

CS724: Topics in Algorithms

Partitional Clustering Algorithms

Prof. Dan A. Simovici



The k -means algorithm is one of the best known clustering algorithms and has been in existence for a long time. The k -means algorithm is considered by some authors to be among the top ten algorithms in data mining.

- The term “ k -means” was introduced by J. B. MacQueen.
- The best-known variant of the algorithm was proposed by S. Lloyd in 1957 as a technique for pulse-code modulation, published outside of Bell Labs 25 years later.
- E. W. Forgy published essentially the same method, known today as *Lloyd-Forgy algorithm*. Due to its simplicity and to its many implementations it is a very popular algorithm despite this requirement.



The k -means algorithm requires the specification of the number of clusters k as an input and computes a k -block partition of a finite set of points in \mathbb{R}^n such that the objects that belong to the same block have a high degree of similarity, and the objects that belong to distinct blocks are dissimilar.

- The k -means algorithm begins with a randomly chosen set of k points $\mathbf{c}_1, \dots, \mathbf{c}_k$ in \mathbb{R}^n called *centroids*.
- An initial partition of the set S of objects is computed by assigning each object \mathbf{u}_i to its closest centroid \mathbf{c}_j and adopting a rule for breaking ties when there are several centroids that are equally distanced from \mathbf{u}_i (e.g., assigning \mathbf{u}_i to the centroid with the lowest index).
- The algorithm alternates between assigning cluster membership for each object and computing the center of each cluster.



The k -means Lloyd-Forgy Algorithm

Data: the set of objects to be clustered $S = \{\mathbf{x}_1, \dots, \mathbf{x}_n\}$ and the number of clusters k ;

Result: a collection of k clusters;

generate a randomly chosen collection of k vectors $\mathbf{c}_1, \dots, \mathbf{c}_k$ in \mathbb{R}^n ;

assign each object \mathbf{x}_i to the closest centroid \mathbf{c}_j breaking ties in some arbitrary manner;

let $\pi = \{U_1, \dots, U_k\}$ be the partition defined by $\mathbf{c}_1, \dots, \mathbf{c}_k$;

Repeat{

 recompute $\mathbf{c}_1, \dots, \mathbf{c}_k$ as the centroids of the clusters U_1, \dots, U_k ;

ForEach ($\mathbf{x}_i \in X$) **do**

 {

if(\mathbf{x}_i is reassigned to a closer \mathbf{c}_j)

 then obj_reassigned++;

 }

}

until (obj_reassigned == 0)



Theorem

The function $sse(\pi)$ does not increase as the k -means through successive iterations of the Lloyd-Forgy Algorithm.



Proof

Let $S = \{\mathbf{x}_1, \dots, \mathbf{x}_n\}$ be the set of objects in \mathbb{R}^m to be clustered. Suppose that the partition $\pi = \{C_1, \dots, C_p, \dots, C_q, \dots, C_k\}$ was built at a certain stage of the algorithm and let $\pi' = \{C'_1, \dots, C'_p, \dots, C'_q, \dots, C'_k\}$ be the partition of X obtained by reassigning an object \mathbf{x}_r from C_p to C_q . We have:

$$C'_i = \begin{cases} C_i & \text{if } i \notin \{p, q\}, \\ C_p - \{\mathbf{x}\} & \text{if } i = p, \\ C_q \cup \{\mathbf{x}\} & \text{if } i = q. \end{cases}$$

This reassignment may take place only if $\|\mathbf{x}_r - \mathbf{c}_p\| \geq \|\mathbf{x}_r - \mathbf{c}_q\|$.



Since

$$\begin{aligned} & \sum \{\| \mathbf{x} - \mathbf{c}_p \|^2 \mid \mathbf{x} \in C_p\} + \sum \{\| \mathbf{x} - \mathbf{c}_q \|^2 \mid \mathbf{x} \in C_q\} \\ & \geq \sum \{\| \mathbf{x} - \mathbf{c}_p \|^2 \mid \mathbf{x} \in C_p - \{\mathbf{x}_r\}\} + \sum \{\| \mathbf{x} - \mathbf{c}_q \|^2 \mid \mathbf{x} \in C_q \cup \{\mathbf{x}_r\}\}, \end{aligned}$$

we have

$$\begin{aligned} sse(\pi) &= \sum_{j=1}^k \sum \{\| \mathbf{x} - \mathbf{c}_j \|^2 \mid \mathbf{x} \in C_j\} \\ &= \sum \left\{ \sum \{\| \mathbf{x} - \mathbf{c}_j \|^2 \mid \mathbf{x} \in C_j\} \mid j \in \{1, \dots, k\} - \{p, q\} \right\} \\ &\quad + \sum \{\| \mathbf{x} - \mathbf{c}_p \|^2 \mid \mathbf{x} \in C_p\} + \sum \{\| \mathbf{x} - \mathbf{c}_q \|^2 \mid \mathbf{x} \in C_q\} \\ &\geq \sum \left\{ \sum \{\| \mathbf{x} - \mathbf{c}_j \|^2 \mid \mathbf{x} \in C_j\} \mid j \in \{1, \dots, k\} - \{p, q\} \right\} \\ &\quad + \sum \{\| \mathbf{x} - \mathbf{c}_p \|^2 \mid \mathbf{x} \in C_p - \{\mathbf{x}_r\}\} \\ &\quad + \sum \{\| \mathbf{x} - \mathbf{c}_q \|^2 \mid \mathbf{x} \in C_q \cup \{\mathbf{x}_r\}\} = sse(\pi') \end{aligned}$$

Thus, $sse(\pi)$ does not increase when \mathbf{x}_r is reassigned.

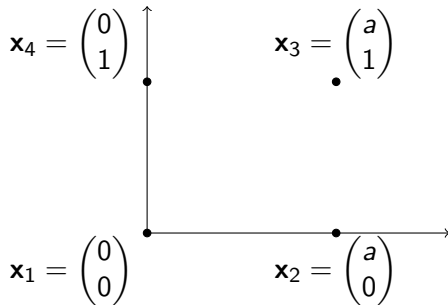


Example

Consider the set $S = \{\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3, \mathbf{x}_4\}$ in \mathbb{R}^n given by

$$\mathbf{x}_1 = \begin{pmatrix} 0 \\ 0 \end{pmatrix}, \mathbf{x}_2 = \begin{pmatrix} a \\ 0 \end{pmatrix}, \mathbf{x}_3 = \begin{pmatrix} a \\ 1 \end{pmatrix}, \mathbf{x}_4 = \begin{pmatrix} 0 \\ 1 \end{pmatrix}$$

shown next:



There are 7 distinct partitions having two blocks on a 4-element set, so there exist seven modalities to cluster these four objects:

Clusters		centroids		$sse(\pi)$
C_1	C_2	c_1	c_2	
$\{x_1\}$	$\{x_2, x_3, x_4\}$	x_1	$\begin{pmatrix} 2a/3 \\ 2/3 \end{pmatrix}$	$\frac{2}{3}(a^2 + 1)$
$\{x_2\}$	$\{x_1, x_3, x_4\}$	x_2	$\begin{pmatrix} a/3 \\ 2/3 \end{pmatrix}$	$\frac{2}{3}(a^2 + 1)$
$\{x_3\}$	$\{x_1, x_2, x_4\}$	x_3	$\begin{pmatrix} a/3 \\ 1/3 \end{pmatrix}$	$\frac{2}{3}(a^2 + 1)$
$\{x_4\}$	$\{x_1, x_2, x_3\}$	x_4	$\begin{pmatrix} 2a/3 \\ 1/3 \end{pmatrix}$	$\frac{2}{3}(a^2 + 1)$
$\{x_1, x_2\}$	$\{x_3, x_4\}$	$\begin{pmatrix} a/2 \\ 0 \end{pmatrix}$	$\begin{pmatrix} a/2 \\ 1 \end{pmatrix}$	a^2
$\{x_1, x_3\}$	$\{x_2, x_4\}$	$\begin{pmatrix} a/2 \\ 1/2 \end{pmatrix}$	$\begin{pmatrix} a/2 \\ 1/2 \end{pmatrix}$	$a^2 + 1$
$\{x_1, x_4\}$	$\{x_2, x_3\}$	$\begin{pmatrix} 0 \\ 1/2 \end{pmatrix}$	$\begin{pmatrix} a \\ 1/2 \end{pmatrix}$	1

It is easy to see that if $a \leq 1$, the least value of $sse(\pi)$ is a^2 ; for $a > 1$, the least value is 1.

If $a < 1$ and the centroids are $\begin{pmatrix} 0 \\ \frac{1}{2} \end{pmatrix}$ and $\begin{pmatrix} a \\ 1/2 \end{pmatrix}$, then the k -means algorithm will return the clustering $\{\{\mathbf{x}_1, \mathbf{x}_4\}, \{\mathbf{x}_2, \mathbf{x}_3\}\}$ whose $\text{sse}(\pi)$ value is 1 instead of the minimal value a^2 .

Similarly, if $a > 1$ and the centroids are $\begin{pmatrix} a/2 \\ 0 \end{pmatrix}$ and $\begin{pmatrix} a/2 \\ 1 \end{pmatrix}$, the algorithm returns the partition $\{\{\mathbf{x}_1, \mathbf{x}_2\}, \{\mathbf{x}_3, \mathbf{x}_4\}\}$ and the value of $\text{sse}(\pi)$ for this partition is a^2 instead of the least value of 1.

We may have gaps between the sum-of-squares value of the partition returned by the k -means algorithm and the minimum value of the objective function.



The next theorem shows a limitation of the k -means algorithm because this algorithm produces only clusters whose convex closures may intersect only at the points of S .

Theorem

Let $S = \{\mathbf{x}_1, \dots, \mathbf{x}_n\} \subseteq \mathbb{R}^m$ be a set of n vectors. If C_1, \dots, C_k is the set of clusters computed by the k -means algorithm in any step, then the convex closure of each cluster C_i , $\mathbf{K}_{\text{conv}}(C_i)$ is included in a polytope P_i that contains \mathbf{c}_i for $1 \leq i \leq k$.



Proof

Suppose that the centroids of the partition $\{C_1, \dots, C_k\}$ are $\mathbf{c}_1, \dots, \mathbf{c}_k$. Let $\mathbf{m}_{ij} = \frac{1}{2}(\mathbf{c}_i + \mathbf{c}_j)$ be the midpoint of the segment $\overline{\mathbf{c}_i \mathbf{c}_j}$ and let H_{ij} be the hyperplane $(\mathbf{c}_i - \mathbf{c}_j)'(\mathbf{x} - \mathbf{m}_{ij}) = 0$ that is the perpendicular bisector of the segment $\overline{\mathbf{c}_i \mathbf{c}_j}$. Equivalently,

$$H_{ij} = \{\mathbf{x} \in \mathbb{R}^m \mid (\mathbf{c}_i - \mathbf{c}_j)' \mathbf{x} = \frac{1}{2}(\mathbf{c}_i - \mathbf{c}_j)'(\mathbf{c}_i + \mathbf{c}_j)\}.$$

The halfspaces determined by H_{ij} are described by the inequalities:

$$\begin{aligned} H_{ij}^+ : (\mathbf{c}_i - \mathbf{c}_j)' \mathbf{x} &\leq \frac{1}{2}(\|\mathbf{c}_i\|_2^2 - \|\mathbf{c}_j\|_2^2) \\ H_{ij}^- : (\mathbf{c}_i - \mathbf{c}_j)' \mathbf{x} &\geq \frac{1}{2}(\|\mathbf{c}_i\|_2^2 - \|\mathbf{c}_j\|_2^2). \end{aligned}$$



Proof (cont'd)

It is easy to see that $\mathbf{c}_i \in H_{ij}^+$ and $\mathbf{c}_j \in H_{ij}^-$. Moreover, if $d_2(\mathbf{c}_i, \mathbf{x}) < d_2(\mathbf{c}_j, \mathbf{x})$, then $\mathbf{x} \in H_{ij}^+$, and if $d_2(\mathbf{c}_i, \mathbf{x}) > d_2(\mathbf{c}_j, \mathbf{x})$, then $\mathbf{x} \in H_{ij}^-$. Indeed, suppose that $d_2(\mathbf{c}_i, \mathbf{x}) < d_2(\mathbf{c}_j, \mathbf{x})$, which amounts to $\|\mathbf{c}_i - \mathbf{x}\|_2^2 < \|\mathbf{c}_j - \mathbf{x}\|_2^2$. This is equivalent to

$$(\mathbf{c}_i - \mathbf{x})'(\mathbf{c}_i - \mathbf{x}) < (\mathbf{c}_j - \mathbf{x})'(\mathbf{c}_j - \mathbf{x}).$$

The last inequality is equivalent to

$$\|\mathbf{c}_i\|_2^2 - 2\mathbf{c}_i'\mathbf{x} < \|\mathbf{c}_j\|_2^2 - 2\mathbf{c}_j'\mathbf{x},$$

which implies that $\mathbf{x} \in H_{ij}^+$.



Proof (cont'd)

In other words, \mathbf{x} is located in the same half-space as the closest centroid of the set $\{\mathbf{c}_i, \mathbf{c}_j\}$. Note also that if $d_2(\mathbf{c}_i, \mathbf{x}) = d_2(\mathbf{c}_j, \mathbf{x})$, then \mathbf{x} is located in $H_{ij}^+ \cap H_{ij}^- = H_{ij}$, that is, on the hyperplane shared by P_i and P_j .

Let P_i be the closed polytope defined by

$$P_i = \bigcap \{H_{ij}^+ \mid j \in \{1, \dots, k\} - \{i\}\}$$

Objects that are closer to \mathbf{c}_i than to any other centroid \mathbf{c}_j are located in the closed polytope P_i . Thus, $C_i \subseteq P_i$ and this implies $\mathbf{K}_{\text{conv}}(C_i) \subseteq P_i$.



Let $S = \{\mathbf{x}_1, \dots, \mathbf{x}_n\}$, where $S \subseteq \mathbb{R}^m$ be the set of objects to be clustered by the k -means algorithm, and let $C = \{\mathbf{c}_1, \dots, \mathbf{c}_k\}$ be a subset of \mathbb{R}^m . For $1 \leq i \leq k$ define the subset C_i of S as consisting of those members of S for which the closest point in C is \mathbf{c}_i (such that ties between distances $d(\mathbf{x}, \mathbf{c}_i)$ and $d(\mathbf{x}, \mathbf{c}_j)$ are broken arbitrarily). The collection $\{C_1, \dots, C_k\}$ is a partition π_C of S .



The k -means algorithm entails choosing the elements of C , to accomplish the minimization of the objective function

$$\text{sse}(\pi_C) = \sum_{i=1}^k \sum_{\mathbf{x}_j \in C_i} \|\mathbf{x}_j - \mathbf{c}_i\|^2 = \sum_{i=1}^k \sum_{j=1}^n z_{ij} \|\mathbf{x}_j - \mathbf{c}_i\|^2,$$

where

$$z_{ij} = \begin{cases} 1 & \text{if } \mathbf{x}_j \in C_i, \\ 0 & \text{otherwise} \end{cases}$$

are binary variables that indicate whether or not a data point \mathbf{x}_j belongs to C_i . The matrix $Z = (z_{ij})$ belongs to $\mathbb{R}^{k \times n}$.



We have:

$$\sum_{i=1}^k z_{ij} = 1 \text{ for } 1 \leq j \leq n, \text{ and}$$
$$\sum_{j=1}^n z_{ij} = |C_i| = n_i \text{ for } 1 \leq i \leq k.$$



Let $X = (\mathbf{x}_1 \ \mathbf{x}_2 \ \cdots \ \mathbf{x}_n) \in \mathbb{R}^{m \times n}$ be a matrix whose columns are the data points of the set S . The set C is represented by the matrix

$$M = (\mathbf{c}_1 \ \mathbf{c}_2 \ \cdots \ \mathbf{c}_k) \in \mathbb{R}^{m \times k}.$$

The Frobenius norm of the matrix X is given by:

$$\|X\|^2 = \sum_{j=1}^n \|\mathbf{x}_j\|^2 = \sum_{j=1}^n \mathbf{x}'_j \mathbf{x}_j = \sum_{j=1}^n (X'X)_{jj} = \text{trace}(X'X).$$

Since each \mathbf{x}_j belongs to exactly one set C_i , it follows that each column of Z contains exactly one entry of 1 and the remaining $k - 1$ entries of 0. This immediately implies that the rows of Z are pairwise orthogonal because

$$z_{ij}z_{i'j} = \begin{cases} 1 & \text{if } i = i' \\ 0 & \text{otherwise.} \end{cases}$$



In turn, this implies that $ZZ' \in \mathbb{R}^{k \times k}$ is a diagonal matrix where

$$(ZZ')_{ii'} = \sum_j (Z)_{ij}(Z')_{ji'} = \sum_j z_{ij}z_{i'j} = \begin{cases} n_i & \text{if } i = i', \\ 0 & \text{otherwise.} \end{cases} \quad (1)$$



Therefore,

$$(ZZ')^{-1} = \begin{pmatrix} \frac{1}{n_1} & 0 & 0 & \cdots & 0 \\ 0 & \frac{1}{n_2} & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \cdots & \vdots \\ 0 & 0 & 0 & \cdots & \frac{1}{n_k} \end{pmatrix}$$

Let $Y = Z'(ZZ')^{-1} \in \mathbb{R}^{n \times k}$. We have $y_{ji} = \frac{z_{ij}}{n_i}$. The columns of the matrix Y correspond to the clusters C_1, \dots, C_k and $\sum_{j=1}^n y_{ij} = 1$ for $1 \leq i \leq n$.



Since

$$\begin{aligned}y_{ji} &= \sum_{\ell=1}^k (Z')_{j\ell} ((ZZ')^{-1})_{\ell i} \\&= \sum_{\ell=1}^k z_{\ell j} ((ZZ')^{-1})_{\ell i} \\&= z_{ij} \frac{1}{n_i},\end{aligned}$$

it follows that $\sum_{j=1}^n y_{ji} = 1$. In other words, the components of each column \mathbf{y}_i of Y are non-negative numbers that sum up to 1, so they can be regarded as probability distributions.

The entropy of the i^{th} column \mathbf{y}_i is $H(\mathbf{y}_i) = -\sum_{j=1}^n y_{ji} \log y_{ji}$ for $1 \leq i \leq k$.



The next theorem shows that to minimize $\text{sse}(\pi_C)$ amounts to minimizing the norm of the matrix $X - MZ$, where $M \in \mathbb{R}^{m \times k}$ and $Z \in \mathbb{R}^{k \times n}$, that is, to find the best approximation of X as product MZ .

Theorem

(Baukhage's Factorization Theorem) *The following equality holds:*

$$\sum_{i=1}^k \sum_{j=1}^n z_{ij} \| \mathbf{x}_j - \mathbf{c}_i \|^2 = \| X - MZ \|^2 .$$



Proof

The left-hand member of the equality of the theorem can be written as

$$\begin{aligned} & \sum_{i=1}^k \sum_{j=1}^n z_{ij} \| \mathbf{x}_j - \mathbf{c}_i \|^2 \\ &= \sum_{i=1}^k \sum_{j=1}^n z_{ij} (\mathbf{x}_j - \mathbf{c}_i)' (\mathbf{x}_j - \mathbf{c}_i) \\ &= \sum_{i=1}^k \sum_{j=1}^n z_{ij} (\mathbf{x}_j' \mathbf{x}_j - 2 \mathbf{x}_j' \mathbf{c}_i + \mathbf{c}_i' \mathbf{c}_i) \\ &= T_1 - 2T_2 + T_3, \end{aligned}$$

where

$$T_1 = \sum_{i=1}^k \sum_{j=1}^n z_{ij} \mathbf{x}_j' \mathbf{x}_j, \quad T_2 = \sum_{i=1}^k \sum_{j=1}^n z_{ij} \mathbf{x}_j' \mathbf{c}_i,$$

$$T_3 = \sum_{i=1}^k \sum_{j=1}^n z_{ij} \mathbf{c}_i' \mathbf{c}_i.$$



Proof (cont'd)

We can further write

$$\begin{aligned} T_1 &= \sum_{i=1}^k \sum_{j=1}^n z_{ij} \mathbf{x}'_j \mathbf{x}_j = \sum_{i=1}^k \sum_{j=1}^n z_{ij} \|\mathbf{x}_j\|^2 = \sum_{j=1}^n \|\mathbf{x}_j\|^2 \sum_{i=1}^k z_{ij} \\ &= \sum_{j=1}^n \|\mathbf{x}_j\|^2 = \text{trace}(X'X). \end{aligned}$$



Proof (cont'd)

$$\begin{aligned}T_2 &= \sum_{i=1}^k \sum_{j=1}^n z_{ij} \mathbf{x}'_j \mathbf{c}_i \\&= \sum_{i=1}^k \sum_{j=1}^n z_{ij} \sum_{\ell=1}^m x_{\ell j} c_{\ell i} = \sum_{j=1}^n \sum_{\ell=1}^m x_{\ell j} \sum_{i=1}^k z_{ij} c_{\ell i} \\&= \sum_{j=1}^n \sum_{\ell=1}^m x_{\ell j} (MZ)_{\ell j} = \sum_{j=1}^n \sum_{\ell=1}^m (X')_{j\ell} (MZ)_{\ell j} \\&= \sum_{j=1}^n (X' M X)_{jj} = \text{trace}(X' M Z).\end{aligned}$$



Proof (cont'd)

$$\begin{aligned} T_3 &= \sum_{i=1}^k \sum_{j=1}^n z_{ij} \mathbf{c}_i' \mathbf{c}_i = \sum_{i=1}^k \sum_{j=1}^n z_{ij} \|\mathbf{c}_i\|^2 \\ &= \sum_{i=1}^k \|\mathbf{c}_i\|^2 \sum_{j=1}^n z_{ij} = \sum_{i=1}^k \|\mathbf{c}_i\|^2 n_i, \end{aligned}$$

where $n_i = |C_i|$.



Proof (cont'd)

For the right-hand member of the equality of the theorem we have

$$\begin{aligned}\|X - MZ\|^2 &= \text{trace}((X - MZ)'(X - MZ)) \\ &= \text{trace}(X'X) - 2\text{trace}(X'MZ) + \text{trace}(Z'M'MZ) \\ &= T_1 - 2T_2 + T_4,\end{aligned}$$

where $T_4 = \text{trace}(Z'M'MZ)$. Now, we have

$$\begin{aligned}T_4 &= \text{trace}(Z'M'MZ) \\ &= \text{trace}(M'MZZ') \\ &\quad (\text{due to the cyclic permutation invariance of the trace}) \\ &= \sum_{i=1}^k (M'MZZ')_{ii} = \sum_{i=1}^k \sum_{\ell=1}^m (M'M)_{i\ell} (ZZ')_{\ell i} \\ &= \sum_{i=1}^k (M'M)_{ii} (ZZ')_{ii} = \sum_{i=1}^k \|\mathbf{c}_i\|^2 n_i.\end{aligned}$$

Thus, $T_4 = T_3$, and this completes the argument.



The centroid matrix $M = (\mathbf{c}_1 \ \mathbf{c}_2 \ \cdots \ \mathbf{c}_k)$ that minimizes the objective function

$$F(M) = \|X - MZ\|^2$$

is obtained as

$$M = XZ'(ZZ')^{-1} = XY, \quad (2)$$

where Y is the matrix $Y = Z'(ZZ')^{-1} \in \mathbb{R}^{n \times k}$ previously introduced.



PAM stands for “partition around medoids”. The algorithm is intended to find a sequence of medoids that are centrally located in clusters. Objects that are tentatively defined as medoids are placed into a set S of *selected objects*. If O is the set of objects that the set $U = O - S$ is the set of *unselected objects*.



The algorithm has two phases:

- In the first phase, BUILD, a collection of k objects are selected for an initial set S .
- In the second phase, SWAP, one tries to improve the quality of the clustering by exchanging selected objects with unselected objects.

The goal of the algorithm is to minimize the average dissimilarity of objects to their closest selected object. Equivalently, we can minimize the sum of the dissimilarities between object and their closest selected object.



For each object p we maintain two numbers:

- D_p , the dissimilarity between p and the closest object in S , and
- E_p , the dissimilarity between p and the second closest object in S .

These numbers *must be updated every time when the sets S and U change*. Note that $D_j \leq E_j$ and that we have $p \in S$ if and only if $D_p = 0$.



The BUILD phase entails the following steps:

- Initialize S by adding to it an object for which the sum of the distances to all other objects is minimal.
- Consider an object $i \in U$ as a candidate for inclusion into the set of selected objects.
- For an object $j \in U$ compute D_j , the dissimilarity between j and the closest object in S .
- If $D_j > d(i, j)$ object j will contribute to the decision to select object i (because the quality of the clustering may benefit); let $C_{ji} = \max\{D_j - d(j, i), 0\}$.
- Compute the total gain obtained by adding i to S as $g_i = \sum_{j \in U} C_{ji}$.
- Choose that object i that maximizes g_i ; let $S := S \cup \{i\}$ and $U = U - \{i\}$.

These steps are performed until k objects have been selected.



The second phase, SWAP, attempts to improve the the set of selected objects and, therefore, to improve the quality of the clustering. This is done by considering all pairs $(i, h) \in S \times U$ and consists of computing the effect T_{ih} on the sum of dissimilarities between objects and the closest selected object caused by swapping i and h , that is, by transferring i from S to U and transferring h to from U to S . The computation of T_{ih} involves the computation of the contribution K_{jih} of each object $j \in U$ to the swap of i and h .




- K_{jih} is computed taking into account the following cases:
 - ▶ if $d(j, i) > D_j$ and $d(j, h) > D_j$ (which means that there is $\ell \in S$ such that $d(j, h) > d(j, \ell)$) then $K_{jih} = 0$;
 - ▶ if $d(j, i) = D_j$, then two cases occur:
 - ★ if $d(j, h) < E_j$, where E_j is the dissimilarity between j and the second closest selected object, then $K_{jih} = d(j, h) - d(j, i)$; note that K_{jih} can be either positive or negative.
 - ★ if $d(j, h) \geq E_j$, then $K_{jih} = E_j - D_j$; in this case $K_{jih} > 0$.
 - ▶ if $d(j, i) > D_j$ and $d(j, h) < D_j$, then

$$K_{jih} = d(j, h) - D_j.$$

- Compute the total result of the swap as

$$T_{ih} = \sum \{K_{jih} \mid j \in U\}.$$

- Select a pair $(i, h) \in S \times U$ that minimizes T_{ih} .
 - If $T_{ih} < 0$ the swap is carried out, D_p and E_p are updated for every object p , and we return at Step 1. If $\min T_{ih} > 0$, the value of the objective cannot be decreased and the algorithm halts. This happens when all values of T_{ih} are positive (the halting condition of the algorithm).
- 



Contribution K_{jih} of object j to the swap of $i \in S$ with $h \in U$.

$$\begin{array}{c} \text{---|---|---} \\ D_j \quad d(j, i) \quad d(j, h) \end{array} \quad K_{jih} = 0$$

(a)

$$\begin{array}{c} d(j, i) = D_j \quad d(j, h) \\ \text{---|---} \\ D_j \quad E_j \end{array} \quad K_{jih} = d(j, h) - d(j, i)$$

(b)-(i)

$$\begin{array}{c} d(j, i) = D_j \quad d(j, h) \\ \text{---|---} \\ D_j \quad E_j \end{array} \quad K_{jih} = E_j - D_j$$

(b)-(ii)

$$\begin{array}{c} d(j, h) \quad d(j, i) \\ \text{---|---} \\ D_j \end{array} \quad K_{jih} = d(j, h) - D_j$$



The function `kmeans` of **R** performs the k -means clustering on a data matrix. The standard usage of this function is

```
kmeans(D, centers, iter.max = 10, nstart = 1, algorithm, trace)
```



Its arguments are:

- `D`: a matrix of data, or an object that can be coerced to such a matrix (such as a numeric vector or a data frame with all numeric columns);
- `centers`: either the number of clusters, say k , or a set of initial (distinct) cluster centroids. In the first case, a random set of (distinct) rows in D is chosen as the set of initial centroids;
- `iter.max`: the maximum number of iterations allowed;
- `nstart`: if `centers` is a number, this parameter indicates the number of random sets;
- `algorithm`: a string that indicates the variant of the algorithm, as discussed previously;



`kmeans` returns an object which the following components:

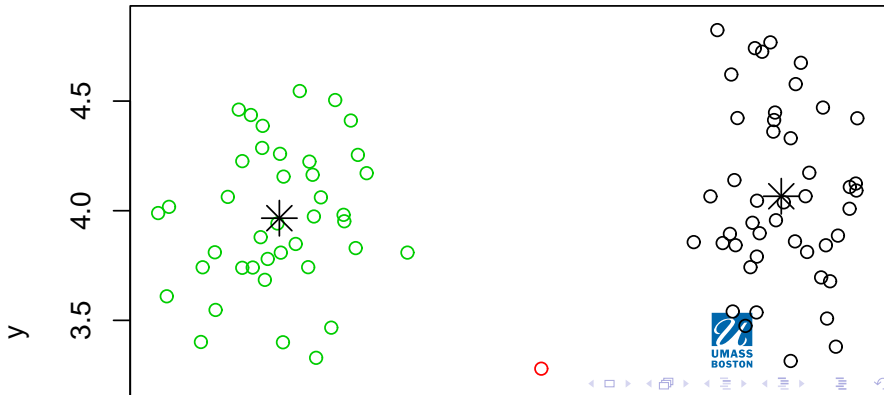
- `cluster`: a vector of integers (from 1 to k) indicating the cluster to which each point is allocated;
- `centers`: a matrix of cluster centres;
- `totss`: the total sum of squares;
- `withinss`: vector of within-cluster sum of squares, one component per cluster;
- `tot.withinss`: total within-cluster sum of squares, that is, the `sum(withinss)`;
- `betweenss`: the between-cluster sum of squares;
- `size`: the number of points in each cluster;
- `iter`: the number of (outer) iterations;
- `ifault`: an integer indicator of a possible algorithm problem.



The kmeans cluster produces the object sp in

```
sp <- kmeans(D,3)
```

that is plotted using `plot(D,col=sp$cluster)`.



The function `pam`, a component of the package `clust` implements the algorithm discussed above.

To apply the algorithm to the data set `D` previously computed we write:

```
pamx <- pam(D, 3)
```



The function `pam`, a component of the package `clust` implements the algorithm discussed above.

To apply the algorithm to the data set `D` previously computed we write:

```
pamx <- pam(D, 3)
```

