(Rel.) c–Medoids (Krishnapuram '01)

- prototypes are data points

$$V \subseteq X$$

- for example

$$v_i = x_j \quad \Rightarrow \quad d_{ik} = \|v_i - x_k\| = \|x_j - x_k\| = r_{jk}$$

- can be used to cluster relational data R
- contribution of cluster i with $v_i = x_j$ to the FCM cost function

$$J_i^* = J_{ij} = \sum_{k=1}^{n} u_{ik}^m r_{jk}^2$$

- optimal choice of medoids

$$w_i = \text{arg min}\{J_{i1}, \ldots, J_{in}\}$$

Relational Fuzzy c–Means (Bezdek '87)

- reformulation: insert optimal V into J

$$J_{\mathsf{RFCM}}(U;R) = \sum_{i=1}^{c} \frac{\sum\limits_{j=1}^{n}\sum\limits_{k=1}^{n} u_{ij}^m u_{ik}^m r_{jk}^2}{\sum\limits_{j=1}^{n} u_{ij}^m}$$

- solution

$$u_{ik} = 1 \Big/ \sum_{j=1}^{n} \frac{\sum\limits_{s=1}^{n} \frac{u_{is}^m r_{sk}}{\sum\limits_{r=1}^{n} u_{ir}^m} - \sum\limits_{s=1}^{n}\sum\limits_{t=1}^{n} \frac{u_{is}^m u_{it}^m r_{st}}{2\left(\sum\limits_{r=1}^{n} u_{ir}^m\right)^2}}{\sum\limits_{s=1}^{n} \frac{u_{js}^m r_{sk}}{\sum\limits_{r=1}^{n} u_{jr}^m} - \sum\limits_{s=1}^{n}\sum\limits_{t=1}^{n} \frac{u_{js}^m u_{jt}^m r_{st}}{2\left(\sum\limits_{r=1}^{n} u_{jr}^m\right)^2}}$$

Non–Euclidean Relations

- problem with RFCM:

$$u_{ik} < 0 \quad \text{or} \quad u_{ik} > 1$$

  if R is not Euclidean (triangle inequality violated)
- solution: transformation of the distance matrix

$$D_\beta = D + \beta \cdot \begin{pmatrix} 0 & 1 & \ldots & 1 & 1 \\ 1 & 0 & \ldots & 1 & 1 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 1 & 1 & \ldots & 0 & 1 \\ 1 & 1 & \ldots & 1 & 0 \end{pmatrix}$$

  with successively increasing $\beta > 0$
- non–Euclidean relational fuzzy c–means (NERFCM)

Prof. Dr. Thomas A. Runkler

Mercer's Theorem (again)

- idea: transform the data $X = \{x_1, \ldots, x_n\} \in \mathrm{IR}^p$ to $X' = \{x'_1, \ldots, x'_n\} \in \mathrm{IR}^q$, $q \gg p$, so that the structure in X' is easier than in X
- support vector machine: non linearly separable data $X \rightarrow$ linearly separable data X'
- relational clustering: complex cluster shapes in $R \rightarrow$ hyperspherical clusters in R'
- Mercer's theorem $\exists$ a mapping $\varphi : \mathrm{IR}^p \rightarrow \mathrm{IR}^q$ so that

$$k(x_j, x_k) = \varphi(x_j) \cdot \varphi(x_k)^T$$

- kernel trick: scalar product in $X' =$ kernel function in X

Kernelization

- relational data

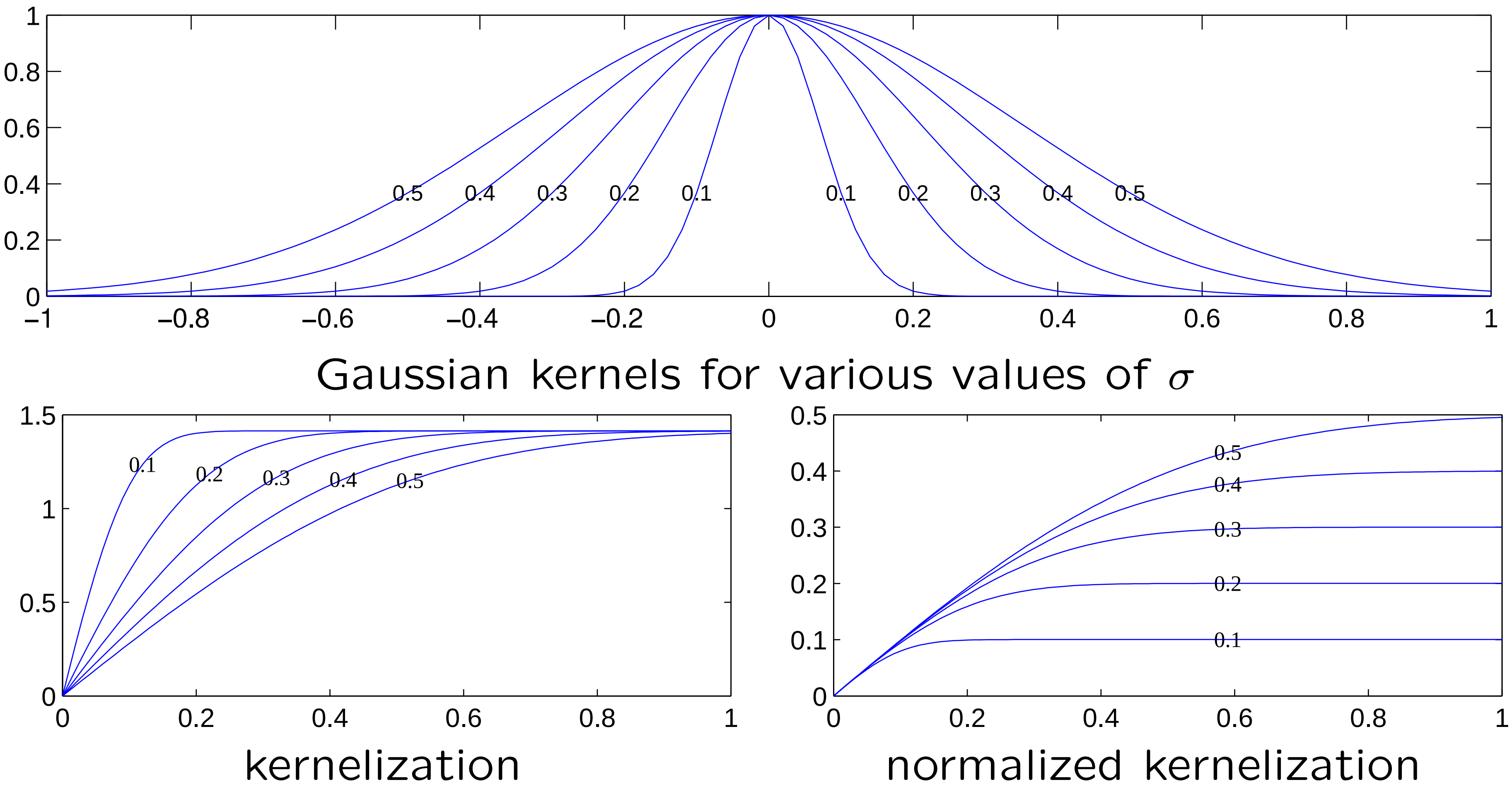
$$
\begin{aligned}
r_{jk}^2 &= \|\varphi(x_j) - \varphi(x_k)\|^2 \\
&= \Big(\varphi(x_j) - \varphi(x_k)\Big)\Big(\varphi(x_j) - \varphi(x_k)\Big)^T \\
&= \varphi(x_j)\varphi(x_j)^T - 2\,\varphi(x_j)\varphi(x_k)^T + \varphi(x_k)\varphi(x_k)^T \\
&= k(x_j, x_j) - 2 \cdot k(x_j, x_k) + k(x_k, x_k) \\
&= 2 - 2 \cdot k(x_j, x_k)
\end{aligned}
$$

- kernelization as preprocessing of R (kNERFCM)

$$
\begin{aligned}
r'_{jk} &= \sqrt{2 - 2 \cdot e^{-\frac{r_{jk}^2}{\sigma^2}}} \\
r'_{jk} &= \sqrt{2 \cdot \tanh\left(\frac{r_{jk}^2}{\sigma^2}\right)}
\end{aligned}
$$

Prof. Dr. Thomas A. Runkler

Effect of Kernelization



Prof. Dr. Thomas A. Runkler

Cluster Tendency: Hopkins Index

- $R = \{r_1, \ldots, r_m\}$: random points in the convex hull of X
- $S = \{s_1, \ldots, s_m\}$: randomly picked data from X, $m << n$
- $d_{r_1}, \ldots, d_{r_m}$: distances of R to the nearest neighbors in X
- $d_{s_1}, \ldots, d_{s_m}$: distances of S to the nearest neighbors in X
- Hopkins index

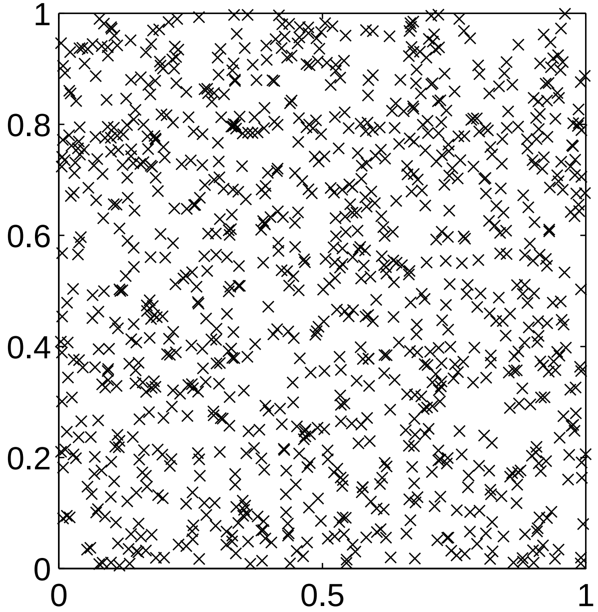
$$h = \frac{\sum\limits_{i=1}^{m} d_{r_i}^p}{\sum\limits_{i=1}^{m} d_{r_i}^p + \sum\limits_{i=1}^{m} d_{s_i}^p}$$

- interpretation:

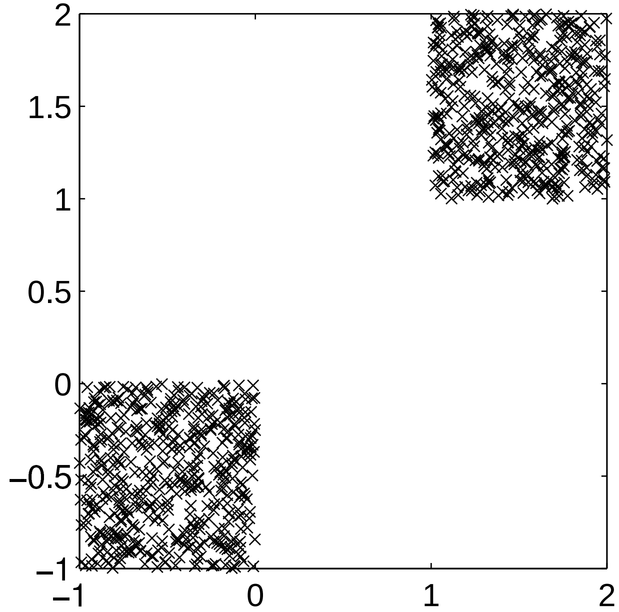
$$\begin{array}{lll} h \approx 0 & \leftrightarrow & X \text{ has regular structure} \\ h \approx 0.5 & \leftrightarrow & X \text{ is randomly distributed} \\ h \approx 1 & \leftrightarrow & X \text{ contains clusters} \end{array}$$

Prof. Dr. Thomas A. Runkler

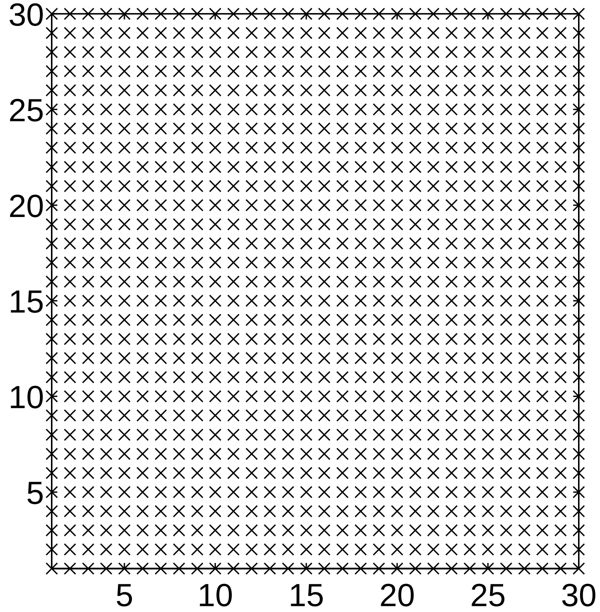
Examples Hopkins Index



$h \approx 0.4229$



$h \approx 0.9988$



$h \approx 0.1664$

Validity Measures

- partition coefficient
  (average square membership)

$$PC = \frac{1}{n} \sum_{k=1}^{n} \sum_{i=1}^{c} u_{ik}^2$$

- classification entropy
  (average entropy)

$$CE = \frac{1}{n} \sum_{k=1}^{n} \sum_{i=1}^{c} -u_{ik} \cdot \log u_{ik}$$

U	$PC(U)$	$CE(U)$
$\begin{pmatrix} 1 & 0 & \ldots & 0 \\ 0 & 1 & & 0 \\ \vdots & & \ddots & \vdots \\ 0 & 0 & \ldots & 1 \end{pmatrix}$	1	0
$\begin{pmatrix} \frac{1}{c} & \cdots & \frac{1}{c} \\ \vdots & \ddots & \vdots \\ \frac{1}{c} & \cdots & \frac{1}{c} \end{pmatrix}$	$\frac{1}{c}$	$\log c$

Self–Organizing Map (Kohonen ’81)

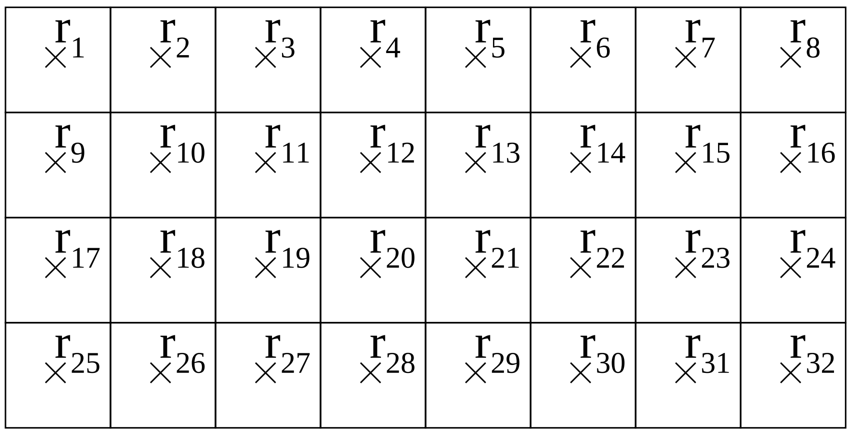
- q–dimensional array of nodes with reference vectors

$$M = \{m_1, \ldots, m_l\} \subset \mathbb{R}^p$$

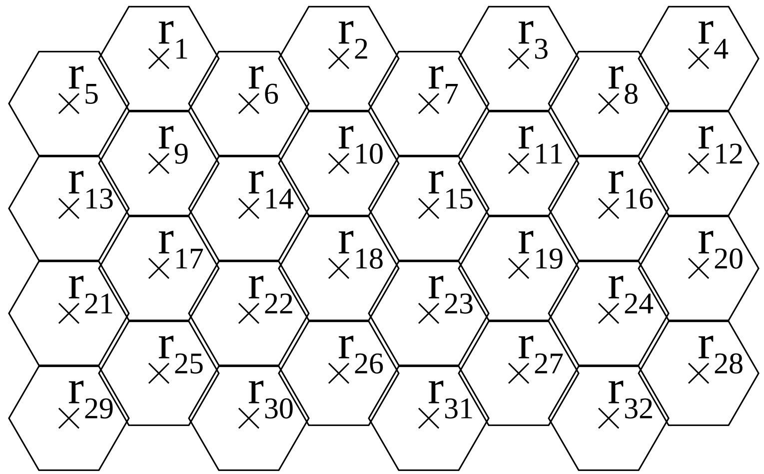
and locations

$$R = \{r_1, \ldots, r_l\} \subset \mathbb{R}^q$$

rectangular



hexagonal



Prof. Dr. Thomas A. Runkler

Self–Organizing Map

- in each learning step t consider the neighbors of each node
- neighborhood between nodes with indices c and i

$$\text{bubble:} \quad h_{ci} = \begin{cases} \alpha(t), & \text{if } \|r_c - r_i\| < \rho(t) \\ 0, & \text{otherwise} \end{cases}$$

$$\text{Gaussian:} \quad h_{ci} = \alpha(t) \cdot e^{-\frac{\|r_c - r_i\|^2}{2 \cdot \rho^2(t)}},$$

- observation radius $\rho(t)$: monotonically decreasing
- learning rate $\alpha(t)$: monotonically decreasing, e.g.

$$\alpha(t) = \frac{A}{B+t}, \quad A, B > 0$$

Self–Organizing Map

- algorithm
  1. input data $X = \{x_1, \ldots, x_n\} \subset R^p$,
     map dimension $q \in \{1, \ldots, p-1\}$,
     node positions $R = \{r_1, \ldots, r_l\} \subset R^q$
  2. initialize $M = \{m_1, \ldots, m_l\} \subset R^p$, $t = 1$
  3. for each x_k, $k = 1, \ldots, n$,
     (a) find winner node m_c with
     $$\|x_k - m_c\| \leq \|x_k - m_i\| \quad \forall i = 1, \ldots, l$$
     (b) update winner and neighbors
     $$m_i = m_i + h_{ci} \cdot (x_k - m_c) \quad \forall i = 1, \ldots, l$$
  4. $t = t + 1$
  5. repeat from (3.) until termination criterion holds
  6. output reference vectors $M = \{m_1, \ldots, m_l\} \subset R^p$

Prof. Dr. Thomas A. Runkler