CENTER FOR MACHINE PERCEPTION

K-means Clustering and its Generalization

authors: J. Matas, T. Werner

lecturer: V. Franc

Formulation of the LSQ clustering problem

In:	$\mathcal{T} = \{\mathbf{x_l}\}_{l=1}^{L}$,	the set of observations
Out:	$(\mathbf{c_k})_{k=1}^{K}$,	the set of cluster prototypes (etalons)
	$\{\mathcal{T}_k\}_{k=1}^{K}$	the clustering (partitioning) of the data

Formulation of the least squares clustering problem:

$$J(\mathbf{c_1}, \mathbf{c_1}, \dots, \mathbf{c_K}) = \sum_{i=1}^{L} \min_k ||\mathbf{x_l} - \mathbf{c_k}||_2^2$$

$$\boxed{(\mathbf{c_1^\star}, \mathbf{c_1^\star}, \dots, \mathbf{c_K^\star}) = \arg\min J(.)}$$

Alternative formulation:

$$J'(\mathbf{c_1}, \mathbf{c_1}, \dots, \mathbf{c_K}; \mathcal{T}_1, \mathcal{T}_2, \dots, \mathcal{T}_K) = \sum_{k=1}^{K} \sum_{\mathbf{x} \in \mathcal{T}_k} ||\mathbf{x} - \mathbf{c_k}||_2^2,$$

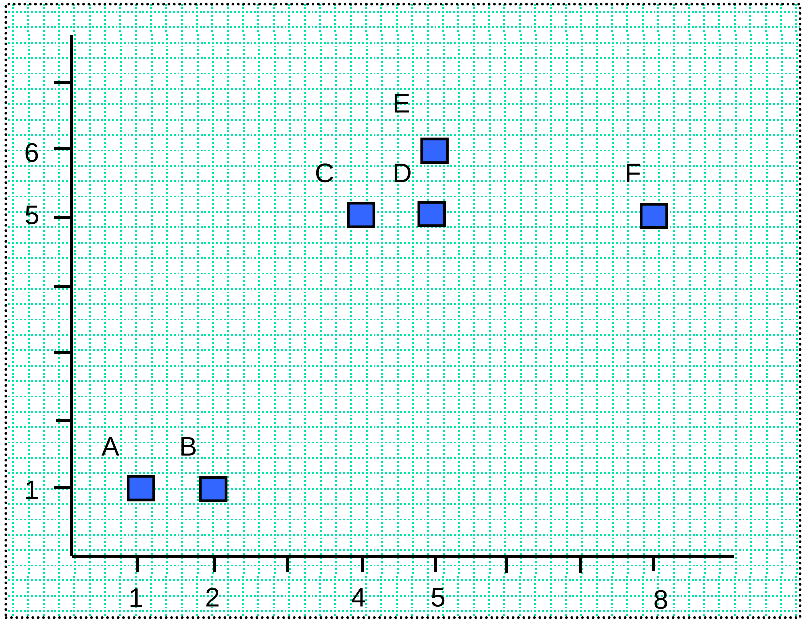
where $\mathcal{T}_k = \{\mathbf{x_l} \in \mathcal{T} | \forall j ||\mathbf{x} - \mathbf{c_k}||_2^2 \leq ||\mathbf{x} - \mathbf{c_j}||_2^2\}$

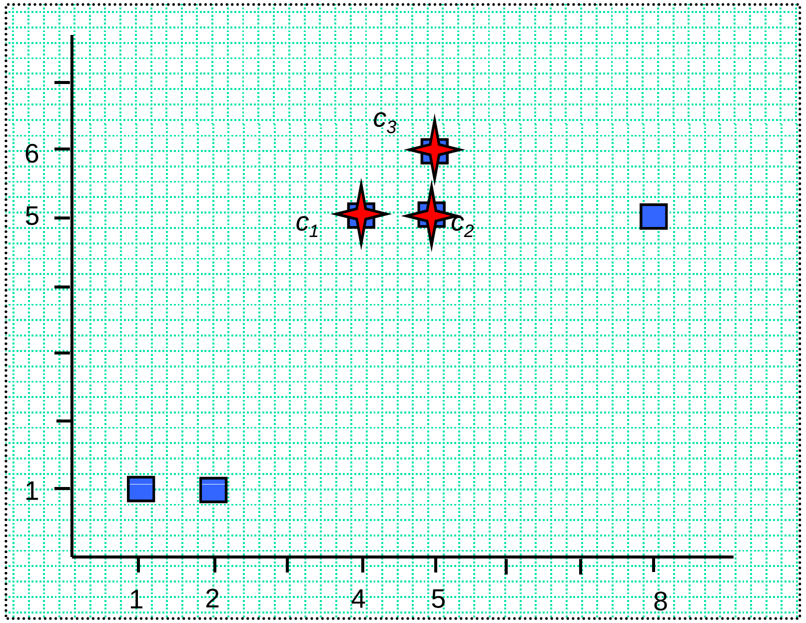
K-means: Algorithm for the LS Clustering Problem

In:	$\mathcal{T} = \{\mathbf{x_l}\}_{l=1}^{L}$,	the set of observations, $\mathbf{x} \in \mathbb{R}^D$
Out:	$(\mathbf{c_k})_{k=1}^{K}$,	the set of cluster prototypes (etalons), $\mathbf{c} \in \mathbb{R}^D$
	$\{\mathcal{T}_k\}_{k=1}^{K}$	the clustering (partitioning) of the data

1. Initialize $\mathbf{c_k}$ (e.g. by assigning random $\mathbf{x_l}$ to $\mathbf{c_k}$)
2. Assignment optimization:
   $\mathcal{T}_k = \{\mathbf{x} \in \mathcal{T} : \forall j, ||\mathbf{x} - \mathbf{c_k}||_2^2 \leq ||\mathbf{x} - \mathbf{c_j}||_2^2\}$
3. Prototype optimization:
   $\mathbf{c_k} = \frac{1}{|\mathcal{T}_k|} \sum_{\mathbf{x} \in \mathcal{T}_k} \mathbf{x}$
4. Terminate If $\mathcal{T}_k^{t+1} = \mathcal{T}_k^{t}, \forall k$; else go to 2

K-means: an example



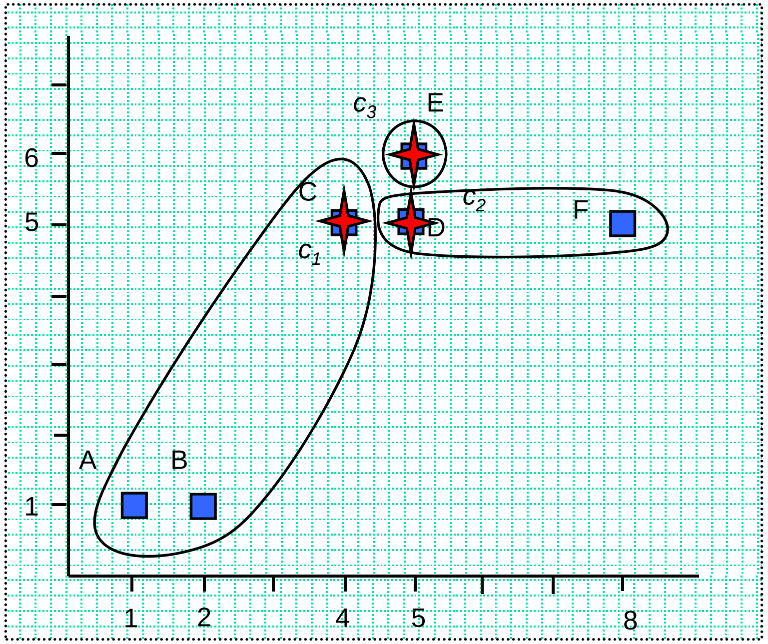


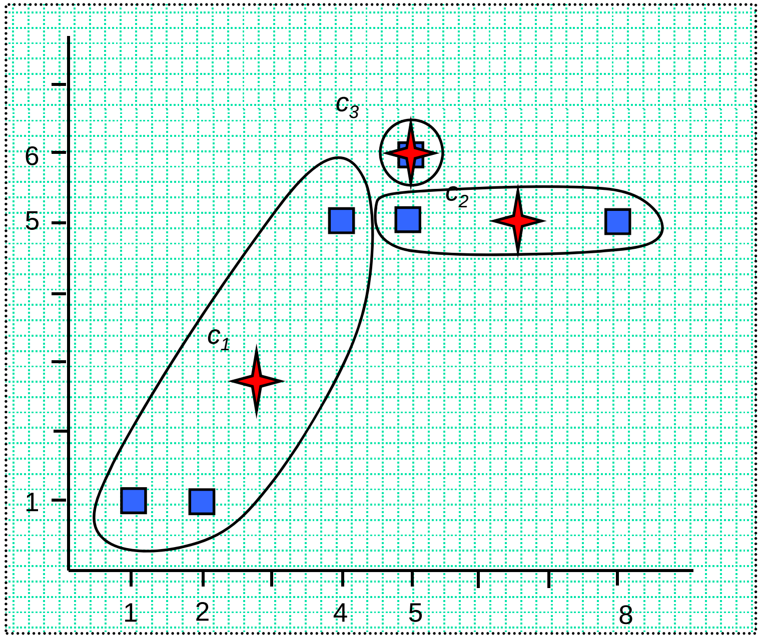
Number of clusters $K=3$

Initialization:

$\boldsymbol{c}_k = \text{random}(\boldsymbol{x}_l)$,
without replacement

K-means: an example





Optimizing partitions:

Euclidean Distances

$$
\begin{array}{c}
 \\ c_1 \\ c_2 \\ c_3
\end{array}
\begin{array}{c}
\begin{array}{cccccc} \mathrm{A} & \mathrm{B} & \mathrm{C} & \mathrm{D} & \mathrm{E} & \mathrm{F} \end{array} \\
\begin{pmatrix}
\mathbf{5} & \mathbf{4{,}5} & \mathbf{0} & 1 & 1{,}4 & 4 \\
5{,}7 & 5 & 1 & \mathbf{0} & 1 & \mathbf{3} \\
6{,}4 & 5{,}8 & 1{,}4 & 1 & \mathbf{0} & 3{,}2
\end{pmatrix}
\end{array}
$$

Sum of squares $= J^1(.) = 9.0$

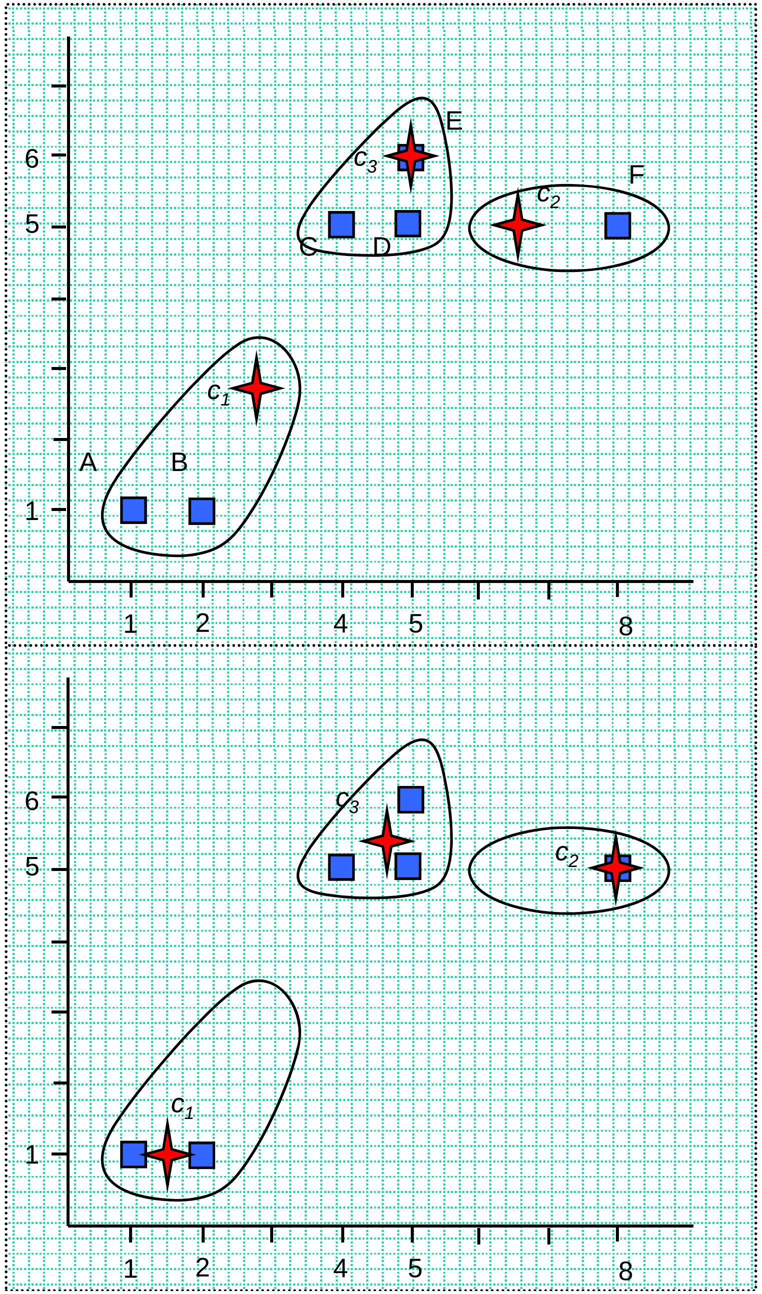
Optimizing prototypes:

$$c_1 = \left(\frac{1+2+4}{3}, \frac{1+1+5}{3}\right) = (2.3, 2.3)$$

$$c_2 = \left(\frac{5+8}{2}, \frac{5+5}{2}\right) = (6.5, 5)$$

$$c_3 = (5, 6)$$

K-means: an example (3/4)



Optimizing partitions:

Euclidean Distances

$$\begin{array}{c} \\ c_1 \\ c_2 \\ c_3 \end{array} \begin{array}{c} \begin{array}{cccccc} \mathrm{A} & \mathrm{B} & \mathrm{C} & \mathrm{D} & \mathrm{E} & \mathrm{F} \end{array} \\ \begin{pmatrix} \mathbf{1{,}9} & \mathbf{1{,}4} & 3{,}1 & 3{,}8 & 4{,}5 & 6{,}3 \\ 6{,}8 & 6 & 2{,}5 & 1{,}5 & 1{,}8 & \mathbf{1{,}5} \\ 6{,}4 & 5{,}8 & \mathbf{1{,}4} & \mathbf{1} & \mathbf{0} & 3{,}2 \end{pmatrix} \end{array}$$

Sum of squares $= J^2(.) = 1.78$

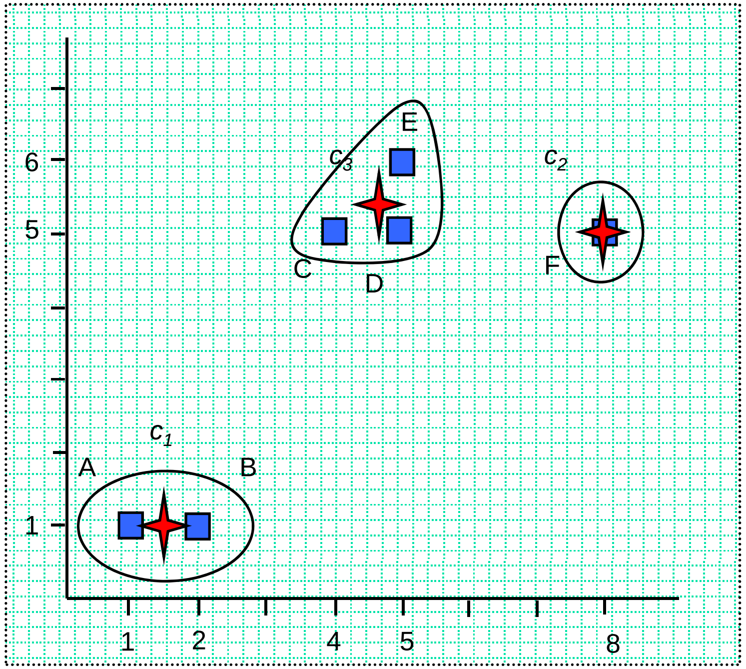
Optimizing prototypes:

$$c_1 = \left(\frac{1+2}{2}, \frac{1+1}{2}\right) = (1.5{,}1)$$

$$c_2 = (8{,}5)$$

$$c_3 = \left(\frac{4+5+5}{3}, \frac{5+5+6}{3}\right) = (4.7{,}5.3)$$

K-means: an example



Optimizing partitions:

Euclidean Distances

$$\begin{array}{c} \\ c_1 \\ c_2 \\ c_3 \end{array} \begin{array}{c} \begin{array}{cccccc} A & B & C & D & E & F \end{array} \\ \begin{pmatrix} \mathbf{0{,}5} & \mathbf{0{,}5} & 4{,}7 & 5{,}3 & 6{,}1 & 7{,}6 \\ 8{,}1 & 7{,}2 & 4 & 3 & 3{,}2 & \mathbf{0} \\ 5{,}7 & 5{,}1 & \mathbf{0{,}7} & \mathbf{0{,}5} & \mathbf{0{,}7} & 3{,}3 \end{pmatrix} \end{array}$$

Sum of squares = $J^3(.) = 0.31$

Assignment unchanged $\Rightarrow$
Terminate

K-means: Termination

- if neither Step 3. nor Step 2. changed J(.), the algorithm terminates, else
- Step 3. reduces J(.), because for a fixed assignment, the mean is the global minimizer of J(.).
- Step 2. reduces J(.), because for every $\mathbf{x_l}$ the contribution to the cost function either stays the same or gets lower.
- The fact that J(.) is reduced implies that no assignment is repeated during the run of the algorithm.
- Since there is a finite number of assignmens (how many?) *the k-means algorithm converges in a finite number of steps*, to a local minimum.

K-means: Notes

- Alternatively, $\mathbf{c_k}$ is initialised, and steps 2. and 3. are swapped
- For a fixed assignment, the mean is the global minimizer of $\frac{1}{|\mathcal{T}_k|}\sum_{\mathbf{x}\in\mathcal{T}_k}\mathbf{x} = \mathbf{c}^\star_\mathbf{k} = \arg\min_\mathbf{c}\sum_{\mathbf{x}\in\mathcal{T}_k}||\mathbf{x}-\mathbf{c_k}||_2^2$, (you should be able to prove this)
- the algorithm also solves the following minimization problem: $J(\mathcal{T}_1,\mathcal{T}_2,\ldots,\mathcal{T}_K) = \sum_{k=1}^{K}\sum_{\mathbf{x_i},\mathbf{x_j}\in\mathcal{T}_k}||\mathbf{x_i}-\mathbf{x_j}||_2^2$,
- The k-means algorithm does not reach a global minimum. This is easily proved by a counter-example.
- Efficiency. The complexity of Step 2. (assignment optimization) dominates, as for every observation the nearest prototype is sought. Trvially implemented, this requires $L\times K$ operations. Any idea for a speed-up?

K-means Generalization

In:	$\mathcal{T} = \{\mathbf{x_l}\}_{l=1}^{L}$,	the set of observations
	$d(.,.)$	"distance function" (may not be a metric)
Out:	$(\mathbf{c_k})_{k=1}^{K}$,	the set of cluster prototypes (etalons)
	$\{\mathcal{T}_k\}_{k=1}^{K}$	the clustering (partitioning) of the data

1. Initialize $\mathbf{c_k}$ (e.g. by assigning random $\mathbf{x_l}$ to $\mathbf{c_k}$)
2. Assignment optimization:
   $\mathcal{T}_k = \{\mathbf{x} \in \mathcal{T} : \forall j, d(\mathbf{x}, \mathbf{c_k}) \leq d(\mathbf{x}, \mathbf{c_j})\}$
3. Prototype optimization:
   $\mathbf{c_k} = \arg\min_{\mathbf{c}} \sum_{\mathbf{x} \in \mathcal{T}_k} d(\mathbf{x}, \mathbf{c})$
4. Terminate If $\mathcal{T}_k^{t+1} = \mathcal{T}_k^{t}, \forall k$; else go to 2

K-means Generalization: K-medians

In:	$\mathcal{T} = \{\mathbf{x_l}\}_{l=1}^{L}$,	the set of observations
	$d(.,.)$	$\|\|\mathbf{c} - \mathbf{x}\|\|_1$, ie. $d(.,.)$ is the L1-metric
Out:	$(\mathbf{c_k})_{k=1}^{K}$,	the set of cluster prototypes (etalons)
	$\{\mathcal{T}_k\}_{k=1}^{K}$	the clustering (partitioning) of the data

1. Initialize $\mathbf{c_k}$ (e.g. by assigning random $\mathbf{x_l}$ to $\mathbf{c_k}$)
2. Assignment optimization:
   $\mathcal{T}_k = \{\mathbf{x} \in \mathcal{T} : \forall j, d(\mathbf{x}, \mathbf{c_k}) \leq d(\mathbf{x}, \mathbf{c_j})\}$
3. Prototype optimization:
   $\mathbf{c_k} = \text{median}\{\mathcal{T}_k\}$
4. Terminate If $\mathcal{T}_k^{t+1} = \mathcal{T}_k^{t}, \forall k$; else go to 2

Median is the minimizer of the L1-norm in a cluster, ie.
median$\{\mathcal{T}_k\} = \mathbf{c}_\mathbf{k}^\star = \arg\min_\mathbf{c} \sum_{\mathbf{x}\in\mathcal{T}_k} ||\mathbf{x} - \mathbf{c_k}||_1$

K-means Generalization: Clustering Strings

In:	$\mathcal{T} = \{\mathbf{x_l}\}_{l=1}^{L}$,	observations $\mathbf{x_l}$ are strings
	$d(s_1, s_2)$	is the Levenshtein distance, ie. the number of edit operations to transform s_1 into s_2
Out:	$(\mathbf{c_k})_{k=1}^{K}$,	the set of cluster prototypes, $\mathbf{c_k}$ are strings
	$\{\mathcal{T}_k\}_{k=1}^{K}$	the clustering (partitioning) of the data

1. Initialize $\mathbf{c_k}$
2. Assignment optimization:
   $\mathcal{T}_k = \{\mathbf{x} \in \mathcal{T} : \forall j, d(\mathbf{x}, \mathbf{c_k}) \leq d(\mathbf{x}, \mathbf{c_j})\}$
3. Prototype optimization:
   $\mathbf{c_k} = \arg\min_{\mathbf{c}} \sum_{\mathbf{x} \in \mathcal{T}_k} d(\mathbf{x}, \mathbf{c})$
4. Terminate If $\mathcal{T}_k^{t+1} = \mathcal{T}_k^{t}, \forall k$; else go to 2

K-means Generalization: Clustering Strings: Notes

- the calculation of $d(.,.)$ might be non-trivial
- It might be very hard to minimize $\sum_{\mathbf{x}\in\mathcal{T}_k} d(\mathbf{x},\mathbf{c})$. over the space of all strings.
  The minimisation can be restricted to $\mathbf{c} \in \mathcal{T}$.
- Is the algorithm guaranteed to terminate, if Step 2. (Step 3.) is only improving J(.), not findind the minimimum (given fixed $\mathcal{T}$ or $\mathbf{c_k}$ respectively)?

macros_rpz.tex
sfmath.sty
cmpitemize.tex

Thank you for your attention.