Applications of Mathematics

Kristian Sabo; Rudolf Scitovski

Interpretation and optimization of the k-means algorithm

Applications of Mathematics, Vol. 59 (2014), No. 4, 391--406

Persistent URL: http://dml.cz/dmlcz/143871

INTERPRETATION AND OPTIMIZATION OF THE k-MEANS ALGORITHM

Kristian Sabo, Rudolf Scitovski, Osijek

(Received October 9, 2012)

Abstract. The paper gives a new interpretation and a possible optimization of the well-known k-means algorithm for searching for a locally optimal partition of the set $\mathcal{A} = \{a_i \in \mathbb{R}^n\colon\ i = 1, \ldots, m\}$ which consists of k disjoint nonempty subsets $\pi_1, \ldots, \pi_k$, $1 \leqslant k \leqslant m$. For this purpose, a new divided k-means algorithm was constructed as a limit case of the known smoothed k-means algorithm. It is shown that the algorithm constructed in this way coincides with the k-means algorithm if during the iterative procedure no data points appear in the Voronoi diagram. If in the partition obtained by applying the divided k-means algorithm there are data points lying in the Voronoi diagram, it is shown that the obtained result can be improved further.

Keywords: clustering; data mining; k-means; Voronoi diagram

MSC 2010: 68T10, 62H30, 91C20, 90C26

1. Introduction

Clustering or grouping a data set into conceptually meaningful clusters has been a well-studied problem in recent literature, and it has practical importance in a wide variety of applications [7], [12], [13], [14], [18], [20].

Let $I = \{1, \ldots, m\}$ and $J = \{1, \ldots, k\}$, $1 \leqslant k \leqslant m$, be sets of indices. A partition of the set $\mathcal{A} = \{a_i \in \mathbb{R}^n\colon\ i = 1, \ldots, m\}$ into k disjoint subsets $\pi_1, \ldots, \pi_k$, $1 \leqslant k \leqslant m$, such that

$$\bigcup_{j=1}^{k} \pi_j = \mathcal{A}, \quad \pi_r \cap \pi_s = \emptyset, \ \ r \neq s, \quad |\pi_j| \geqslant 1, \ \ j = 1, \ldots, k, \tag{1.1}$$

will be denoted by $\Pi(\mathcal{A}) = \{\pi_1, \ldots, \pi_k\}$, and the elements $\pi_1, \ldots, \pi_k$ of such partition are called *clusters in* $\mathbb{R}^n$.

If $d\colon\ \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}_+$, $\mathbb{R}_+ = [0, \infty\rangle$, is a distance-like function (see e.g. [12], [14], [23]), then with each cluster $\pi_j \in \Pi$ we can associate its center c_j defined by

$$c_j = c(\pi_j) := \operatorname*{argmin}_{x \in \operatorname{conv}(\pi_j)} \sum_{a_i \in \pi_j} d(x, a_i), \tag{1.2}$$

where $\operatorname{conv}(\pi_j)$ is the convex hull of the set π_j.

If we introduce an objective function $\mathcal{F}\colon\ \mathcal{P}(\mathcal{A}; k) \to \mathbb{R}_+$ on the set of all partitions $\mathcal{P}(\mathcal{A}; k)$ of the set $\mathcal{A}$, we can define the quality of a partition and search for a *k-globally optimal partition by solving the following optimization problem:*

$$\operatorname*{argmin}_{\Pi \in \mathcal{P}(\mathcal{A}; k)} \mathcal{F}(\Pi), \quad \mathcal{F}(\Pi) = \sum_{j=1}^{k} \sum_{a_i \in \pi_j} d(c_j, a_i), \tag{1.3}$$

where $c_j = c(\pi_j)$ *is given by* (1.2).

Conversely, for a given set of different assignment points $z_1, \ldots, z_k \in \mathbb{R}^n$, applying the minimal distance condition, we can define the partition $\Pi = \{\pi_1, \ldots, \pi_k\}$ of the set $\mathcal{A}$ in the following way:

$$\pi_j = \{a \in \mathcal{A}\colon\ d(z_j, a) \leqslant d(z_s, a),\ \forall\, s \in J\}, \quad j \in J, \tag{1.4}$$

where one has to take care that every element of the set $\mathcal{A}$ occurs in one and only one cluster. Therefore, the problem of finding an optimal partition of the set $\mathcal{A}$ can be reduced to the optimization problem

$$\operatorname*{argmin}_{z_1, \ldots, z_k \in \mathbb{R}^n} F(z_1, \ldots, z_k), \tag{1.5}$$

$$F(z_1, \ldots, z_k) = \sum_{i=1}^{m} \min_{1 \leqslant j \leqslant k} d(z_j, a_i) = \sum_{i=1}^{m} \sum_{j=1}^{k} w_i^{(j)} d(z_j, a_i),$$

where $F\colon\ \mathbb{R}^{kn} \to \mathbb{R}_+$, and

$$w_i^{(j)} = \begin{cases} 1, & a_i \in \pi(z_j), \\ 0, & a_i \notin \pi(z_j), \end{cases} \tag{1.6}$$

where $j \in J$, and for all $i \in I$ we have

$$\sum_{j=1}^{k} w_i^{(j)} = 1. \tag{1.7}$$

The solutions of (1.3) and (1.5) coincide (see e.g. [1], [21]). A global optimization problem (1.5) can also be found in literature as a *center-based clustering problem* or *k-means/k-median problem* [9], [15], [19], [23].

Thereby, the objective function F is a symmetric Lipschitz continuous function which can have a large number of independent variables (the number of clusters in the partition multiplied by the dimension of data points $(k \cdot n)$), it has to be neither convex nor differentiable, and generally it may have at least $k!$ local and global minima [8]. Therefore, this becomes a complex global optimization problem [6], [18].

The paper is organized as follows. In the next section, two well-known algorithms for searching for the locally optimal partition, i.e., the k-means algorithm and the smoothed k-means algorithm are briefly described and a new divided k-means algorithm is proposed. In Section 3, some properties of the new algorithm and its connection with the k-means algorithm is shown. In Section 4, some numerical experiments are given that point out the advantage of the proposed algorithm. Finally, some conclusions are given in Section 5.

2. Algorithms for searching for the locally optimal partition

In the sequel, a special and well-known *least square distance-like function* given by $d(x, y) = \|x - y\|_2^2$, $x, y \in \mathbb{R}^n$ will be used as a distance-like function.

2.1. k-means algorithm. There are various notation variants of this well-known algorithm (see e.g. [12], [15], [17]). For further usage in this paper, the algorithm will be written in the following way.

Algorithm 1 (k-means algorithm)**.**

Step 0: Input $1 \leqslant k \leqslant m$; $I = \{1, \ldots, m\}$; $J = \{1, \ldots, k\}$; $\mathcal{A} = \{a_i \in \mathbb{R}^n \colon i \in I\}$. Choose mutually different assignment points $z_1, \ldots, z_k \in \operatorname{conv}(\mathcal{A})$.

Step 1: (Assignment step) Define clusters

$$\pi(z_j) = \{a_i \in \mathcal{A} \colon d(z_j, a_i) \leqslant d(z_s, a_i),\ \forall\, s \in J\}, \quad j \in J,$$

where one has to take care that every element of the set $\mathcal{A}$ occurs in one and only one cluster. Define weights $w_i^{(j)}$ according to (1.6).

Calculate $F_0 = \sum\limits_{j=1}^{k} \Big(\sum\limits_{i=1}^{m} w_i^{(j)} d(z_j, a_i) \Big)$.

Step 2: (Update step) Determine

$$c_j = \operatorname*{argmin}_{x \in \mathbb{R}^n} \sum_{i=1}^{m} w_i^{(j)} d(x, a_i) = \frac{1}{\sum_{l=1}^{m} w_l^{(j)}} \sum_{i=1}^{m} w_i^{(j)} a_i, \quad j \in J;$$

$$\pi(c_j) = \{a_i \in \mathcal{A} \colon d(c_j, a) \leqslant d(c_s, a),\ \forall\, s \in J\}, \quad j \in J.$$

Define new weights

$$w_i^{(j)} = \begin{cases} 1, & a_i \in \pi(c_j), \\ 0, & a_i \notin \pi(c_j), \end{cases}$$

where $j \in J$, such that

$$\sum_{j=1}^{k} w_i^{(j)} = 1.$$

Calculate $F_1 = \sum_{j=1}^{k} \Big(\sum_{i=1}^{m} w_i^{(j)} d(c_j, a_i) \Big)$.

Step 3: If $F_1 < F_0$, set $z_j = c_j$ for all $j \in J$ and $F_0 = F_1$ and go to Step 1.
Else set $c_j^\star = c_j$ for all $j \in J$ and STOP.

Points $z_1, \ldots, z_k$ from Step 1 and points $c_1, \ldots, c_k$ from Step 2 are called *assignment points* and *centroids* of the clusters, respectively. Centroids in Step 2 become assignment points on the basis of which we define new clusters.

Algorithm 1 is finite and in every step it reduces the value of the objective function [12], [21]. Centroids $(c_1^\star, \ldots, c_k^\star)$ obtained by applying Algorithm 1 are called *locally optimal centroids*, and the corresponding partition $\{\pi_1^\star, \ldots, \pi_k^\star\}$ is called a *locally optimal partition*.

In addition to that, it may happen that one of the clusters becomes an empty set [12]. In relation to that, [22] gives a sufficient condition under which the functional (1.5) attains its local minimum at the point $(c_1^\star, \ldots, c_k^\star)$. A partition determined by this point is called a *stable partition* [12], [23]. Also, in accordance with [22], a stable partition does not contain empty clusters.

If we have a good initial approximation, the k-means algorithm can provide an acceptable solution [24]. A good initial approximation can be obtained by some of the genetic algorithms, such as the FIREFLY heuristic algorithm [26], or by using some of the global optimization methods, such as DIRECT [5], [11]. The symmetry property of the objective function F was a motive for developing a very efficient special version of the DIRECT algorithm for symmetric functions in [8]. In case we do not have a good initial approximation, the k-means algorithm should be restarted with various random initializations, as proposed by [15]. A very good approximate globally optimal partition can be obtained by using some of the incremental algorithms as different modifications of the global k-means method [2], [3], [16].

2.2. Smoothed k-means algorithm (SMOKA). The smoothed k-means algorithm (SMOKA) has appeared relatively recently in literature as a natural generalization of the well-known Weiszfeld algorithm for the Fermat-Weber location problem

(see e.g. [10]). In the sequel, we will briefly describe this algorithm and give its most important properties. Consider the optimization problem

$$\min_{z_1,\dots,z_k\in\mathbb{R}^n} F(z_1,\dots,z_k), \quad F(z_1,\dots,z_k)=\sum_{i=1}^{m}\min_{1\leqslant j\leqslant k} d(z_j,a_i). \tag{2.1}$$

Since every vector $r=(r_1,\dots,r_n)\in\mathbb{R}^n$ satisfies (see e.g. [12])

$$\max_{1\leqslant j\leqslant k} r_j=\lim_{\varepsilon\to 0+}\varepsilon\ln\sum_{j=1}^{n}\exp\Big(\frac{r_j}{\varepsilon}\Big),$$

and $\min\limits_{1\leqslant j\leqslant k} r_j=-\max\limits_{1\leqslant j\leqslant k}(-r_j)$, the functional (2.1) can be approximated by

$$F_\varepsilon(z_1,\dots,z_k)=-\varepsilon\sum_{i=1}^{m}\ln\sum_{j=1}^{k}\mathrm{e}^{-d(z_j,a_i)/\varepsilon}, \tag{2.2}$$

and instead of solving the non-differentiable optimization problem (2.1), we can solve the following differentiable optimization problem (see [12], [23])

$$\min_{z_1,\dots,z_k\in\mathbb{R}^n} F_\varepsilon(z_1,\dots,z_k). \tag{2.3}$$

Let us note that $\hat\theta:=(\hat c_1,\dots,\hat c_k)\in\mathbb{R}^{nk}$ is a stationary point of the functional F_ε if and only if for every $j\in J$ we have

$$\hat c_j=\frac{1}{\sum_{l=1}^{m}\omega_l^{(j)}(\varepsilon)}\sum_{i=1}^{m}\omega_i^{(j)}(\varepsilon)a_i,\ \text{ where }\ \omega_i^{(j)}(\varepsilon)=\frac{\mathrm{e}^{-d(\hat c_j,a_i)/\varepsilon}}{\sum_{s=1}^{k}\mathrm{e}^{-d(\hat c_s,a_i)/\varepsilon}},\ \ i\in I. \tag{2.4}$$

Therefore, the stationary point $\hat\theta:=(\hat c_1,\dots,\hat c_k)\in\mathbb{R}^{nk}$ of the functional F_ε can be searched for by the iterative procedure for $t=0,1,\dots$

$$c_j^{(t+1)}=\frac{1}{\sum_{l=1}^{m}\omega_l^{(j)}(\varepsilon)}\sum_{i=1}^{m}\omega_i^{(j)}(\varepsilon)a_i,\quad \text{where}\quad \omega_i^{(j)}(\varepsilon)=\frac{\mathrm{e}^{-d(c_j^{(t)},a_i)/\varepsilon}}{\sum_{s=1}^{k}\mathrm{e}^{-d(c_s^{(t)},a_i)/\varepsilon}}, \tag{2.5}$$

whereby $\theta^{(0)}=(c_1^{(0)},\dots,c_k^{(0)})\in\mathbb{R}^{nk}$ is some initial approximation—initial assignment points. In every step, the iterative procedure (2.5) determines the next approximation of the j-th component of the vectors of centers θ as a weighted arithmetic mean of data $a_i\in\mathcal{A}$ with weights $\omega_i^{(j)}(\varepsilon)$.

From the construction it can be seen that this algorithm is numerically very demanding and practically it cannot compete with the k-means algorithm.

The properties of the iterative procedure (2.5) are given in [12], [23], and sufficient conditions under which the functional F_ε at the stationary point attains its local minimum are given in [22]. Specially, in [19], this problem is considered for an l_1-metric function.

SMOKA also appears to be a special case of fuzzy C-means where each data point has a degree of belonging to clusters, rather than belonging completely to just one cluster [25].

2.3. Divided k-means algorithm (DKM)**.** In this section, we will analyze properties of weighted functions $\varepsilon \mapsto \omega_i^{(j)}(\varepsilon)$, $i \in I$, $j \in J$, used in the iterative procedure (2.5) and define a new algorithm for searching for the locally optimal partition.

Suppose we are given a set of data $\mathcal{A}$ and a set of mutually different assignment points $z_1, \ldots, z_k$. As already mentioned in Section 2.2, the SMOKA algorithm is determined by the iterative procedure (2.5), which in every step of the given assignment points defines new centers as weighted arithmetical means of data $a_i \in \mathcal{A}$ with weights $\omega_i^{(j)}(\varepsilon)$ given by

$$\omega_i^{(j)}(\varepsilon) = \frac{\mathrm{e}^{-d(z_j,a_i)/\varepsilon}}{\sum_{s=1}^{k} \mathrm{e}^{-d(z_s,a_i)/\varepsilon}}, \quad i \in I,\ j \in J. \tag{2.6}$$

Note that weights (2.6) satisfy the following simple conditions:

$$0 < \omega_i^{(j)}(\varepsilon) < 1, \tag{2.7}$$

$$\sum_{j=1}^{k} \omega_i^{(j)}(\varepsilon) = 1. \tag{2.8}$$

Specially, if $k = |J| = 1$, then $\omega_i^{(1)}(\varepsilon) = 1$ for every $i \in I$.

Furthermore, for every $a_i \in \mathcal{A}$ define a set of indices of the nearest assignment points

$$U_i = \{j \in J\colon\ d(z_j, a_i) \leqslant d(z_s, a_i),\ \forall\, s \in J\}. \tag{2.9}$$

Note that the set U_i is unempty, and that it can be a single member set (if $a_i \notin V[z_1, \ldots, z_k]$) or a multi-member set (if $a_i \in V[z_1, \ldots, z_k]$). If for every $a_i \in \mathcal{A}$ the set U_i is a single member set, then the corresponding partition $\Pi = \{\pi(z_1), \ldots, \pi(z_k)\}$ is said to be a *well-separated partition*, i.e. the partition Π is said to be a well-separated partition if and only if the following holds:

$$(\forall\, a_i \in \mathcal{A})(\exists\, j \in J)\ d(z_j, a_i) < d(z_s, a_i), \quad \forall\, s \in J \setminus \{j\}. \tag{2.10}$$

R e m a r k 2.1. An element $a_i \in \mathcal{A}$ occurs rarely on the Voronoi diagram, but an element $a_i \in \mathcal{A}$ may very often occur in the immediate neighborhood of the Voronoi diagram. The following algorithm for each $a_i \in \mathcal{A}$ determines the set U_i defined by (2.9) with accuracy of up to the machine epsilon ε_M (see e.g. [4]).

1. $U_i = \emptyset$; $d_{\min} := \min\limits_{s \in J} d(z_s, a_i)$;
2. for $j = 1, \ldots, k$ do
3. $\quad \Delta_j := d(z_j, a_i) - d_{\min}$;
4. $\quad$ If $\Delta_j < \varphi(\varepsilon_M)$, $U_i = U_i \cup \{j\}$
5. end for

where $\varphi(\varepsilon_M)$ is a calculation error due to machine accuracy.

The following lemma gives behavior of weights (2.6) depending on sets U_i.

Lemma 2.1. *Let $\mathcal{A} = \{a_i \colon i \in I\}$ be a set of data points, and $z_1, \ldots, z_k$, $k > 1$, a set of assignment points. Let $U_i, |U_i| = \mu_i \leqslant k$, be the set of indices associated with the element $a_i \in \mathcal{A}$ by* (2.9).

(i) *If $\mu_i < k$, then the functions given by* (2.6) *for every $i \in I$ satisfy*

$$v_i^{(j)} := \lim_{\varepsilon \to 0^+} \omega_i^{(j)}(\varepsilon) = \begin{cases} \dfrac{1}{\mu_i}, & \text{if } j \in U_i \\ 0, & \text{if } j \in J \setminus U_i, \end{cases} \tag{2.11}$$

$$\sum_{j \in U_i} v_i^{(j)} = \sum_{j \in U_i} \frac{1}{\mu_i} = 1, \tag{2.12}$$

whereby the functions $\varepsilon \mapsto \omega_i^{(r)}(\varepsilon)$, $r \in U_i$, are strictly monotonically decreasing on the interval $\langle 0, \infty \rangle$;

(ii) *If $\mu_i = k$, then for every $j \in J$ and every $\varepsilon \in \langle 0, \infty \rangle$ the functions $\varepsilon \mapsto \omega_i^{(j)}(\varepsilon) = 1/k$ are constants.*

P r o o f. (i) Let us choose $r \in U_i$ and denote the function $\varepsilon \mapsto \omega_i^{(j)}(\varepsilon)$ given by (2.6) as

$$\omega_i^{(j)}(\varepsilon) = \begin{cases} \dfrac{1}{\mu_i + \sum_{s \in J \setminus U_i} \mathrm{e}^{-(1/\varepsilon)(d(z_s,a_i) - d(z_r,a_i))}} & \text{if } j \in U_i, \\ \dfrac{\mathrm{e}^{-(1/\varepsilon)(d(z_j,a_i) - d(z_r,a_i))}}{\mu_i + \sum_{s \in J \setminus U_i} \mathrm{e}^{-(1/\varepsilon)(d(z_s,a_i) - d(z_r,a_i))}} & \text{if } j \in J \setminus U_i. \end{cases} \tag{2.13}$$

Since $1 < k < \mu_i$, it holds that $J \setminus U_i \neq \emptyset$. Hence, in accordance with definition (2.9) of the set U_i, for every $r \in U_i$ and every $s \in J \setminus U_i$ we have that $d(z_s, a_i) > d(z_r, a_i)$. Therefore, (2.11) and (2.12) follow directly from (2.13).

Further, for $r \in U_i$, the derivative of the function $\varepsilon \mapsto \omega_i^{(r)}(\varepsilon)$ given by (2.6) can be written as

$$\frac{\mathrm{d}}{\mathrm{d}\varepsilon}(\omega_i^{(r)}(\varepsilon)) = -\frac{1}{\varepsilon^2}\mathrm{e}^{-d(z_r,a_i)/\varepsilon}\Big(\sum_{s=1}^{k}\mathrm{e}^{-d(z_s,a_i)/\varepsilon}\Big)^{-2} \times \sum_{s=1}^{k}\mathrm{e}^{-d(z_s,a_i)/\varepsilon}(d(z_s,a_i)-d(z_r,a_i)). \tag{2.14}$$

Since $d(z_s,a_i) > d(z_r,a_i)$ holds for every $s \in J \setminus U_i$, from (2.14) it follows that $\frac{\mathrm{d}}{\mathrm{d}\varepsilon}(\omega_i^{(r)}(\varepsilon)) < 0$. Hence, functions $\varepsilon \mapsto \omega_i^{(r)}(\varepsilon)$ for all $r \in U_i$ are strictly monotonically decreasing on the interval $\langle 0,\infty\rangle$.

(ii) If $\mu_i = k$, then the data a_i is situated on the border of all clusters so that $\omega_i^{(r)}(\varepsilon) = 1/k$ holds for every $\varepsilon \in \langle 0,\infty\rangle$, from where the assertion follows. □

Note that the weights $v_i^{(j)}$ defined by (2.11) in this way retain the property (1.7), whereas the property $w_i^{(j)} \in \{0,1\}$, $i \in I$, $j \in J$, relaxes into a more general form $v_i^{(j)} \in \{0,1,1/2,\dots,1/k\} \subset [0,1]$, $i \in I$, $j \in J$.

By modifying the k-means algorithm (Algorithm 1) such that the weights $w_i^{(j)}$ are redefined according to (2.11), we obtain a new algorithm that will be called the *divided k-means algorithm* (DKM). In this way, the effect will be such as if the data $a_i \in \mathcal{A}$ that appeared in the Voronoi diagram $V[z_1,\dots,z_k]$ was evenly distributed to all clusters whose borders it is located on. If in every step of the k-means algorithm no data $a_i \in \mathcal{A}$ appear in the Voronoi diagram, then the DKM algorithm becomes a common k-means algorithm. Similarly to the k-means algorithm, such algorithm is finite and in every step it reduces the objective function value.

Algorithm 2 (Divided k-means algorithm—DKM)**.**

Step 0: Input $1 \leqslant k \leqslant m$; $I = \{1,\dots,m\}$; $J = \{1,\dots,k\}$; $\mathcal{A} = \{a_i \in \mathbb{R}^n\colon\ i \in I\}$.
Choose mutually different assignment points $z_1,\dots,z_k \in \mathrm{conv}(\mathcal{A})$.

Step 1: (Assignment step) For each $j \in J$ define clusters $\pi(z_j) = \{a_i \in \mathcal{A}\colon d(z_j,a_i) \leqslant d(z_s,a_i)$ for all $s \in J\}$.
Determine sets U_i, $i \in I$ and according to (2.11) corresponding new weights $v_i^{(j)}$.
Calculate $F_0 = \sum_{j=1}^{k}\Big(\sum_{i=1}^{m} v_i^{(j)} d(z_j,a_i)\Big)$.

Step 2: (Update step) Determine centers of clusters

$$c_j = \operatorname*{argmin}_{x\in\mathbb{R}^n}\sum_{i=1}^{m} v_i^{(j)} d(x,a) = \frac{1}{\sum_{l=1}^{m} v_l^{(j)}}\sum_{i=1}^{m} v_i^{(j)} a_i, \quad j \in J.$$

Define new clusters $\pi(c_j) = \{a_i \in \mathcal{A}\colon\ d(c_j,a_i) \leqslant d(c_s,a_i)$ for all $s \in J\}$, $j \in J$.

Determine sets U_i, $i \in I$ and according to (2.11) corresponding new weights $v_i^{(j)}$

Calculate $F_1 = \sum_{j=1}^{k} \Big(\sum_{i=1}^{m} v_i^{(j)} d(c_j, a_i) \Big)$;

Step 3: If $F_1 < F_0$, set $z_j = c_j$ for all $j \in J$ and $F_0 = F_1$ and go to Step 1.
Else set $c_j^\star = c_j$ for all $j \in J$ and STOP.

It is obvious that $\sum_{j=1}^{k} v_i^{(j)} = 1$ holds for every $i \in I$ in Step 1 and Step 2.

In contrast to the common k-means algorithm, by stopping the DKM algorithm it is possible to get a partition such that some elements lie on the border between two clusters, i.e. in the Voronoi diagram.

E x a m p l e 2.1. Given are the data points $\mathcal{A} = \{a_1, \ldots, a_8\} \subset \mathbb{R}^2$, where

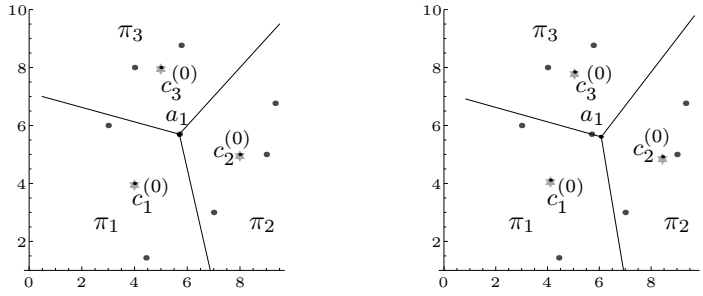
$$\mathcal{A} = \big\{ \big(\tfrac{57}{10}, \tfrac{57}{10}\big), (3,6), \big(\tfrac{133}{30}, \tfrac{43}{30}\big), (7,3), (9,5), \big(\tfrac{280}{30}, \tfrac{203}{30}\big), (4,8), \big(\tfrac{173}{30}, \tfrac{263}{30}\big) \big\}$$

and initial assignment points (see Figure 1a),

$$c_1^{(0)} = (4,4), \quad c_2^{(0)} = (8,5), \quad c_3^{(0)} = (5,8).$$

According to (2.11), we associate the weights $v_i^{(1)}, v_i^{(2)}, v_i^{(3)}$ with each data point $a_i \in \mathcal{A}$ in the following way (see Figure 1a)

j \ i	1	2	3	4	5	6	7	8
1	1/3	1	1	0	0	0	0	0
2	1/3	0	0	1	1	1	0	0
3	1/3	0	0	0	0	0	1	1



(a) Initial approximation (b) Solution

Figure 1. Divided k-means algorithm

After two iterations of the DKM algorithm we obtain locally optimal centroids (see Figure 1b). The corresponding clusters will be denoted as pairs of elements of the

set $\mathcal{A}$ with the corresponding weights

$$\pi_1 = \big\{\big(a_1, \tfrac{1}{2}\big), (a_2, 1), (a_3, 1)\big\}, \quad \pi_2 = \{(a_4, 1), (a_5, 1), (a_6, 1)\},$$
$$\pi_3 = \big\{\big(a_1, \tfrac{1}{2}\big), (a_7, 1), (a_8, 1)\big\}.$$

Note that the element a_1 takes place in the Voronoi diagram of an optimal partition. The flow of the iterative procedure is shown in Table 1.

	$c_1^{(t)}$	$c_2^{(t)}$	$c_3^{(t)}$	$F(c_1^{(t)}, c_2^{(t)}, c_3^{(t)})$
$t = 0$	(4,4)	(8,5)	(5,8)	30.630
$t = 1$	(4,4)	(8.17,5)	(5,8)	30.534
$t = 2$	(4.113, 4.113)	(8.444, 4.922)	(5.047, 7.847)	29.891

Table 1. Iterative procedure

3. Properties of the DKM algorithm and its connection with the k-means algorithm

Suppose that by applying the DKM algorithm we obtained centroids $c_1^\star, \ldots, c_k^\star$, whereby there exists an element $a_{i_0} \in \mathcal{A}$ lying in the Voronoi diagram $V[c_1^\star, \ldots, c_k^\star]$, such as e.g. in Example 2.1. Let us show that then the objective function value can be reduced so that by using the minimal distance principle we define a partition $\widehat{\Pi} = \{\hat{\pi}_1, \ldots, \hat{\pi}_k\}$ by which the element a_{i_0} is completely associated with only one of the clusters on whose edge that element lies.

Theorem 3.1. *Let $\mathcal{A} = \{a_i \in \mathbb{R}^n \colon i \in I\}$ be a set of data points, and let $c_1^\star, \ldots, c_k^\star \in \mathbb{R}^n$ be the centroids obtained by the* DKM *algorithm. Let U_i, $|U_i| = \mu_i \leqslant k$ be the set of indices associated with the element $a_i \in \mathcal{A}$ by* (2.9).

If there exists $i_0 \in I$, such that $|U_{i_0}| > 1$, then there exist $\hat{c}_1, \ldots, \hat{c}_k \in \mathbb{R}^n$ such that

$$F(\hat{c}_1, \ldots, \hat{c}_k) := \sum_{i=1}^{m} \min_{1 \leqslant j \leqslant k} d(\hat{c}_j, a_i) \leqslant F(c_1^\star, \ldots, c_k^\star). \tag{3.1}$$

Proof. Let us notice that for given $c_1^\star, \ldots, c_k^\star \in \mathbb{R}^n$ and $v_i^{(j)} \in [0, 1]$ given by (2.11), there always exists $w_i^{(j)} \in \{0, 1\}$, $\sum\limits_{j=1}^{k} w_i^{(j)} = 1$, such that

$$F(c_1^\star, \ldots, c_k^\star) = \sum_{i=1}^{m} \sum_{j=1}^{k} v_i^{(j)} d(c_j^\star, a_i) \geqslant \sum_{i=1}^{m} \min_{1 \leqslant j \leqslant k} d(c_j^\star, a_i) = \sum_{i=1}^{m} \sum_{j=1}^{k} w_i^{(j)} d(c_j^\star, a_i)$$
$$\geqslant \sum_{j=1}^{k} \left(\min_{x \in \mathbb{R}^n} \sum_{i=1}^{m} w_i^{(j)} d(x, a_i) \right) = \sum_{i=1}^{m} \sum_{j=1}^{k} w_i^{(j)} d(\hat{c}_j, a_i) = \widehat{F}(\hat{c}_1, \ldots, \hat{c}_k),$$

whereby

$$\hat{c}_j = \operatorname*{argmin}_{x \in \mathbb{R}^n} \sum_{i=1}^{m} w_i^{(j)} d(x, a_i), \quad j \in J.$$

□

The following example shows how a better locally optimal partition can be found by means of an improved DKM algorithm based upon Theorem 3.1 (in the sequel *a modified* DKM *algorithm*) in relation to a locally optimal partition obtained by the k-means algorithm.

E x a m p l e 3.1. By applying the DKM algorithm to the data from Example 2.1 we used a locally optimal partition

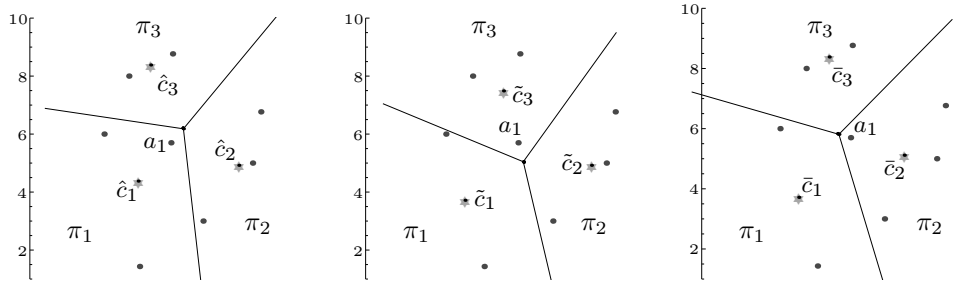
$$\pi_1 = \{(a_1, \tfrac{1}{2}), (a_2, 1), (a_3, 1)\}, \quad \pi_2 = \{(a_4, 1), (a_5, 1), (a_6, 1)\},$$
$$\pi_3 = \{(a_1, \tfrac{1}{2}), (a_7, 1), (a_8, 1)\}.$$

The element a_1 that appears in the Voronoi diagram, is divided into clusters π_1 and π_3 (see Figure 1b), attaining in this way the objective function value $F^\star = 29.8908$.

If the element a_1 is associated with the cluster π_1 (Figure 2a), we obtain new centers $\hat{c}_i$ and a smaller objective function value of 28.8419 (Correction 1). If the same element a_1 is associated with the cluster π_3 (Figure 2b), we obtain new centroids $\tilde{c}_i$ and the same smaller objective function value 28.8419 (Correction 2). Results obtained in this way are compared with the results obtained by the k-means algorithm (see Table 2). With the same initial assignment points $c_i^{(0)}$, the k-means algorithm gives a weaker locally optimal partition

$$\pi_1 = \{a_2, a_3\}, \quad \pi_2 = \{a_1, a_4, a_5, a_6\}, \quad \pi_3 = \{a_7, a_8\},$$

with centroids $\bar{c}_i$ (see Figure 2c). Hence, application of the modified DKM algorithm can give better results in comparison with the k-means algorithm.



(a) Correction 1 ($a_1 \in \pi_1$) (b) Correction 2 ($a_1 \in \pi_3$) (c) Locally optimal k-means partition

Figure 2. Locally optimal partitions

	Centroids	Objective function value
DKM	$(4.113, 4.113)$ $(8.444, 4.922)$ $(5.047, 7.847)$	29.891
Correction 1	$(4.378, 4.378)$ $(8.444, 4.922)$ $(4.883, 8.383)$	28.842
Correction 2	$(3.717, 3.717)$ $(8.444, 4.922)$ $(5.156, 7.489)$	28.842
k-means	$(3.717, 3.717)$ $(7.758, 5.117)$ $(4.883, 8.383)$	29.700

Table 2. Iterative procedure

Association of the data point a_1 with the cluster π_1 or π_3 yields lower, but mutually equal objective function values. The following sample example shows that the objective function value can differ depending on the choice of the cluster with which the data point from the Voronoi diagram is associated.

Example 3.2. Given are the data points $\mathcal{A} = \{1, 2, 6, 11.4\} \subset \mathbb{R}$. Partition

$$\Pi = \{\pi_1, \pi_2\}, \quad \pi_1 = \{(1, 1), (2, 1), (6, \tfrac{1}{2})\}, \quad \pi_2 = \{(6, \tfrac{1}{2}), (11.4, 1)\},$$

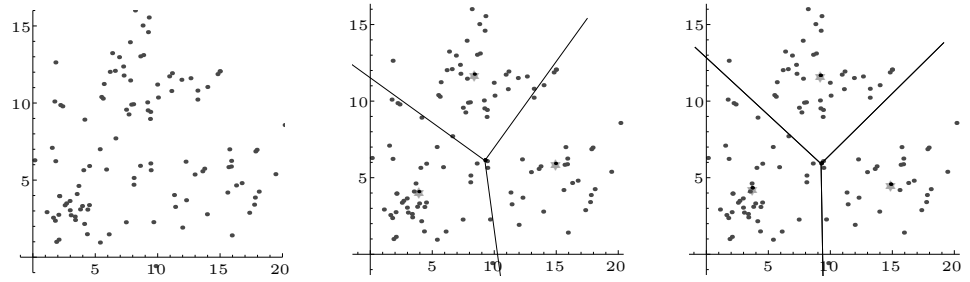
is locally optimal in terms of the DKM algorithm, whereby the corresponding locally optimal centroids are $c_1^* = 2.4$ and $c_2^* = 8.6$, and the objective function value is $F^* = 18.32$. If the data point 6 is associated entirely with the cluster π_1, we obtain new centroids $\hat{c}_1 = 3$ and $\hat{c}_2 = 11.4$ and the objective function value $\widehat{F} = 14$. On the other hand, if the data point 6 is associated entirely with the cluster π_2, we obtain new centroids $\tilde{c}_1 = 1.5$ and $\tilde{c}_2 = 8.7$ and a higher objective function value $\widetilde{F} = 15.08$.

4. Numerical experiments

The next numerical experiment shows that it is possible to construct a set of data with which for a specially given initial approximation the k-means algorithm gives a significantly worse partition than DKM, i.e. SMOKA. The example is constructed so that part of data belongs to the Voronoi diagram of the initial assignment points. Many numerical experiments show that by choosing some other initial approximation all three algorithms yield the same though a higher value of the objective function.

Example 4.1. Let us choose three points $c_1 = (3.9, 4)$, $c_2 = (7.8, 12)$, $c_3 = (15, 4.9) \in \mathbb{R}^2$. In the neighborhood of each point c_i, 40 random points from $\mathcal{N}(c_i, \sigma^2\mathbf{I})$ are generated, where $\mathbf{I}$ is the identity matrix. Also, on the Voronoi diagram $V[c_1, c_2, c_3]$, 15 random points from uniform distribution are generated. In this way, the set $\mathcal{A} = \{a_i \in \mathbb{R}^2 \colon i = 1, \ldots, m\}$ with $m = 135$ points is defined (see Figure 3a). By applying the k-means algorithm, the DKM algorithm and SMOKA with initial assignment points c_1, c_2, c_3 we obtain locally optimal partitions, and the corresponding objective function values. This experiment was repeated for $\sigma^2 = 0.1, 0.5, 1, 1.5, 2$.

As expected, SMOKA and the DKM algorithm yield the same locally optimal partitions that are better than the ones obtained by applying the k-means algorithm (see Table 3). Figure 3b and Figure 3c show the k-means locally optimal partition and the DKM/SMOKA locally optimal partition, respectively.



(a) 135 randomly generated points (b) locally optimal k-means partition (c) locally optimal DKM/SMOKA partition

Figure 3. Locally optimal partitions for randomly generated points with $\sigma^2 = 2.0$

$F^\star$	$\sigma^2 = 0.1$	$\sigma^2 = 0.5$	$\sigma^2 = 1.$	$\sigma^2 = 1.5$	$\sigma^2 = 2.0$
k-means	485.29	521.27	681.90	1027.77	1211.71
DKM	485.29	519.66	667.93	1007.74	1194.17
SMOKA ($\varepsilon = 0.005$)	485.29	519.66	667.93	1007.74	1194.17

Table 3. Objective function values

For the purpose of illustrating the efficiency of the DKM algorithm in relation to SMOKA, we will carry out the following simple numerical experiment motivated by the example from [19].

E x a m p l e 4.2. In a hypercube $H = [0, 1000]^n$ we choose k points $c_1, \ldots, c_k \in H$ at random. The data set $\mathcal{A}$ containing m randomly chosen points from the hypercube H is generated in the following way:

(i) let $i_1, \ldots, i_k$ be randomly generated integers such that $\sum_{s=1}^{k} i_s = m$;

(ii) in the neighborhood of the center c_s we generate a set $\mathcal{A}_s$, which consists of i_s random points from $\mathcal{N}(c_s, 5\mathbf{I})$, where $\mathbf{I}$ is the identity matrix;

(iii) $\mathcal{A} = \bigcup_{s=1}^{k} \mathcal{A}_s$.

We are going to split the set $\mathcal{A}$ into k clusters by applying SMOKA for $\varepsilon = 0.005$ and the DKM algorithm. The experiment will be performed by taking $n \in \{2, 5, 10\}$, $m \in \{1000, 5000, 10000, 20000\}$, and $k \in \{5, 10, 20\}$. Applying the DIRECT algorithm [5],

[11] for solving global optimization problem (1.5) with a relatively low accuracy, we obtain a solution that will be used as an initial approximation for both algorithms.

Figure 4 shows the movement of the CPU times in seconds for each running depending on the number of data points for SMOKA and DKM on a Pentium M processor with 1.4 GHz, respectively. We can notice that the CPU execution time of the DKM algorithm is significantly shorter than the corresponding CPU time required by SMOKA. Let us also mention that in all the experiments the values of the objective function in the centers obtained by DKM and SMOKA differed in less than 10^{-14}.

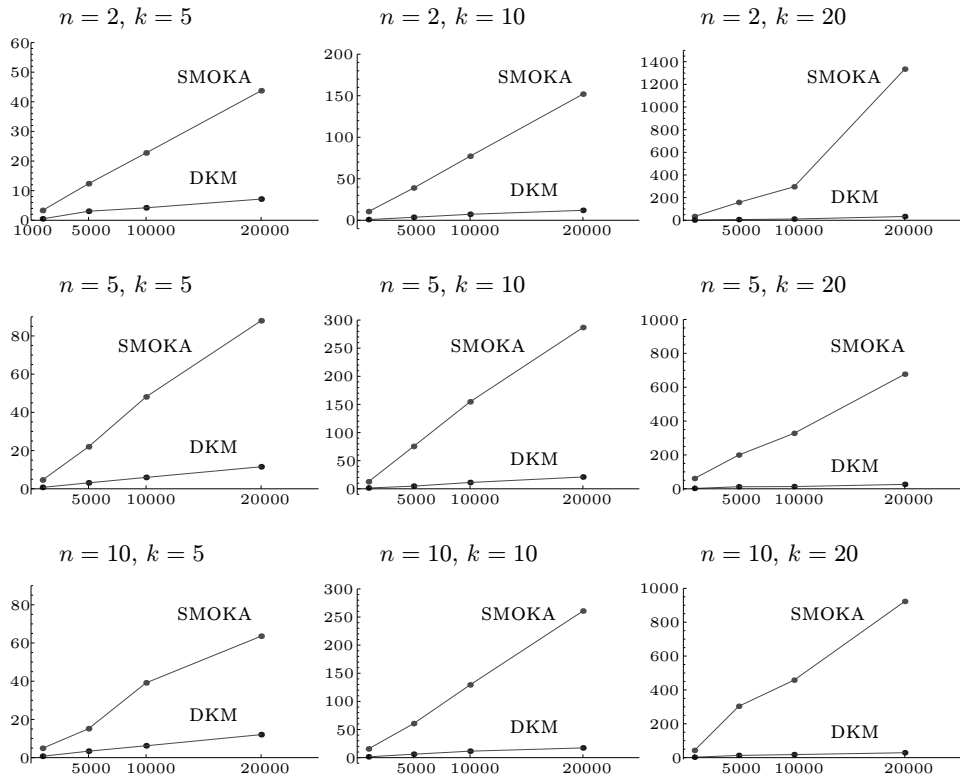


Figure 4. CPU time (in seconds) necessary for the execution of SMOKA and DKM algorithm

Hence, the DKM algorithm gives the same locally optimal partition as SMOKA. If there are no data points on the Voronoi diagram $V[c_1^\star, \ldots, c_k^\star]$, this partition coincides with the k-means locally optimal partition. Otherwise this partition can be improved according to Theorem 3.1. The efficiency measured by the necessary CPU-time is significantly higher by the DKM algorithm than by SMOKA.

5. Conclusions

In this paper, our aim was to point out the mathematical background of the well-known k-means algorithm for searching for the locally optimal partition of the set $\mathcal{A} = \{a_i \in \mathbb{R}^n \colon i = 1, \ldots, m\}$. It has been shown that the k-means algorithm is directly connected with the limit case of another known algorithm for searching for the locally optimal partition, i.e. SMOKA. In this sense, a new DKM algorithm is constructed as a limit case of SMOKA, which differs from the k-means algorithm only in case that during the iterative process some data points appear in the Voronoi diagram. It has been shown that in this case the results can still be improved. In this way, the DKM algorithm gives an improvement of the k-means algorithm. We should thereby stress its efficiency measured by the necessary CPU-time.

Taking into account that the SMOKA algorithm came into existence as a natural generalization of the well-known Weiszfeld algorithm for solving the Fermat-Weber location problem [10] for the case of applying least squares distance-like functions, cases when some other distance-like functions are applied could be treated in a similar way [14].

A c k n o w l e d g e m e n t . The authors would like to thank anonymous referees and journal editors for their careful reading of the paper and insightful comments that helped us improve the paper.

References

[1] *F. Aurenhammer, R. Klein*: Voronoi Diagrams. Handbook of Computational Geometry (J.-R. Sack et al., eds.). North-Holland, Amsterdam, 2000, pp. 201–290.
[2] *A. M. Bagirov*: Modified global k-means algorithm for minimum sum-of-squares clustering problems. Pattern Recognition *41* (2008), 3192–3199.
[3] *A. M. Bagirov, J. Ugon, D. Webb*: Fast modified global k-means algorithm for incremental cluster construction. Pattern Recognition *44* (2011), 866–876.
[4] *J. E. jun Dennis, R. B. Schnabel*: Numerical Methods for Unconstrained Optimization and Nonlinear Equations. Classics in Applied Mathematics 16, SIAM, Philadelphia, 1996.
[5] *D. E. Finkel*: DIRECT Optimization Algorithm User Guide, Center for Research in Scientific Computation. North Carolina State University, 2003, http://www4.ncsu.edu/definkel/research/index.html.
[6] *C. A. Floudas, C. E. Gounaris*: A review of recent advances in global optimization. J. Glob. Optim. *45* (2009), 3–38.
[7] *G. Gan, C. Ma, J. Wu*: Data Clustering: Theory, Algorithms, and Applications. ASA-SIAM Series on Statistics and Applied Probability 20, SIAM, Philadelphia, 2007.
[8] *R. Grbić, E. K. Nyarko, R. Scitovski*: A modification of the DIRECT method for Lipschitz global optimization for a symmetric function. J. Glob. Optim. *57* (2013), 1193–1212.
[9] *C. Iyigun*: Probabilistic Distance Clustering, Ph.D. thesis. Graduate School-New Brunswick, Rutgers, 2007.

[10] *C. Iyigun, A. Ben-Israel*: A generalized Weiszfeld method for the multi-facility location problem. Oper. Res. Lett. *38* (2010), 207–214.
[11] *D. R. Jones, C. D. Perttunen, B. E. Stuckman*: Lipschitzian optimization without the Lipschitz constant. J. Optimization Theory Appl. *79* (1993), 157–181.
[12] *J. Kogan*: Introduction to Clustering Large and High-Dimensional Data. Cambridge University Press, Cambridge, 2007.
[13] *J. Kogan, C. Nicholas, M. Wiacek*: Hybrid clustering of large high dimensional data. Proceedings of the Workshop on Text Mining (M. Castellanos et al., eds.). SIAM, 2007.
[14] *J. Kogan, M. Teboulle*: Scaling clustering algorithms with Bregman distances. Proceedings of the Workshop on Text Mining (M. W. Berry et al., eds.), 2006.
[15] *F. Leisch*: A toolbox for K-centroids cluster analysis. Comput. Stat. Data Anal. *51* (2006), 526–544.
[16] *A. Likas, N. Vlassis, J. J. Verbeek*: The global k-means clustering algorithm. Pattern Recognition *36* (2003), 451–461.
[17] *M. K. Ng*: A note on constrained k-means algorithms. Pattern Recognition *33* (2000), 525–519.
[18] *J. D. Pintér*: Global Optimization in Action. Continuous and Lipschitz Optimization: Algorithms, Implementations and Applications. Nonconvex Optimization and Its Applications 6, Kluwer Academic Publishers, Dordrecht, 1996.
[19] *K. Sabo, R. Scitovski, I. Vazler*: One-dimensional center-based l_1-clustering method. Optim. Lett. *7* (2013), 5–22.
[20] *K. Sabo, R. Scitovski, I. Vazler, M. Zekić-Sušac*: Mathematical models of natural gas consumption. Energy Conversion and Management *52* (2011), 1721–1727.
[21] *H. Späth*: Cluster-Formation und Analyse. Theorie, FORTRAN-Programme und Beispiele. R. Oldenburg Verlag, München, 1983.
[22] *Z. Su, J. Kogan*: Second order conditions for k-means clustering: Partitions vs. centroids. Text Mining 2008 Workshop (held in conjuction with the 8th SIAM International Conference on Data Mining). Atlanta, 2008.
[23] *M. Teboulle*: A unified continuous optimization framework for center-based clustering methods. J. Mach. Learn. Res. *8* (2007), 65–102.
[24] *V. Volkovich, J. Kogan, C. Nicholas*: Building initial partitions through sampling techniques. Eur. J. Oper. Res. *183* (2007), 1097–1105.
[25] *W. Wang, Y. Zhang*: On fuzzy cluster validity indices. Fuzzy Sets Syst. *158* (2007), 2095–2117.
[26] *X. S. Yang*: Firefly algorithms for multimodal optimization. Stochastic Algorithms: Foundations and Applications (O. Watanabe et al., ed.). 5th international symposium, SAGA 2009, Sapporo, Japan. Proceedings. Springer, Berlin. Lecture Notes in Computer Science *5792* (2009), 169–178.

Authors' address: *Kristian Sabo*, *Rudolf Scitovski*, Department of Mathematics, University of Osijek, Osijek, Croatia, e-mail: `ksabo@mathos.hr`, `scitowsk@mathos.hr`.