussian copula function

# Spectral clustering in a nutshell

---

- input: a set of objects,
- described as a graph,
  - edges encode similarity,
- graph decomposed into components = clusters,
- graph partitioned by its spectral properties.

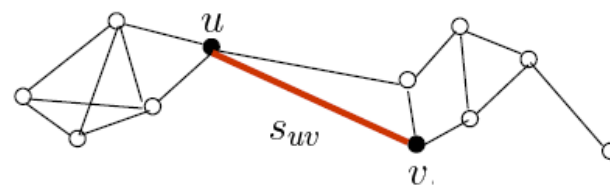


Azran: A Tutorial on Spectral Clustering

# Graph theory – basic terms

- vertex (object) similarity (affinity)

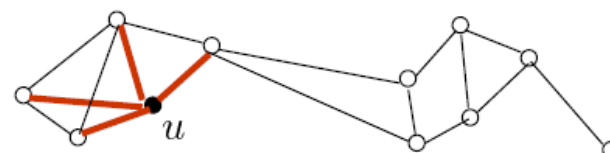
- $s_{uv} = \langle u, v \rangle$ ,



- vertex degree (volume), degree matrix

- $d_u = \sum_{v=1}^m s_{uv}$ ,

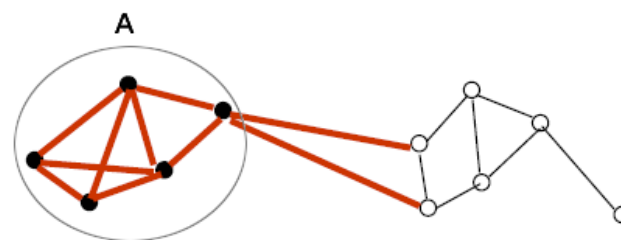
- $\mathcal{D} = \text{diag}(d_1, \dots, d_m)$ ,



- size and degree of a vertex set (cluster)

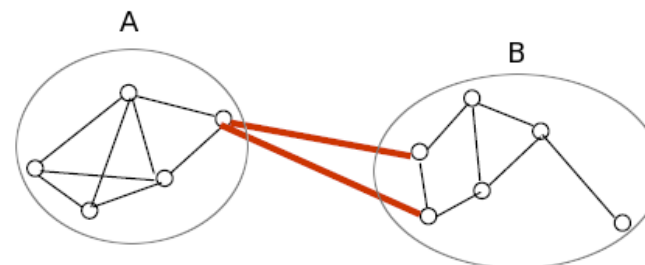
- $|A|$  ... the number of vertices in  $A$ ,

- $\text{vol}(A) = \sum_{u \in A} d_u$ ,



- an edge cut between two components

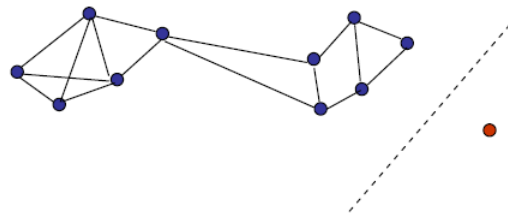
- $\text{cut}(A, B) = \sum_{u \in A} \sum_{v \in B} s_{uv}$ .



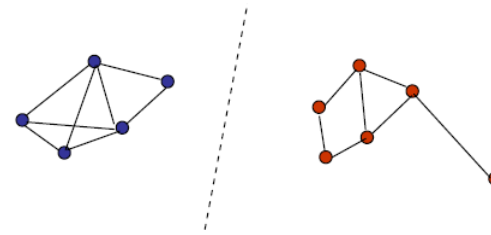
Azran: A Tutorial on Spectral Clustering

# Spectral clustering as an approximated minimum graph cut

- clustering  $\sim$  partition the similarity graph into components,
- can be solved as an optimization problem
  - search for a minimum edge cut in the similarity graph  $\mathcal{S}$  to make it disconnected
    - \*  $\min_{ACS} cut(A, \bar{A})$ ,
    - \* a computationally feasible problem, but rather unsatisfactory partitions,



mincut, incorrect



RatioCut, Ncut, correct

- a “reasonable” size of the components needs to be required
  - \* minimize one of the balanced cut criteria,
  - \*  $RatioCut(A, B) = cut(A, B) \left( \frac{1}{|A|} + \frac{1}{|B|} \right)$ ,
  - \*  $Ncut(A, B) = cut(A, B) \left( \frac{1}{vol(A)} + \frac{1}{vol(B)} \right)$ ,
  - \* the dark side of the coin: NP-hard problems,

- spectral clustering provides a relaxed and feasible solution to the balanced cut problem.

## Spectral clustering – algorithm

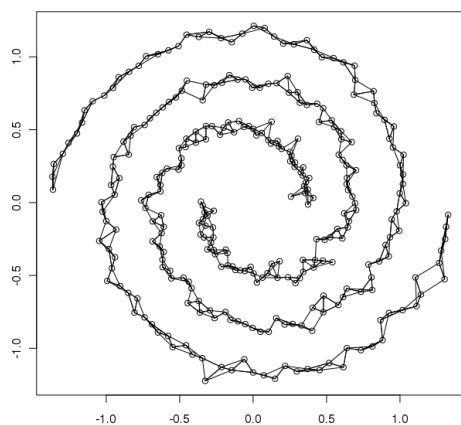
---

- inputs:  $\mathcal{X} = [x_{ij}]_{m \times n} = \{x_1, \dots, x_m\} \subset \mathbb{R}^n$ ,  $k$ 
  1. select the similarity function
    - linear, RBF, polynomial, etc.
    - a general rule assigning functions to problems does not exist,
  2. compute the similarity (adjacency) matrix  $\mathcal{S} = [s_{ij}]_{m \times m}$ 
    - (a new implicit feature space originates),
  3. construct a “reasonable” similarity graph by editing  $\mathcal{S}$ 
    - $\mathcal{S}$  is a complete graph, vertices  $\sim$  objects, similarities  $\sim$  edges,
    - remove long (improper) edges,
  4. derive the Laplace matrix  $\mathcal{L}$  out of the similarity matrix  $\mathcal{S}$ 
    - unnormalized:  $\mathcal{L} = \mathcal{D} - \mathcal{S}$ ,
    - normalized:  $\mathcal{L}_{rw} = \mathcal{D}^{-1}\mathcal{L} = \mathcal{I} - \mathcal{D}^{-1}\mathcal{S}$ ,
  5. project into an explicit space of  $k$  first eigenvectors of  $\mathcal{L}$ ,
    - $\mathcal{V} = [v_{ij}]_{m \times k}$ , eigenvectors of  $\mathcal{L}$  as columns,
  6. k-means clustering in  $\mathcal{V}$  matrix
    - $\mathcal{V}$  rows interpreted as new object positions in k-dimensional space.

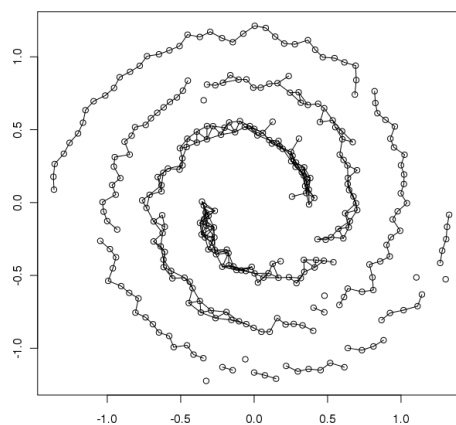


## Spectral clustering – similarity graph

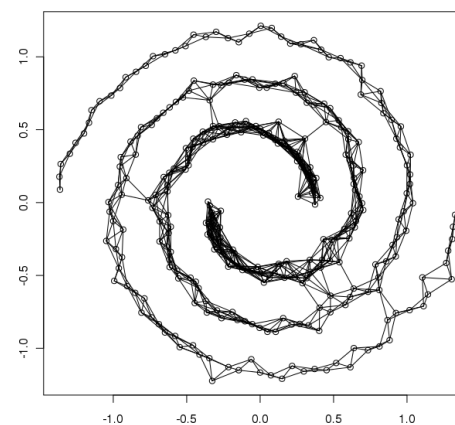
- reduce the complete graph to an undirected graph concerning local neighborhoods,
- vertices shall have a reasonable degree ( $\ll m$ ),
- basic approaches
  - $\epsilon$ -neighborhood
    - \*  $s_{ij} > \epsilon \rightarrow$  vertices  $i$  and  $j$  connected by an edge, otherwise  $s_{ij} = 0$ ,
  - k-nearest neighbors
    - \* symmetric: connect  $i$  and  $j$  if  $i$  belongs to  $k$  nearest neighbors of  $j$  **or** vice versa,
    - \* mutual: connect  $i$  and  $j$  if  $i$  belongs to  $k$  nearest neighbors of  $j$  **and** vice versa,
  - keep the complete graph
    - \* usually with the RBF or other strictly local kernel.



Euclidean similarity, 3 nearest neighbors (var sym)



RBF kernel, too small  $\epsilon$



RBF kernel, a suitable  $\epsilon$

## Spectral clustering – graph Laplacian

---

- concern the unnormalized option:  $\mathcal{L} = \mathcal{D} - \mathcal{S}$
- then for  $\forall f \in \mathbb{R}^m$

$$\begin{aligned} f' \mathcal{L} f &= f' \mathcal{D} f - f' \mathcal{S} f = \\ &= \sum_{i=1}^m d_i f_i^2 - \sum_{i,j=1}^m f_i f_j s_{ij} = \\ &= \frac{1}{2} \left( \sum_{i=1}^m \left( \sum_{j=1}^m s_{ij} \right) f_i^2 - 2 \sum_{i,j=1}^m f_i f_j s_{ij} + \sum_{j=1}^m \left( \sum_{i=1}^m s_{ij} \right) f_j^2 \right) = \\ &= \frac{1}{2} \sum_{i,j=1}^m s_{ij} (f_i - f_j)^2 \end{aligned}$$

- measures the variation of function  $f$  along the graph
  - the value  $f' \mathcal{L} f$  is low when close vertices agree in their  $f_i$ ,
  - assumes that near objects shall have close function values ( $f$ ),
- the discrete Laplace operator encodes the same property,
- an interesting case:  $f = \mathbb{1}_A$  ( $f_i = 1$  if  $v_i \in A$  otherwise  $f_i = 0$ ),  $A$  is a graph component.

## Spectral clustering – eigenvectors of $\mathcal{L}$

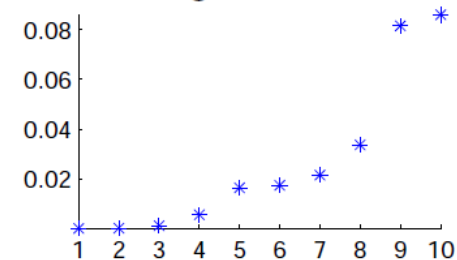
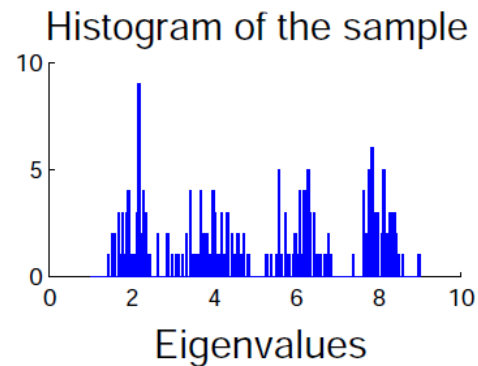
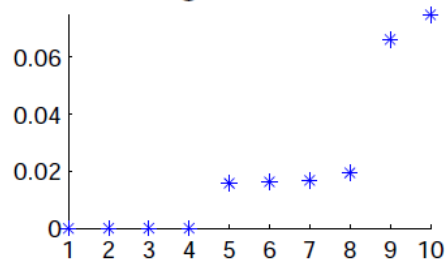
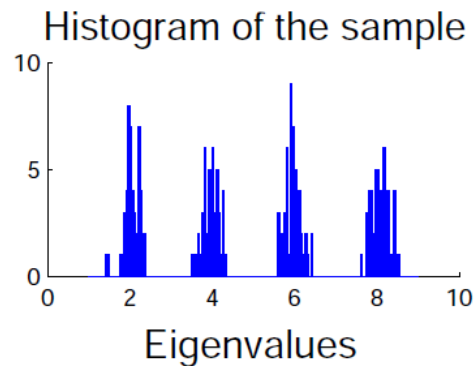
- eigenvectors  $x$  of  $\mathcal{L}$  matrix ( $\mathcal{L}x = \lambda x$ ) provide a good graph partitioning indication,
- an ultimate (ideal) case: graph has exactly  $k$  components
  - $k$  smallest eigenvectors ideally split  $k$  clusters,
  - $\lambda_1 = \dots = \lambda_k = 0 < \lambda_{k+1} \leq \dots \leq \lambda_m \rightarrow x_1, \dots, x_k$ ,
- other (usual) cases: a connected graph,  $k$  component candidates exist
  - the space of  $k$  smallest eigenvectors (with nonzero  $\lambda$ ) allows to form  $k$  clusters.

$$\begin{array}{c}
 \Sigma=0 \\
 \begin{array}{c}
 L \\
 \left[ \begin{array}{cccc}
 \mathbf{d}_1 & -s_{12} & -s_{1i} & 0 & 0 \\
 -s_{21} & \mathbf{d}_2 & -s_{2i} & 0 & 0 \\
 \dots & \dots & \dots & \dots & \dots \\
 -s_{i1} & -s_{i2} & \mathbf{d}_i & 0 & 0 \\
 0 & 0 & 0 & \mathbf{d}_{i+1} & -s_{(i+1)m} \\
 \dots & \dots & \dots & \dots & \dots \\
 0 & 0 & 0 & -s_{m(i+1)} & \mathbf{d}_m
 \end{array} \right]
 \end{array}
 \end{array}
 \begin{array}{c}
 x_1 \\
 \left[ \begin{array}{c}
 1 \\
 1 \\
 \dots \\
 1 \\
 0 \\
 \dots \\
 0
 \end{array} \right]
 \end{array}
 =
 \begin{array}{c}
 \lambda_1 x_1 \\
 \left[ \begin{array}{c}
 0 \\
 0 \\
 \dots \\
 0 \\
 0 \\
 \dots \\
 0
 \end{array} \right]
 \end{array}
 =
 \begin{array}{c}
 0 \\
 \left[ \begin{array}{c}
 1 \\
 1 \\
 \dots \\
 1 \\
 0 \\
 \dots \\
 0
 \end{array} \right]
 \end{array}$$

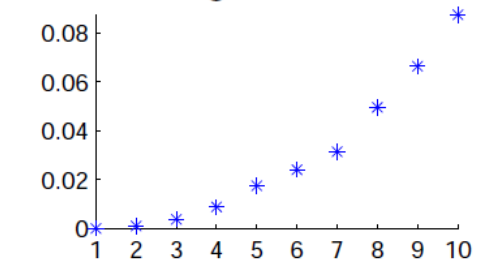
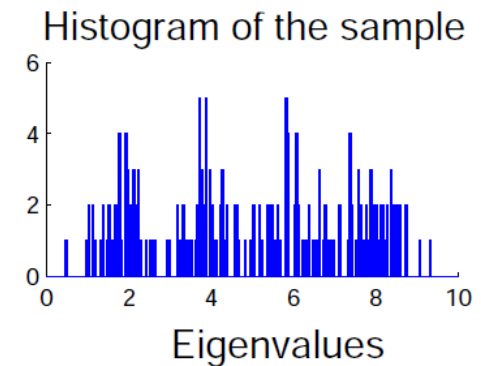
The ideal case for  $k = 2$ .

# Spectral clustering – eigenvalues of $\mathcal{L}$

- provided  $k$  is unknown, **eigengap** statistic
  - a k-means gap heuristic analogy,
  - concern only small eigenvectors before the first jump in eigenvalues,
  - the number of clusters matches the number of selected eigenvectors.

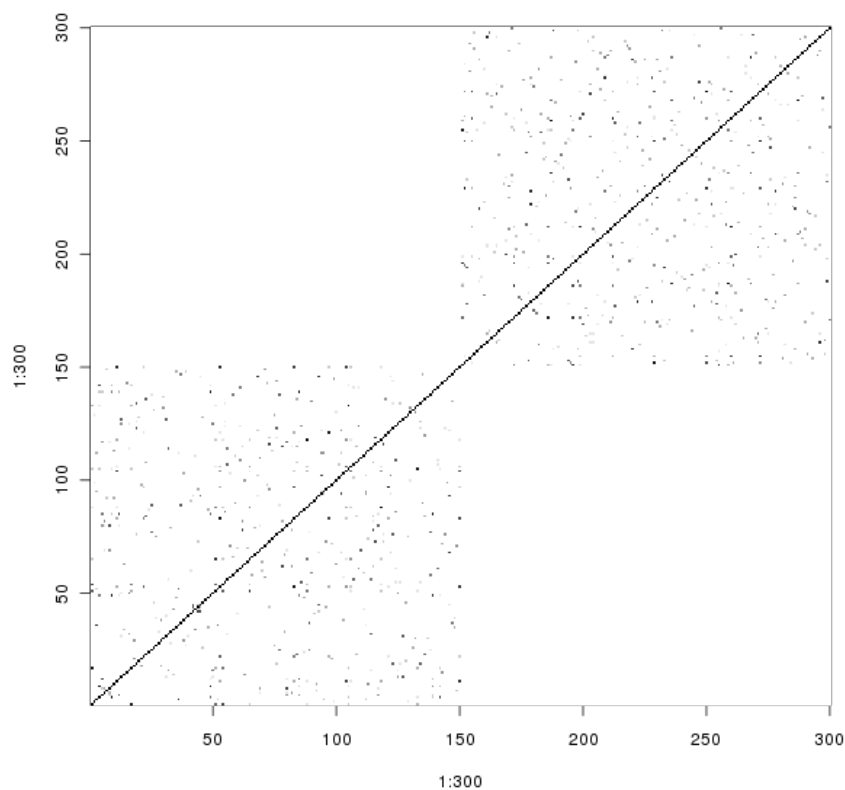


Luxburg: Clustering.

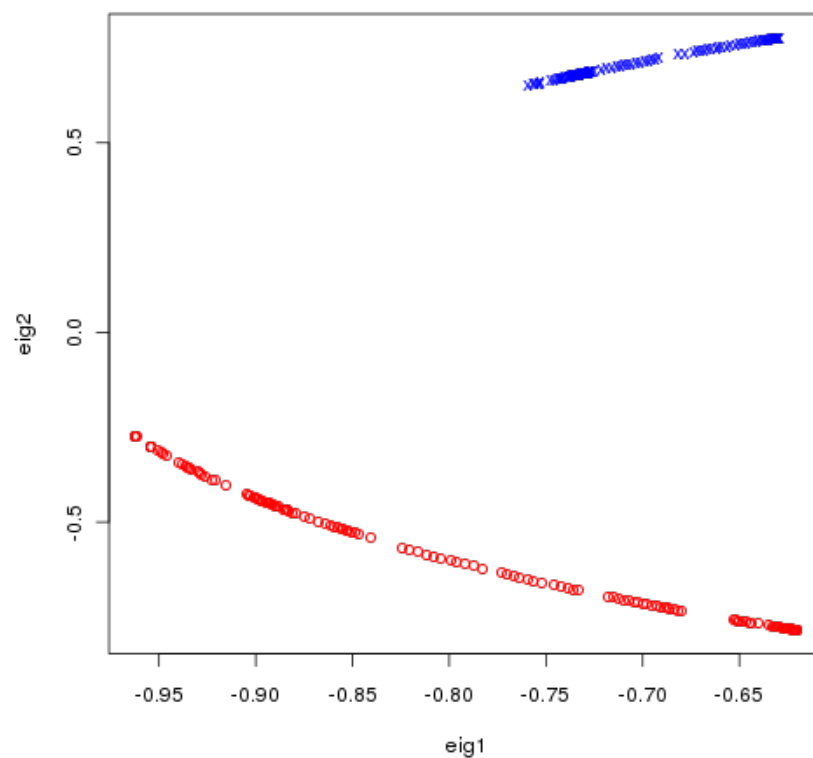


## Example: spirals – eigenvectors

- similarity matrix splits the graph into components nearly ideally,
- the second eigenvector of  $\mathcal{L}$  is a perfect component indicator.



Similarity matrix for RBF kernel with a suitable  $\sigma$   
the instance order is illustrative and keeps the real spiral membership



Projection – the first and second eigenvector space  $\mathcal{S}$   
colors give the real spiral membership, k-means clustering is trivial

## Spectral clustering – summary

---

### ■ advantages

- does not make strong assumptions on cluster shape,
- simple to implement – uses existing algorithms,
- does not have a local optima, cannot stuck,
- a modular approach applicable in a range of problems
  - \* modify the kernel or similarity graph to adapt to a new problem,
- eigengap heuristic to find an optimal cluster number,
- successful in a range of real problems,

### ■ disadvantages

- can be sensitive to choice of parameters, unclear how to set them,
  - \* kernels (eg.  $\sigma$  for RBF), graph similarity ( $\epsilon$  or  $k$ ),
- computationally expensive on large non-sparse graphs,
  - \* use only after simpler algorithms fail,
- not really clear what it does on non-regular graphs (e.g. power law graphs),

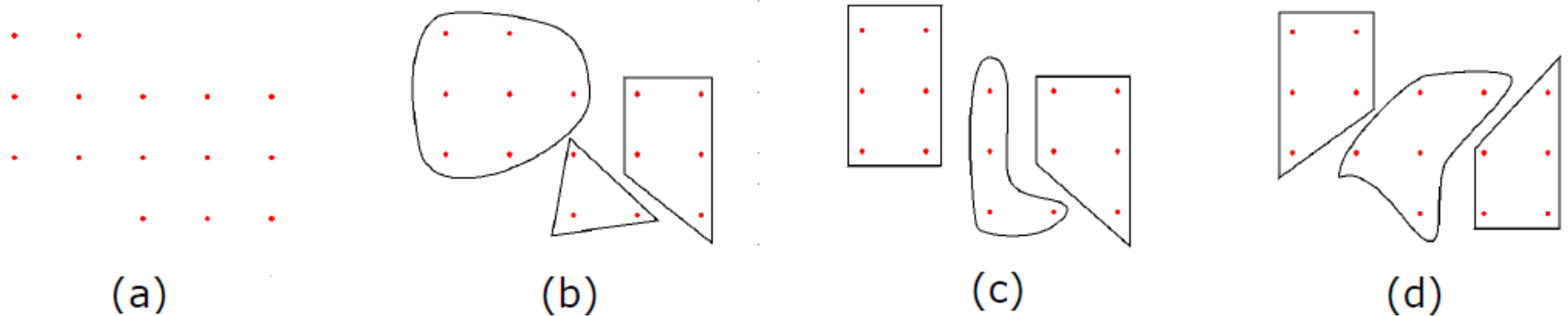
### ■ demo

- <http://www.ml.uni-saarland.de/GraphDemo/GraphDemo.html>.

## Advanced clustering – summary

---

- Clustering covers a wide range of methods
  - not merely an instance set partitioning in  $\mathbb{R}^n$  dealing with disjoint clusters,
  - in general, it discovers arbitrary frequent co-occurrence of events in data,
- unsupervised = subjective approach
  - “gold true” does not exist (compare with classification),
  - the outcome is influenced by the employed implicit and explicit knowledge,

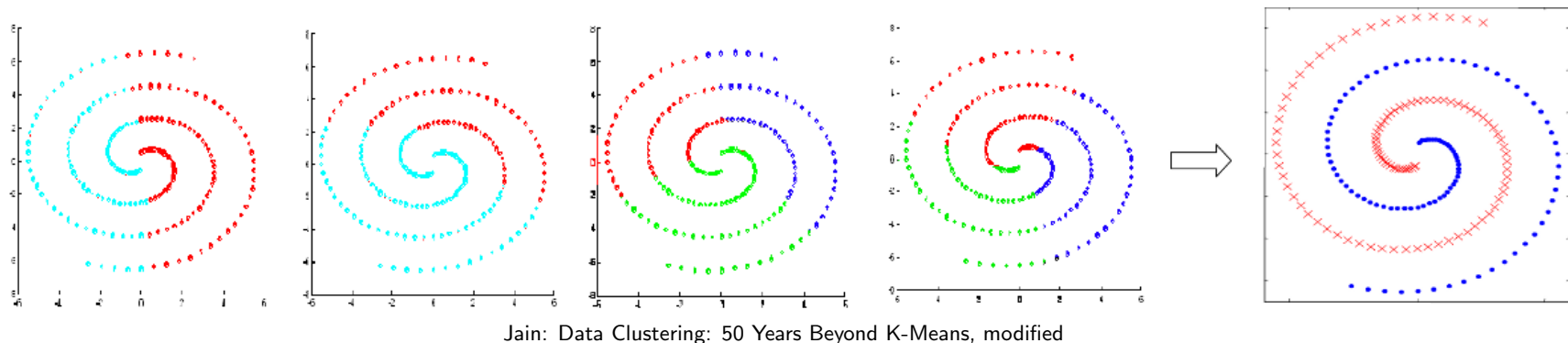


Jain: Data Clustering: 50 Years Beyond K-Means, modified

- tightly related to learning
  - conceptual clustering – knowledge-based with cluster/concept descriptions,
  - semi-supervised clustering – both annotated and unannotated instances,

## Advanced clustering – summary

- exists in many modifications
  - bi-clustering
    - \* rather the local (context-sensitive) than global similarity.
- topics not covered
  - heterogenous data
    - \* that cannot be represented as a constant-size feature vector,
  - large scale clustering
    - \* efficient NN, incremental clustering, sampling, distributed computing, prior data summarization,
  - clustering ensembles
    - \* the result obtained by combining multiple partitions.





## Recommended reading, lecture resources

---

### :: Reading

- von Luxburg: **Lectures on Clustering.**
  - PASCAL Bootcamp in Machine Learning, Vilanova (Barcelona), 2007,
  - [http://videlectures.net/bootcamp07\\_luxburg\\_clu/](http://videlectures.net/bootcamp07_luxburg_clu/),