Iranian Journal of Operations Research
Vol. 8, No. 2, 2017, pp. 02-24

A DC Optimization Algorithm for Clustering Problems with L_1-norm

A.M. Bagirov[1,*], S. Taheri[2]

Clustering problems with the similarity measure defined by the L_1-norm are studied. Characterizations of different stationary points of these problems are given using their difference of convex representations. An algorithm for finding the Clarke stationary points of the clustering problems is designed and a clustering algorithm is developed based on it. The clustering algorithm finds a center of a data set at the first iteration and gradually adds one cluster center at each consecutive iteration. The proposed algorithm is tested using large real world data sets and compared with other clustering algorithms.

***Keywords**: Cluster analysis, Nonsmooth optimization, Smoothing techniques, Incremental algorithm.*

Manuscript was received on 03/10/2017, revised on 11/12/2017 and accepted for publication on 18/12/2017.

1. Introduction

Clustering is an unsupervised partitioning technique dealing with the problems of organization of a collection of patterns into groups based on similarity. It has applications in medicine, engineering and business, to name just a few. There are different types of clustering including fuzzy [12, 20] and hard clustering [17]. Most hard clustering algorithms are either hierarchical or partitional clustering algorithms. Hierarchical clustering algorithms generate a dendrogram representing the nested grouping of patterns and similarity levels at which groupings change [18, 20]. Partitional clustering algorithms find the partition that optimizes a clustering criterion [18]. Here, we are to develop a partitional clustering algorithm.

The similarity measure is essential in clustering. It can particularly be defined using different norms. Clustering problems with the similarity measure defined by the squared Euclidean norm are known as *the minimum sum-of-squares clustering problems*. There are many algorithms for solving such problems including heuristics such as the k-means algorithm and its modifications, metaheuristics such as the simulated annealing, tabu search, variable neighborhood search, genetic algorithms and optimization algorithms such as the branch and bound, cutting plane and interior point (see [17, 24, 30] and references therein).

Clustering problems with the L_1-norm have attracted significantly less attention than those with the squared Euclidean norm. In some applications clustering algorithms with the L_1-norm produce

* Corresponding Author.
[1] Faculty of Science and Technology, Federation University, Ballarat, Australia, Email: a.bagirov@federation.edu.au.
[2] Faculty of Science and Technology, Federation University, Ballarat, Australia, Email: s.taheri@federation.edu.au

[DOI: 10.29252/iors.8.2.2]

easy interpreting results than those with the squared L_2-norm. The former algorithms are more preferred in high dimensional data mining applications [1] and they are also more robust to outliers [34].

To the best of our knowledge, clustering problem with the L_1-norm was first considered in [13] (see also [20]). The k-median algorithm was proposed in [28] and the ISODATA algorithm was introduced in [19]. The X-means algorithm, introduced in [25], allows one to use the L_1-norm based similarity measure. The local search optimization based clustering algorithm is presented [27]. The optimization algorithm using smoothing techniques was introduced in [7] and the nonsmooth optimization clustering algorithm was proposed in [6].

Here, we propose a rather different approach for solving clustering problems with the L_1-norm. This approach is based on a difference of convex (DC) representations of clustering functions. None of the above mentioned algorithms exploits this special structure of the clustering problem. Such a representation has been used to develop algorithms for the minimum sum-of-squares clustering problems. The authors of [2] present a modification of the DCA algorithm and the authors of [10] develop a nonsmooth optimization algorithm for such problems.

We represent the clustering problem as an unconstrained DC programming problem and design an incremental algorithm using this representation. The main contributions of our work are: (i) development of optimality conditions for the clustering problem using the DC representation of its objective function, (ii) a partial smoothing technique for approximation of the clustering functions, (iii) development of a DC optimization algorithm for finding Clarke stationary points of the clustering problem with the L_1-norm, and (iv) design and numerical evaluation of the DC optimization based incremental clustering algorithm and its comparison with other clustering algorithms using large data sets.

The rest of our work is organized as follows. Clustering problems and their DC representations are given in Section 2. In Section 3, smoothing of cluster functions is discussed. Section 4 presents a DC optimization clustering algorithm. Numerical results are reported in Section 5 and Section 6 contains some concluding remarks.

The following notations are used throughout the paper: $\mathbb{R}^n$ is the n-dimensional Euclidean space with the inner product $\langle x, y\rangle = \sum_{i=1}^{n} x_i y_i$ and the associated norm $\|x\| = \langle x, x\rangle^{1/2}$, $x, y \in \mathbb{R}^n$, $B_\varepsilon(x) = \{y \in \mathbb{R}^n : \|y - x\| < \varepsilon\}$ is the open ball centered at x with the radius $\varepsilon > 0$, "conv" is the convex hull of a set. Throughout the paper, vectors are considered as one row and subscripts are used for their coordinates.

We use the Clarke subdifferential as the main tool to design clustering algorithms. A function $f: \mathbb{R}^n \to \mathbb{R}$ is called locally Lipschitz on $\mathbb{R}^n$ if for any bounded subset $X \subset \mathbb{R}^n$ there exists $L > 0$ such that $|f(x) - f(y)| \le L\|x - y\|$, $\forall\, x, y \in X$. The generalized derivative of a locally Lipschitz function f at a point x with respect to a direction $u \in \mathbb{R}^n$ is defined to be [15]

$$f^0(x, u) = \limsup_{y \to x,\ \alpha \downarrow 0} \frac{f(y + \alpha u) - f(y)}{\alpha}.$$

The set $\partial f(x) = \{\xi \in \mathbb{R}^n : f^0(x, u) \ge \langle \xi, u\rangle,\ \forall u \in \mathbb{R}^n\}$ is called the Clarke subdifferential of the function f at x. Each vector $\xi \in \partial f(x)$ is called a subgradient. For convex functions the set $\partial f(x)$ coincides with the classical subdifferential [15].

[DOI: 10.29252/iors.8.2.2]

2. DC Programming Approach to Clustering Problems

In this section, we formulate the clustering and auxiliary clustering problems and give their DC representations.

A function $f: \mathbb{R}^n \to \mathbb{R}$ is called DC if there exist convex functions $g, h: \mathbb{R}^n \to \mathbb{R}$ such that: $f(x) = g(x) - h(x),\ x \in \mathbb{R}^n$. Here, $g - h$ is called a DC decomposition of f while g and h are DC components of f (for more details on DC functions, see [3, 16, 29, 31]).

An unconstrained DC programming problem is

$$\text{minimize } f(x) = g(x) - h(x) \quad \text{subject to} \quad x \in \mathbb{R}^n \tag{1}$$

In general, $\partial f(x) \subseteq \partial g(x) - \partial h(x)$. For a point x^* to be a local minimizer of problem (1), it is necessary to have $0 \in \partial f(x^*)$ [26].

2.1. The Clustering Problem with the L_1-norm

Assume that a finite point set $A = \{a^1, \dots, a^m\} \subset \mathbb{R}^n$ is given. The hard clustering problem is the distribution of the points of the set A into a given number k of disjoint subsets $A^j, j = 1, \dots, k$, such that

$$A^j \neq \emptyset,\ \ A^j \cap A^l = \emptyset,\ \ j, l = 1, \dots, k,\ \ j \neq l,\ \ \text{and}\ \ A = \bigcup_{j=1}^{k} A^j .$$

The sets $A^j,\ j = 1, \dots, k$, are called clusters. The cluster A^j is identified by its center $x^j \in \mathbb{R}^n, j = 1, \dots, k$. The problem of finding these centers is called *the k-clustering* (or *k-partition*) problem. *The similarity (or dissimilarity) measure* is essential to formulate the clustering problem. We define this measure between points $u, v \in \mathbb{R}^n$ using the L_1-norm:

$$d_1(u, v) = \sum_{i=1}^{n} |u_i - v_i|.$$

A nonsmooth optimization formulation of the clustering problem is [9, 11]:

$$\text{minimize } f_k(x) \ \ \text{subject to} \ \ x = \left(x^1, \dots, x^k\right) \in \mathbb{R}^{n,k}, \tag{2}$$

where

$$f^0\left(x^1, \dots, x^k\right) = \frac{1}{m} \sum_{a \in} \min_{j=1,\dots,k} d_1\left(x^j, a\right). \tag{3}$$

The problem (2) is called *the minimum sum-of-absolutes clustering (MSAC) problem* and f_k is called *the k-th cluster function*.

The function f_k, defined in (3), can be expressed as a DC:

[DOI: 10.29252/iors.8.2.2]

$$f_k(x) = g_k(x) - h_k(x), \qquad x = \left(x^1, \dots, x^k\right) \in \mathbb{R}^{n,k}, \tag{4}$$

where

$$g_k(x) = \frac{1}{m}\sum_{a\in A}\sum_{j=1}^{k} d_1\left(x^j, a\right), \qquad h_k(x) = \frac{1}{m}\sum_{a\in A}\max_{j=1,\dots,k}\sum_{s=1,s\neq j}^{k} d_1(x^s, a). \tag{5}$$

Since the function d_1 is nonsmooth in x, both g_k and h_k are, in general, nonsmooth functions.

Let $y \in \mathbb{R}^n$. It is clear that $\partial d_1(y,a) = J_1 \otimes J_2 \otimes \dots \otimes J_n,\ a \in A$ where for $i = 1, \dots, n$, we have

$$J_i = \begin{cases} 1, & \text{if } y_i > a_i, \\ [-1,1], & \text{if } y_i = a_i, \\ -1, & \text{if } y_i < a_i. \end{cases} \tag{6}$$

Let, for convenience, $(B,C) = B \otimes C,\ B, C \subset \mathbb{R}^n$. Then, the subdifferential of the function g_k, defined in (5), at $x = \left(x^1, \dots, x^k\right) \in \mathbb{R}^{n,k}$ is:

$$\partial g_k(x) = \frac{1}{m}\sum_{a\in A}\left(\partial d_1(x^1,a), \partial d_1(x^2,a), \dots, \partial d_1\left(x^k,a\right)\right). \tag{7}$$

To compute the subdifferential of the function h_k, defined in (5), for a given $a \in A$, consider the function

$$\varphi_a(x) = \max_{j=1,\dots,k}\sum_{s=1,s\neq j}^{k} d_1(x^s, a),$$

and define the set

$$R_a(x) = \left\{ j \in \{1, \dots, k\}: \sum_{s=1,s\neq j}^{k} d_1(x^s, a) = \varphi_a(x) \right\}.$$

The set $R_a(x)$ contains indices of clusters to which the point $a \in A$ belongs. The subdifferential $\partial\varphi_a(x)$ of the function φ_a at x is:

$$\partial\varphi_a(x) = \operatorname{conv}\left\{\left(\xi^1, \dots, \xi^{\,j-1}, 0_n, \xi^{j+1}, \dots, \xi^k\right), j \in R_a(x), \xi^t \in \partial d_1(x^t, a), t = 1, \dots, k, t \neq j\right\}.$$

This subdifferential can be rewritten as:

$$\partial\varphi_a(x) = \operatorname{conv}\left\{\left(\partial d_1(x^1,a), \dots, \partial d_1\left(x^{j-1},a\right), 0_n, \partial d_1\left(x^{j+1},a\right), \dots, \partial d_1\left(x^k,a\right)\right),\ j \in R_a(x)\right\}.$$

Then, the subdifferential $\partial h_k(x)$ is expressed as:

[DOI: 10.29252/iors.8.2.2]

$$\partial h_k(x) = \frac{1}{m}\sum_{a\in A}\partial\varphi_a(x). \tag{8}$$

2.2. The Auxiliary Clustering Problem

The problem (2) is a global optimization problem, its objective function f_k has many local minimizers and only its global or deep local minimizers are of interest. The success of local search algorithms for solving problem (2) strongly depends on the choice of the starting cluster centers. Different approaches have been proposed to choose such centers. We apply an approach introduced in [24]. It involves the solution of the so-called auxiliary clustering problem. Next, we describe this problem and give its DC decomposition.

Given the solution $x^1, \ldots, x^{k-1},\ k \geq 2$, to the $(k-1)$-clustering problem, for a data point $a \in A$, we define $r_{k-1}^a = \min\{d_1(x^1, a), \ldots, d_1(x^{k-1}, a)\}$. The function

$$\bar{f}_k(y) = \frac{1}{m}\sum_{a\in A}\min\{r_{k-1}^a, d_1(y,a)\}, \qquad y \in \mathbb{R}^n \tag{9}$$

is called *the k-th auxiliary cluster function* [4, 24]. This function is nonsmooth and as a sum of minimum of convex functions it is, in general, nonconvex. A problem

$$\text{minimize } \bar{f}_k(y) \quad \text{subject to} \quad y \in \mathbb{R}^n \tag{10}$$

is called *the k-th auxiliary clustering problem*.

The function $\bar{f}_k$, given in (9), is a DC, expressed by

$$\bar{f}_k(y) = \bar{g}_k(y) - \bar{h}_k(y) \tag{11}$$

where

$$\bar{g}_k(y) = \frac{1}{m}\sum_{a\in A}\left(r_{k-1}^a + d_1(y,a)\right), \qquad \bar{h}_k(y) = \frac{1}{m}\sum_{a\in A}\max\{r_{k-1}^a, d_1(y,a)\} \tag{12}$$

The subdifferential of the function $\bar{g}_k(y)$ at y can be expressed as:

$$\partial\bar{g}_k(y) = \frac{1}{m}\sum_{a\in A}\partial d_1(y,a). \tag{13}$$

To write the subdifferential $\partial\bar{h}_k(y)$, consider the following sets at y:

$$\bar{A}_1(y) = \{a \in A:\ r_{k-1}^a > d_1(y,a)\}, \qquad \bar{A}_2(y) = \{a \in A:\ r_{k-1}^a < d_1(y,a)\},$$
$$\bar{A}_3(y) = \{a \in A:\ r_{k-1}^a = d_1(y,a)\}.$$

The set $\bar{A}_1(y)$ contains all the points $a \in A$ attracted by the point y, the set $\bar{A}_2(y)$ contains all the points which are closer to their own cluster center than the point y, and the set $\bar{A}_3(y)$ contains the

[DOI: 10.29252/iors.8.2.2]

points which are attracted by the point y and at least one cluster center. Using these sets, the function $\bar{h}_k$, defined in (12), can be rewritten as

$$\bar{h}_k(y) = \frac{1}{m}\left(\sum_{a \in \bar{A}_1(y)} r_{k-1}^a + \sum_{a \in \bar{A}_2(y)} d_1(y,a) + \sum_{a \in \bar{A}_3(y)} \max\{r_{k-1}^a, d_1(y,a)\} \right).$$

Then,

$$\partial \bar{h}_k(y) = \frac{1}{m}\left(\sum_{a \in \bar{A}_2(y)} \partial d_1(y,a) + \sum_{a \in \bar{A}_3(y)} \text{conv}\{0_n, \partial d_1(y,a)\} \right). \tag{14}$$

3. Partial Smoothing of Cluster Functions

Nonsmooth DC functions, in general, are not Clarke regular, that is, their directional and generalized directional derivatives do not always coincide (for the definition of regular functions, see [15]). For such functions, the subdifferential calculus exists in the form of inclusions and is not always applicable to calculate subgradients.

Both the clustering and auxiliary clustering functions are nonsmooth DCs. Their first DC components are quite simple nonsmooth functions, but the second components in both functions are more complex. We propose smoothing their first DC components to allow one to get the full subdifferential calculus.

Note that the direct smoothing of both the clustering and auxiliary clustering functions leads to more complex functions involving many smoothing parameters. However, the use of DC representations allows one to use only one smoothing parameter.

We propose to use the hyperbolic smoothing technique to approximate the first DC components of the clustering functions. Details of this technique can be found in [5, 32, 33].

3.1. Partial Smoothing of the Auxiliary Clustering Function

For given $y \in \mathbb{R}^n$ and $a \in A$, the function d_1 can be rewritten as

$$d_1(y,a) = \sum_{i=1}^{n} [(a_i - y_i) + 2\max(0, y_i - a_i)].$$

Applying the hyperbolic smoothing, we get the following approximation of d_1:

$$d_{1\tau}(y,a) = \sum_{i=1}^{n} ((y_i - a_i)^2 + \tau^2)^{1/2}. \tag{15}$$

[DOI: 10.29252/iors.8.2.2]

Here, $\tau > 0$ is a smoothing parameter. It is obvious that $d_{1\tau}(y,a) \geq d_1(y,a)$, for all $\tau > 0$. The gradient of $d_{1\tau}$ at $y \in \mathbb{R}^n$ is

$$\nabla d_{1\tau}(y,a) = \left(\frac{y_1 - a_1}{((y_1 - a_1)^2 + \tau^2)^{1/2}}, \ldots, \frac{y_n - a_n}{((y_n - a_n)^2 + \tau^2)^{1/2}}\right). \tag{16}$$

We have the following approximation of the function $\bar{g}_k$:

$$\hat{g}_k(y,\tau) = \frac{1}{m}\sum_{a\in A}\left(r_{k-1}^a + d_{1\tau}(y,a)\right). \tag{17}$$

It is easy to see that $\hat{g}_k(y,\tau) \geq \bar{g}_k(y)$, for all $\tau > 0$. For a given $\tau > 0$, the gradient of $\hat{g}_k(\cdot,\tau)$ at $y \in \mathbb{R}^n$ is

$$\nabla \hat{g}_k(y,\tau) = \frac{1}{m}\sum_{a\in A}\nabla d_{1\tau}(y,a). \tag{18}$$

The function $\bar{f}_k$, defined by (11), can be approximated by the following function:

$$\hat{f}_k(y,\tau) = \hat{g}_k(y,\tau) - \bar{h}_k(y), \qquad y \in \mathbb{R}^n.$$

Proposition 1. For any $y \in \mathbb{R}^n$, we have $0 < \hat{f}_k(y,\tau) - \bar{f}_k(y) \leq n\tau$.

Proof. It is clear that $\hat{f}_k(y,\tau) - \bar{f}_k(y) > 0$ for all $\tau > 0$. From (12) and (17) we have

$$\begin{aligned}\hat{f}_k(y,\tau) - \bar{f}_k(y) &= \frac{1}{m}\sum_{a\in A}\left(d_{1\tau}(y,a) - d_1(y,a)\right)\\ &= \frac{1}{m}\sum_{a\in A}\sum_{i=1}^{n}\left(((y_i - a_i)^2 + \tau^2)^{\frac{1}{2}} - |y_i - a_i|\right)\\ &\leq n\tau.\end{aligned}$$

■

Next, we study the relationship between gradients of the function $\hat{g}_k$ and the subdifferential of the function $\bar{g}_k$.

Proposition 2. Let $\{y^j\} \subset \mathbb{R}^n$ and $\{\tau_j\}$ be sequences such that $y^j \to \bar{y},\ \tau_j \downarrow 0$ as $j \to \infty$. Let also $D(\bar{y})$ be a set of limit points of the sequence $\left\{\nabla d_{1\tau_j}(y^j,a)\right\}$. Then,

$$D(\bar{y}) \subseteq \partial d_1(\bar{y},a). \tag{19}$$

Proof. Let $\xi \in \mathbb{R}^n$ be any limit point of the sequence $\left\{\nabla d_{1\tau_j}(y^j,a)\right\}$. Then, there exists the subsequence $\{j_p\}$ such that $j_p \to \infty$ as $p \to \infty$ and

$$\xi = \lim_{p\to\infty}\nabla d_{1\tau_{j_p}}(y^{j_p},a).$$

[DOI: 10.29252/iors.8.2.2]

Let $I = \{1, \dots, n\}$. Consider the following index sets at the point $\bar{y} = (\bar{y}_1, \dots, \bar{y}_n)$:

$$R_1(\bar{y}, a) = \{i \in I: \bar{y}_i < a_i\},\ R_2(\bar{y}, a) = \{i \in I: \bar{y}_i = a_i\},\ R_3(\bar{y}, a) = \{i \in I: \bar{y}_i > a_i\}.$$

It is obvious that for any $\varepsilon > 0$ there exists $\delta > 0$ such that

$$R_1(\bar{y}, a) \subseteq R_1(y, a), \qquad R_3(\bar{y}, a) \subseteq R_3(y, a), \tag{20}$$

for all $y \in B_\delta(\bar{y})$. Take any $i \in I$. If $i \in R_1(\bar{y}, a)$, then according to (20) there exists $p_1 > 0$ such that $i \in R_1\left(y^{j_p}, a\right)$ for all $p > p_1$. It follows from (16) that in this case,

$$\lim_{p\to\infty} \frac{\partial d_{1\tau_{j_p}}\left(y^{j_p}, a\right)}{\partial y_i} = -1.$$

If $i \in R_3(\bar{y}, a)$, then according to (20) there exists $p_2 > 0$ such that $i \in R_3\left(y^{j_p}, a\right)$, for all $p > p_2$. Then, (16) implies that

$$\lim_{p\to\infty} \frac{\partial d_{1\tau_{j_p}}\left(y^{j_p}, a\right)}{\partial y_i} = 1.$$

Finally, assume that $i \in R_2(\bar{y}, a)$. If there exists $p_3 > 0$ such that $i \in R_2\left(y^{j_p}, a\right)$ for all $p > p_3$, then

$$\lim_{p\to\infty} \frac{\partial d_{1\tau_{j_p}}\left(y^{j_p}, a\right)}{\partial y_i} = 0.$$

Otherwise, there might exist three subsequences $\left\{y^{j_{p_t}}\right\},\ \{j_{p_t}\} \subset \{j_p\},\ t = 1,2,3$, such that $\{j_{p_1}\} \cap \{j_{p_2}\} = \{j_{p_1}\} \cap \{j_{p_3}\} = \{j_{p_2}\} \cap \{j_{p_3}\} = \emptyset$ and $i \in R_1\left(y^{j_{p_1}}, a\right),\ i \in R_2\left(y^{j_{p_2}}, a\right), i \in R_3\left(y^{j_{p_3}}, a\right)$. It is obvious that

$$\frac{\partial d_{1\tau_q}\left(y^q, a\right)}{\partial y_i} = 0, \qquad \forall\, q \in \{j_{p_2}\}.$$

Therefore, we consider only two other subsequences. For any $q \in \{j_{p_1}\} \cup \{j_{p_3}\}$, we have

$$\frac{\partial d_{1\tau_q}(y^q, a)}{\partial y_i} = \frac{y_i^q - a_i}{\left(\left(y_i^q - a_i\right)^2 + \tau_q^2\right)^{\frac{1}{2}}} = \frac{y_i^q - a_i}{\left|y_i^q - a_i\right|\left(1 + \dfrac{\tau_q^2}{\left(y_i^q - a_i\right)^2}\right)^{\frac{1}{2}}}.$$

It is clear that any limit of the sequence $\{\frac{\tau_q^2}{\left(y_i^q - a_i\right)^2}\}$ belongs to $[0, \infty) \cup \{\infty\}$. Then, all limits of the sequence $\left(1 + \frac{\tau_q^2}{\left(y_i^q - a_i\right)^2}\right)^{-1/2}$ are in [0,1], meaning that limits of the sequence

[DOI: 10.29252/iors.8.2.2]

$$\left\{\frac{\partial d_{1\tau_q}(y^q,a)}{\partial y_i}\right\}, q \in \{j_{p_1}\} \cup \{j_{p_3}\}$$

belong to $[-1,1]$. It follows from the expression (6) of the subdifferential $\partial d_1(\cdot,a)$ that $\xi \in \partial d_1(\bar{y},a)$. ■

Corollary 1. Let $a \in A$ and $\bar{y} \in \mathbb{R}^n$ be given. For any $\varepsilon > 0$, there exist $\tau_0 = \tau_0(\varepsilon) > 0$ and $\delta = \delta(\varepsilon) > 0$ such that

$$\min_{\xi \in \partial d_1(\bar{y},a)} \|\xi - \nabla d_{1\tau}(y,a)\| < \varepsilon,$$

for all $\tau \in (0,\tau_0)$ and $y \in B_\delta(\bar{y})$.

Proof. Assume the contrary. Then, there exists $\varepsilon_0 > 0$ such that for any $\tau_0 > 0$ and $\delta > 0$, there can be found $\tau \in (0,\tau_0)$ and $y \in B_\delta(\bar{y})$ such that

$$\|\xi - \nabla d_{1\tau}(y,a)\| \geq \varepsilon_0, \qquad \forall\, \xi \in \partial d_1(\bar{y},a).$$

This means that there exist sequences $\{y^j\}$ and $\{\tau_j\}$ such that $y^j \to \bar{y},\ \tau_j \downarrow 0$, as $j \to \infty$ and

$$\left\|\xi - \nabla d_{1\tau_j}(y^j,a)\right\| \geq \varepsilon_0 \ \ \forall \xi \in \partial d_1(\bar{y},a) \ \text{ and } \ j \geq 1.$$

This contradicts Proposition 2 and therefore, the proof is complete. ■

Corollary 2. Let $\{y^j\} \subset \mathbb{R}^n$ and $\{\tau_j\}$ be sequences such that $y^j \to \bar{y},\ \tau_j \downarrow 0$, as $j \to \infty$. Let also $D_0(\bar{y})$ be a set of limit points of the sequence $\{\nabla \hat{g}_k(y^j,\tau_j)\}$. Then,

$$D_0(\bar{y}) \subseteq \partial \bar{g}_k(\bar{y}).$$

Proof. The proof follows from Proposition 2, the expression (13) for the subdifferential $\partial \bar{g}_k$ and the expression (18) for the gradient $\nabla \hat{g}_k(\cdot,\tau)$. ■

Corollary 3. Let $\bar{y} \in \mathbb{R}^n$. For any $\varepsilon > 0$, there exist $\tau_0 = \tau_0(\varepsilon) > 0$ and $\delta = \delta(\varepsilon) > 0$ such that

$$\min_{\xi \in \partial \bar{g}_k(\bar{y})} \|\xi - \nabla \hat{g}_k(y,\tau)\| < \varepsilon,$$

for all $\tau \in (0,\tau_0)$ and $y \in B_\delta(\bar{y})$.

Proof. The proof follows from Corollary 1, the expression (13) for the subdifferential $\partial \bar{g}_k$ and the expression (18) for the gradient $\nabla \hat{g}_k(\cdot,\tau)$. ■

Proposition 3. Let $\tau > 0$. The generalized subdifferential $\partial \hat{f}_k(y,\tau)$ of the function $\hat{f}_k$ at $y \in \mathbb{R}^n$ is

$$\partial \hat{f}_k(y,\tau) = \nabla \hat{g}_k(y,\tau) - \partial \bar{h}_k(y).$$

Proof. The proof is similar to that of Proposition 2 from [10] and therefore is omitted. ■

[DOI: 10.29252/iors.8.2.2]

Let $y \in \mathbb{R}^n$ be any given point. Consider the following set:

$$V(y) = \left\{v \in \mathbb{R}^n: \exists \left(\{\tau_j\}, \tau_j \downarrow 0, \{y^j\}, y^j \to y, j \to \infty, v^j \in \partial \hat{f}_k\left(y^j, \tau_j\right)\right),\ v = \lim_{j\to\infty} v^j\right\}.$$

Proposition 4. For any $\bar{y} \in \mathbb{R}^n$, we have

$$V(\bar{y}) \subseteq \partial \bar{f}_k(\bar{y}).$$

Proof. Let $\bar{y} \in \mathbb{R}^n$ be any point. The generalized derivative of the function $\hat{f}_k$ at the point $y \in \mathbb{R}^n$ with respect to a direction $u \in \mathbb{R}^n$ is

$$\begin{aligned} \hat{f}_k^0\big((y,\tau),u\big) &= \limsup_{z\to y,\alpha\downarrow 0} \frac{\hat{f}_k(z+\alpha u,\tau) - \hat{f}_k(z,\tau)}{\alpha} \\ &= \limsup_{z\to y,\alpha\downarrow 0} \left(\frac{\hat{g}_k(z+\alpha u,\tau) - \hat{g}_k(z,\tau)}{\alpha} - \frac{\bar{h}_k(z+\alpha u,\tau) - \bar{h}_k(z)}{\alpha}\right). \end{aligned} \quad (21)$$

Since for any $\tau > 0$ the function $\hat{g}_k(\cdot,\tau)$ is continuously differentiable, according to the mean value theorem there exists $\theta \equiv \theta(z,u,\alpha) \in (0,1)$ such that

$$\frac{\hat{g}_k(z+\alpha u,\tau) - \hat{g}_k(z,\tau)}{\alpha} = \langle \nabla \hat{g}_k(z+\theta\alpha u,\tau), u\rangle.$$

It is obvious that if $y \in B_{\delta/2}(\bar{y})$ for $\delta > 0$, then for a given $u \in \mathbb{R}^n, u \neq 0_n$, there exists $\alpha_1 > 0$ such that $z + \theta\alpha u \in B_\delta(\bar{y})$, for all $\alpha \in (0,\alpha_1)$ and $z \in B_{\delta/4}(y)$. It follows from Corollary 3 that for any $\varepsilon > 0$ there exist $\tau_0 > 0$ and $\delta_1 > 0$ such that

$$\nabla \hat{g}_k(z+\theta\alpha u,\tau) \in \partial \bar{g}_k(\bar{y}) + B_\varepsilon(0_n),$$

for all $\tau \in (0,\tau_0),\ z + \theta\alpha u \in B_{\delta_1}(\bar{y})$ and $\theta \in (0,1)$. Then, the convexity of the function $\bar{g}_k$ implies that

$$\langle \nabla \hat{g}_k(z+\theta\alpha u,\tau), u\rangle < \bar{g}_k'(\bar{y},u) + \varepsilon\|u\|. \quad (22)$$

Since the function $\bar{h}_k$ is convex, it is regular and $\bar{h}_k'(y,u) = \bar{h}_k^0(y,u), y \in \mathbb{R}^n$. This means that for $\varepsilon > 0$, there exist $\alpha_2 > 0$ and $\delta_2 > 0$ such that

$$\left|\frac{\bar{h}_k(z+\alpha u) - \bar{h}_k(z)}{\alpha} - \frac{\bar{h}_k(\bar{y}+\alpha u) - \bar{h}_k(\bar{y})}{\alpha}\right| < \varepsilon, \quad (23)$$

for all $\alpha \in (0,\alpha_2)$ and $z \in B_{\delta_2}(\bar{y})$. Let $\delta_0 = \min\{\delta_1,\delta_2\}$. Then, from (22) and (23) we get that for all $\tau \in (0,\tau_0),\ \alpha \in (0,\alpha_2)$ and $z \in B_{\delta_0}(\bar{y})$,

$$\frac{\hat{g}_k(z+\alpha u,\tau) - \hat{g}_k(z,\tau)}{\alpha} - \frac{\bar{h}_k(z+\alpha u) - \bar{h}_k(z)}{\alpha} <$$

[DOI: 10.29252/iors.8.2.2]

$$\bar{g}_k'(\bar{y},u)-\frac{\bar{h}_k(\bar{y}+\alpha u)-\bar{h}_k(\bar{y})}{\alpha}+\varepsilon\,(\|u\|+1).$$

This with (21) imply that for any $y\in B_{\delta_0}(\bar{y})$ and $\tau\in(0,\tau_0)$,

$$\begin{aligned}\hat{f}_k^0\big((y,\tau),u\big)&\le\limsup_{\alpha\downarrow 0}\left(\bar{g}_k'(\bar{y},u)-\frac{\bar{h}_k(\bar{y}+\alpha u)-\bar{h}_k(\bar{y})}{\alpha}+\varepsilon\,(\|u\|+1)\right)\\&\le\bar{g}_k'(\bar{y},u)-\bar{h}_k'(\bar{y},u)+\varepsilon\,(\|u\|+1)\\&=\bar{f}_k'(\bar{y},u)+\varepsilon\,(\|u\|+1)\\&\le\bar{f}_k^0(\bar{y},u)+\varepsilon\,(\|u\|+1)\end{aligned}\tag{24}$$

Take any $v\in V(\bar{y})$. This means that $v^j\to v$, for some $v^j\in\partial\hat{f}_k\big(y^j,\tau_j\big),\tau_j\downarrow 0,y^j\to\bar{y},j\to\infty$. There exists sufficiently large $j_0>0$ such that $\tau_j\in(0,\tau_0)$ and $y^j\in B_{\delta_0}(\bar{y})$, for all $j>j_0$. Then, it follows from (24) that for all $u\in\mathbb{R}^n$ and $j>j_0$,

$$\langle v^j,u\rangle\le\hat{f}_k^0\left(\big(y^j,\tau_j\big),u\right)\le\bar{f}_k^0(\bar{y},u)+\varepsilon(\|u\|+1),$$

or

$$\langle v^j,u\rangle\le\bar{f}_k^0(\bar{y},u)+\varepsilon(\|u\|+1).$$

Since $\varepsilon>0$ is arbitrary, we have that $\langle v,u\rangle\le\bar{f}_k^0(\bar{y},u)$, for all $u\in\mathbb{R}^n$. The convexity of the set $\partial\bar{f}_k(\bar{y})$ implies that $v\in\partial\bar{f}_k(\bar{y})$. This completes the proof. ■

Corollary 4. Let $\bar{y}\in\mathbb{R}^n$ be any point. For any $\varepsilon>0$, there exist $\tau_0>0$ and $\delta>0$ such that

$$\partial\hat{f}_k(y,\tau)\subset\partial\bar{f}_k(\bar{y})+B_\varepsilon(0_n),$$

for all $\tau\in(0,\tau_0)$ and $y\in B_\delta(\bar{y})$.

Proof. Assume the contrary. Then, there exists $\varepsilon_0>0$ such that for any $\tau_0>0$ and $\delta>0$, there can be found $\tau\in(0,\tau_0)$ and $y\in B_\delta(\bar{y})$ so that $\partial\hat{f}_k(y,\tau)\not\subset\partial\bar{f}_k(\bar{y})+B_{\varepsilon_0}(0_n)$. This means there exist sequences $\{\tau_j\}$ and $\{y^j\}$ such that $\tau_j\downarrow 0$ and $y^j\to\bar{y}$, as $j\to\infty$, and $\partial\hat{f}_k\big(y^j,\tau_j\big)\not\subset\partial\bar{f}_k(\bar{y})+B_{\varepsilon_0}(0_n)$. In turn, this implies that there exists a sequence $\{v^j\}$ such that $v^j\in\partial\hat{f}_k\big(y^j,\tau_j\big)$ and $v^j\notin\partial\bar{f}_k(\bar{y})+B_{\varepsilon_0}(0_n)$, for all $j>0$. As a bounded sequence $\{v^j\}$ has at least one limit point, without loss of generality, assume that $v^j\to v$ as $j\to\infty$. Then, $v\in V(y)$ and $v\notin\partial\bar{f}_k(\bar{y})+B_{\varepsilon_0}(0_n)$. This contradicts Proposition 4. ■

3.2. Partial Smoothing of the Clustering Function

Applying (15) to $d_1(x^s,a),s=1,\dots,k,\ a\in A$, we get the following approximation of the first DC component g_k of the clustering function (4)

[DOI: 10.29252/iors.8.2.2]

$$\tilde{g}_k(x,\tau) = \frac{1}{m}\sum_{a\in A}\sum_{s=1}^{k} d_{1\tau}(x^s, a). \tag{25}$$

The gradient of the function $\tilde{g}_k$ at $x \in \mathbb{R}^{n,k}$ is

$$\nabla\tilde{g}_k(x,\tau) = \frac{1}{m}\sum_{a\in A}\Big(\eta_a(x^1,\tau),\dots,\eta_a\big(x^k,\tau\big)\Big), \tag{26}$$

where

$$\eta_a(x^s,\tau) = \left(\frac{x_1^s - a_1}{((x_1^s-a_1)^2+\tau^2)^{1/2}},\dots,\frac{x_n^s-a_n}{((x_n^s-a_n)^2+\tau^2)^{1/2}}\right), s = 1,\dots,k, a \in A.$$

Using the function $\tilde{g}_k$, we get the following approximation for the function f_k, defined in (3):

$$\tilde{f}_k(x,\tau) = \tilde{g}_k(x,\tau) - h_k(x).$$

Proposition 5. For any $x \in \mathbb{R}^n$,

$$0 < \tilde{f}_k(x,\tau) - f_k(x) \le kn\tau.$$

Proof. The proof is similar to that of Proposition 1. ■

Proposition 6. Let $\bar{x} \in \mathbb{R}^{n,k}$ and $\{x^j\} \subset \mathbb{R}^{n,k}$, $\{\tau_j\}$ be sequences such that $x^j \to \bar{x}, \tau_j \downarrow 0$, as $j \to \infty$. Let also $U_0(\bar{x})$ be a set of limit points of the sequence $\{\nabla\tilde{g}_k(x^j,\tau_j)\}$. Then,

$$U_0(\bar{x}) \subseteq \partial g_k(\bar{x}). \tag{27}$$

Proof. The proof follows from Proposition 2, the expression (7) for the subdifferential ∂g_k and the expression (26) for the gradient $\nabla\tilde{g}_k(\cdot,\tau)$. ■

Corollary 5. Let $\bar{x} \in \mathbb{R}^{nk}$. For any $\varepsilon > 0$, there exist $\tau_0 = \tau_0(\varepsilon) > 0$ and $\delta = \delta(\varepsilon) > 0$ such that

$$\min_{\xi\in\partial g_k(\bar{x})}\|\nabla\tilde{g}_k(x,\tau) - \xi\| < \varepsilon, \tag{28}$$

for all $\tau \in (0,\tau_0)$ and $x \in B_\delta(\bar{x})$.

Proof. The proof follows from Proposition 6, the expression (7) for the subdifferential ∂g_k and the expression (26) for the gradient $\nabla\tilde{g}_k(\cdot,\tau)$. ■

Proposition 7. Let $\tau > 0$. The generalized subdifferential of the function $\tilde{f}_k$ at $x \in \mathbb{R}^{n,k}$ is:

$$\partial\tilde{f}_k(x,\tau) = \nabla\tilde{g}_k(x,\tau) - \partial h_k(x).$$

For a given point $x \in \mathbb{R}^{n,k}$ consider the following set:

[DOI: 10.29252/iors.8.2.2]

$$\tilde{V}(x) = \left\{v \in \mathbb{R}^{n,k}: \exists \left(\{\tau_j\}, \tau_j \downarrow 0, \{x^j\}, x^j \to x, j \to \infty, v^j \in \partial \tilde{f}_k(x^j, \tau_j)\right), \quad v = \lim_{j\to\infty} v^j\right\}.$$

Proposition 8. Let $\bar{x} \in \mathbb{R}^{n,k}$ be a given point. Then $\tilde{V}(\bar{x}) \subseteq \partial f_k(\bar{x})$.

Proof. The proof is similar to that of Proposition 4 and is therefore omitted. ■

Corollary 6. At a point $\bar{x} \in \mathbb{R}^{n,k}$ for any $\varepsilon > 0$ there exist $\tau_0 > 0$ and $\delta > 0$ such that

$$\partial \tilde{f}_k(x, \tau) \subset \partial f_k(\bar{x}) + B_\varepsilon(0_{n,k}),$$

for all $\tau \in (0, \tau_0)$ and $x \in B_\delta(\bar{x})$.

4. A Clustering Algorithm

Here, we first design an algorithm for solving optimization problems (2) and (10). Then, using this algorithm we introduce an incremental algorithm for solving the clustering problems.

4.1. An Algorithm for Solving Optimization Problems

First, let us consider problem (10). Take a sequence $\{\tau_j\}$ such that $\tau_j \downarrow 0$ as $j \to \infty$. This problem is replaced by the following sequence of problems:

$$\text{minimize } \hat{f}_k(y, \tau_j) \quad \text{subject to} \quad y \in \mathbb{R}^n, \tag{29}$$

For each $\tau_j > 0$, the objective function in problem (29) is represented as a difference of smooth and nonsmooth convex functions. An algorithm for solving such problems is proposed in [10] (Algorithm 3). This algorithm generates a sequence whose all limit points are Clarke stationary points of Problem (29). Applying this algorithm one can design the following algorithm for solving Problem (10).

Algorithm 1. An algorithm for solving problem (10).

1: (Initialization). Select any starting point $y^1 \in \mathbb{R}^n$, a sequence $\{\tau_j\}$ such that $\tau_j \downarrow 0$ as $j \to \infty$ and an optimality tolerance $\sigma > 0$. Set $j := 1$.
2: Apply Algorithm 3 of [10] starting from the point y^j to find the Clarke stationary point y^{j+1} of Problem (29).
3: Set $j := j + 1$. If $\tau_j < \sigma$ then stop, else go to Step 2.

The convergence of Algorithm 1 is studied in the next proposition.

Proposition 9. Assume that the level set $\mathcal{J}(y^1) = \{y \in \mathbb{R}^n: \bar{f}_k(y) \le \bar{f}_k(y^1)\}$ is bounded and $\sigma = 0$. Then, all the limit points of the sequence $\{y^j\}$ generated by Algorithm 1are Clarke stationary points of problem (10).

Proof. First, we show that there exists $\tau_0 > 0$ such that the level sets

$$\mathcal{T}(y^1, \tau) = \{y \in \mathbb{R}^n: \hat{f}_k(y, \tau) \le \hat{f}_k(y^1, \tau)\}$$

[DOI: 10.29252/iors.8.2.2]

are bounded for all $\tau \in (0, \tau_0)$. Assume the contrary. Then, for any $\tau_0 > 0$ there exists $\tau \in (0, \tau_0)$ such that the set $\mathcal{T}(y^1, \tau)$ is not bounded. This means that there exists a sequence $\{\bar{\tau}_p\}$ such that $\bar{\tau}_p \downarrow 0$ as $p \to \infty$ and all sets $\mathcal{T}(y^1, \bar{\tau}_p)$ are not bounded. It follows from Proposition 1 that for any $y \in \mathcal{T}(y^1, \bar{\tau}_p)$,

$$\bar{f}_k(y) < \hat{f}_k(y, \bar{\tau}_p) \leq \hat{f}_k(y^1, \bar{\tau}_p) \leq \bar{f}_k(y^1) + n\bar{\tau}_p. \tag{30}$$

Consider the set $\mathcal{T}_1(y^1, \bar{\tau}_p) = \{y \in \mathbb{R}^n : \bar{f}_k(y) \leq \bar{f}_k(y^1) + n\bar{\tau}_p\}$. It follows from (30) that

$$\mathcal{T}(y^1, \bar{\tau}_p) \subseteq \mathcal{T}_1(y^1, \bar{\tau}_p), \text{ for all } p \geq 1.$$

Since the set $\mathcal{T}(y^1, \bar{\tau}_p)$ is not bounded, the set $\mathcal{T}_1(y^1, \bar{\tau}_p)$ is also not bounded for all $p \geq 1$. This implies that the set $\mathcal{J}(y^1)$ is not bounded which contradicts the assumption of the proposition. Therefore, there exists $j_0 > 0$ such that the sets $\mathcal{T}(y^1, \tau_j)$ are bounded, for all $\tau_j, j > j_0$, and the algorithm finds the Clarke stationary point y^{j+1} of Problem (29) in Step 2. Then,

$$0_n \in \partial \hat{f}(y^{j+1}, \tau_j). \tag{31}$$

Since $y^j \in \mathcal{T}(y^1, \tau_j)$ for all $j > j_0$, and the sets $\mathcal{T}(y^1, \tau_j)$ are bounded, then the sequence $\{y^j\}$ has at least one limit point. Let $\bar{y}$ be a limit point of the sequence $\{y^j\}$. This means that there exists a subsequence $\{y^{j_i}\}$ of the sequence $\{y^j\}$ such that $\{y^{j_i}\} \to \bar{y}$, as $i \to \infty$. It is obvious that $\tau_{j_i} \to 0$, as $i \to \infty$. Then, it follows from Corollary 4 that for any $\varepsilon > 0$, there exists $i_0 > 0, j_{i_0} > j_0 + 1$, such that

$$\partial \hat{f}_k(y^{j_i}, \tau_{j_i - 1} \subset \partial \bar{f}_k(\bar{y}) + B_\varepsilon(0_n) \ \forall i > i_0.$$

Taking into account (31), we have $0_n \in \partial \bar{f}_k(\bar{y}) + B_\varepsilon(0_n)$. Since ε is arbitrary, we get that $0_n \in \partial \bar{f}_k(\bar{y})$ and therefore, $\bar{y}$ is the Clarke stationary point of Problem (10). ■

Algorithm 1 can be modified to solve problem (2). In this modification we replace the starting y^1 in Step 1 by $x^1 \in \mathbb{R}^{n,k}$, the sequence $\{y^j\}$ by the sequence $x^j \subset \mathbb{R}^{n,k}$ and problem (29) in Step 2 by the following problem:

$$\text{minimize } \tilde{f}_k(x, \tau_j) \text{ subject to } x \in \mathbb{R}^{n,k}.$$

The convergence of this modified algorithm is given in the next proposition.

Proposition 10. Assume that the level set $\bar{\mathcal{J}}(x^1) = \{x \in \mathbb{R}^{n,k} : f_k(x) \leq f_k(x^1)\}$ is bounded for a starting point $x^1 \in \mathbb{R}^{n,k}$ and $\sigma = 0$. Then, all the limit points of the sequence $\{x^j\}$ generated by the modified Algorithm 1 are Clarke stationary points of problem (2).

Proof. The proof can be easily obtained form the proof of Proposition 9 by applying Proposition 5 and Corollary 6 instead of Proposition 1 and Corollary 4, respectively. ■

[DOI: 10.29252/iors.8.2.2]

4.2. Incremental Clustering Algorithm

Algorithm 1 is a local search algorithm and its success in finding global or near global minimizers of problem (2) strongly depends on the choice of the starting cluster centers. Different algorithms have been developed to generate starting cluster centers [11, 14, 21, 23, 32]. One such algorithm is based on an incremental approach which has been extensively used to design clustering algorithms [4, 7, 8, 24]. In these incremental algorithms, a data set is static and clusters are computed incrementally. Next, we briefly describe the incremental algorithm (for more details, see [7, 8, 24]).

The incremental clustering algorithm starts with the calculation of the center for the whole data set. Assume that the solution $\left(x^1, \ldots, x^{k-1}\right)$ to the $(k-1)$-partition problem is at hand. In order to solve the k-partition problem we apply the special procedure introduced in [24] to generate a set S_k^1 of starting points for the k-th cluster center. Then, Algorithm 1 is applied to solve problem (10) starting from each point of S_k^1. As a result, we obtain a new set S_k^2 of points which are stationary for problem (10). These points provide more decrease of the clustering function value than their starting points from the set S_k^1. In the next step of the incremental algorithm, each point from the set S_k^2 is added to the set of $k-1$ cluster centers from the previous iteration to obtain a starting point for solving the k-partition problem (2). This means that we will get a set of stationary points of the k-partition problem. The best solution among these stationary points is chosen to be a solution of the k-partition problem. The incremental algorithm terminates when the required number of clusters are computed.

The incremental algorithm, in addition to the k-partition problem, solves also all the intermediate l-partition problems, where $l = 1, \ldots, k-1$. Such an algorithm allows one to find a good quality solution to the nonconvex clustering problem. However, there is no guarantee that this algorithm will always find global solutions of these problems. Since an optimization algorithm based on DC representations of the clustering and auxiliary clustering functions is used within the incremental algorithm, we call the proposed algorithm as IDCClust (Incremental DC Clustering).

5. Numerical Results

To test the IDCClust algorithm and to compare it with other clustering algorithms, numerical experiments with a number of real-world data sets were carried out. We used the following two algorithms for comparison:

1. An algorithm for clustering with the L_1-norm based on smoothing techniques (SMOOTH), as proposed in [7].
2. The multi-start k-medians algorithm (MS-KMD).

The IDCClust algorithm contains a special procedure to generate the starting cluster centers and the implementation of this procedure is discussed in [24]. This algorithm also uses a minimization algorithm from [10], where details of its implementation are given. The implementation of the SMOOTH algorithm is described in [7]. In MS-KMD, the initial cluster centers are randomly generated among data points. In each data set, this algorithm is allowed to use CPU time and the distance function evaluations more than those used by the IDCClust algorithm.

All algorithms were implemented in Fortran 95, compiled using the *g95* compiler and the calculations were carried out on a 2.90 GHz Intel Core i5-3470S machine with 8 GB of RAM. We used

[DOI: 10.29252/iors.8.2.2]

12 data sets in numerical experiments, with brief description as given in Table 1. Their detailed description can be found in [22, 26]. Data sets contain only numeric attributes and do not have missing values. In these data sets, the number of attributes ranges from 2 to 128 and the number of data points ranges from hundreds (smallest 150) to hundreds of thousands (largest 434,874).

The following notations are used to present computational results:

- m is the number of observations (data points);
- n is the number of attributes;
- k is the number of clusters;
- f_{best} (multiplied by the number shown after name of data set) is the best known value of the cluster function (3) (multiplied by m) for the corresponding number of clusters;
- The sign "-"shows that an algorithm failed to compute clusters in the given time frame.

The error E_A is computed as:

$$E_A = \frac{f_A - f_{best}}{f_{best}} \times 100\%,$$

where f_A is the value of the clustering function f_k obtained by an algorithm A. If $E_A = 0$, then an algorithm finds the best known solution.

Table 1. A brief description of the data sets.

Data sets	m	n
Iris Plants	150	4
TSPLIB3038	3038	2
Page Blocks	5473	10
Gas Sensor Array Drift	13910	128
EEG Eye State	14980	14
Letter Recognition	20000	16
KEGG Metabolic Relation Network	53413	20
Shuttle Control	58000	9
Pla85900	85900	2
Localization Data for Person Activity	164860	3
Skin Segmentation	245057	3
3D Road Network	434874	3

We computed up to 10 clusters in Iris Plants data set and up to 25 clusters in other 11 data sets. The CPU time used by the algorithms is limited to 20 hours. Results for cluster function values found by different algorithms are presented in tables 2 and 3. In these tables, for brevity, IDC stands for IDCClust, SM for SMOOTH and MS for MS-KDM algorithms.

Table 2 presents results for data sets with $m \leq 20{,}000$. In two smallest data sets, Iris Plants and TSPLIB3038, all algorithms were equally successful. They were able to find the best solutions with high accuracies almost in all cases. On the Page Blocks data set, which has no well separated clusters, the SMOOTH algorithm is the most successful. The IDCClust algorithm is able to find the near best solutions, and the MS-KMD algorithm fails in most cases. On the Gas Sensor Array Drift data set, which has the largest number of attributes among all the data sets, the MS-KMD algorithm was the most successful. Although the SMOOTH algorithm found the best solution, it calculated only seven

[DOI: 10.29252/iors.8.2.2]

clusters within the 20 hours time period. Outcomes by the IDCClust algorithm were the worst in this data set.

Table 2. Best cluster function values and errors.

k	f_{best}	E_{IDC}	E_{SM}	E_{MS}	k	f_{best}	E_{IDC}	E_{SM}	E_{MS}
	Iris ($\times 10^4$)					TSPLIB3038 ($\times 10^6$)			
2	2.1670	0.00	0.00	0.00	2	3.7308	0.00	0.00	0.00
3	1.5920	0.00	0.00	0.00	3	3.0056	0.00	0.00	0.00
4	1.3650	0.00	0.00	0.00	5	2.2551	0.00	0.00	0.00
5	1.2460	0.00	0.96	0.00	7	1.8932	0.01	0.02	0.00
6	1.1530	0.00	0.00	0.00	10	1.5447	0.54	0.03	0.00
7	1.0620	0.00	0.00	0.00	12	1.3940	0.75	0.18	0.00
8	1.0010	0.00	0.30	0.00	15	1.2295	0.03	0.10	0.00
9	0.9510	0.00	0.63	0.00	20	1.0595	0.00	0.02	0.16
10	0.9070	0.00	0.44	0.00	25	0.9435	0.18	0.08	0.00
	Page Blocks ($\times 10^7$)					Gas Sensor Array Drift ($\times 10^9$)			
2	0.8414	0.00	0.00	25.28	2	2.2743	0.95	0.00	0.00
3	0.6747	0.00	0.00	0.00	3	1.8987	0.88	0.00	0.00
5	0.4882	0.03	0.00	0.00	5	1.4508	8.37	0.01	0.00
7	0.3909	0.68	0.00	5.22	7	1.2371	1.95	0.01	0.00
10	0.3170	0.16	0.00	13.48	10	1.0653	4.76	-	0.00
12	0.2849	1.48	0.00	14.78	12	0.9765	6.77	-	0.00
15	0.2562	1.27	0.00	22.94	15	0.8873	7.50	-	0.00
20	0.2193	2.42	0.00	25.30	20	0.7957	6.15	-	0.00
25	0.1979	2.51	0.00	28.38	25	0.7313	5.34	-	0.00
	EEG Eye State ($\times 10^7$)					Letter Recognition ($\times 10^6$)			
2	0.5289	0.00	0.00	15.46	2	0.4833	0.00	0.02	0.00
3	0.4197	0.00	0.00	0.00	3	0.4576	0.00	5.64	0.04
5	0.2944	0.00	0.00	0.00	5	0.4225	1.61	8.75	0.00
7	0.2493	0.00	0.00	4.93	7	0.4038	1.36	6.71	0.00
10	0.2173	0.00	0.00	13.78	10	0.3778	0.00	8.69	0.23
12	0.2090	0.01	0.00	13.13	12	0.3644	0.00	7.76	0.47
15	0.1966	0.36	0.00	23.10	15	0.3519	0.00	3.80	0.25
20	0.1827	0.03	0.00	22.99	20	0.3329	0.00	4.45	0.28
25	0.1740	0.11	0.00	26.83	25	0.3188	0.00	3.35	0.57

The performance of the IDCClust and SMOOTH algorithms on the EEG Eye State data set was similar, whereas MS-KDM failed in this data set. The IDCClust and the MS-KDM algorithms showed a similar performance on the Letter Recognition data set, and the SMOOTH algorithm failed to find the best or near best solutions in most cases. These results show that optimization based clustering algorithms are not always efficient in data sets with the large number of attributes, as large scale optimization problems are required to be solved at each iteration of the incremental algorithm.

Results for the data sets with $m > 50{,}000$ are presented in Table 3. The MS-KMD algorithm demonstrated a good performance on the Pla85900 data set, failed on two other data sets: Shuttle Control and KEGG Metabolic Relation Network. This algorithm was not applicable to very large data sets and in such data sets it cannot be considered as an alternative to the other two algorithms. The SMOOTH algorithm was successful on all the data sets except the KEGG Metabolic Relation

[DOI: 10.29252/iors.8.2.2]

Network with $k = 7,10$ and the 3D Road Network with $k = 20, 25$. Overall, the IDCClust algorithm demonstrated the best performance on all the data sets with $m > 50{,}000$.

Table 3. Best cluster function values and errors (continued).

k	f_{best}	E_{IDC}	E_{SM}	E_{MS}	k	f_{best}	E_{IDC}	E_{SM}	E_{MS}
	KEGG Relation Network ($\times 10^6$)					Shuttle Control ($\times 10^7$)			
2	3.5860	0.12	1.08	0.00	2	0.3891	0.00	0.01	0.00
3	2.7979	0.27	0.04	0.00	3	0.3443	0.00	0.02	2.47
4	2.0779	0.00	1.26	0.00	5	0.3001	0.58	0.72	0.00
5	1.7222	1.14	2.78	4.93	7	0.2624	0.01	0.00	1.14
6	1.4497	0.00	3.70	13.78	10	0.2311	0.00	0.03	3.52
7	1.3294	0.00	0.73	13.13	12	0.2172	0.31	0.00	2.14
8	1.2130	0.14	0.00	23.10	15	0.1973	1.34	0.00	2.39
9	1.0765	0.00	0.06	22.99	20	0.1763	1.02	0.00	6.73
10	0.9894	0.00	0.64	26.83	25	0.1603	0.00	0.00	10.51
	Pla85900 ($\times 10^{10}$)					Localization Data ($\times 10^5$)			
2	2.0656	0.00	0.21	0.42	2	1.7626	0.00	0.00	-
3	1.6259	0.00	0.14	0.02	3	1.5151	0.00	0.00	-
5	1.2571	0.12	1.77	0.00	5	1.2717	0.00	0.00	-
7	1.0615	0.00	0.83	0.53	7	1.1044	0.00	0.00	-
10	0.8946	0.00	1.50	0.07	10	0.9612	0.01	0.00	-
12	0.8169	0.15	1.40	0.00	12	0.9050	0.18	0.00	-
15	0.7330	0.07	1.59	0.00	15	0.8477	0.01	0.00	-
20	0.6362	0.22	1.27	0.00	20	0.7709	0.00	0.07	-
25	0.5709	0.00	1.40	0.05	25	0.7192	0.00	0.18	-
	Skin Segmentation ($\times 10^7$)					3D Road Network ($\times 10^6$)			
2	2.3069	0.00	0.00	-	2	3.7950	0.00	0.00	-
3	1.8485	0.00	0.00	-	3	2.6869	0.00	0.00	-
5	1.3539	0.00	0.00	-	5	1.7861	0.00	0.00	-
7	1.0504	0.00	0.00	-	7	1.3790	0.00	0.00	-
10	0.8490	0.00	0.05	-	10	1.0719	0.00	0.00	-
12	0.7565	0.00	0.03	-	12	0.9494	0.00	0.00	-
15	0.6799	0.00	0.03	-	15	0.8248	0.01	0.00	-
20	0.5940	0.00	0.01	-	20	0.7006	0.00	2.35	-
25	0.5284	0.00	0.00	-	25	0.6243	0.00	1.78	-

Fig. 1 depicts the dependence of the number of distance function evaluations on the number of clusters for the IDCClust and SMOOTH algorithms. We do not include the MS-KMD algorithm, since its CPU time and the number of distance function evaluations were fixed beforehand. This figure demonstrates that in most cases, specially in three largest data sets, the SMOOTH algorithm required more distance function evaluations than the IDCClust algorithm.

The dependence of the CPU time used by algorithms on the number of clusters are given in Fig. 2 In general, on large data sets, the IDCClust algorithm required less CPU time than the SMOOTH algorithm. As the number of attributes increased, both algorithms became more time consuming. However, in this case the SMOOTH requires significantly more CPU time than the IDCClust. This is due to the fact that the smooth approximations of clustering functions in the SMOOTH algorithm were more complex than those in the IDCClust algorithm.

[DOI: 10.29252/iors.8.2.2]

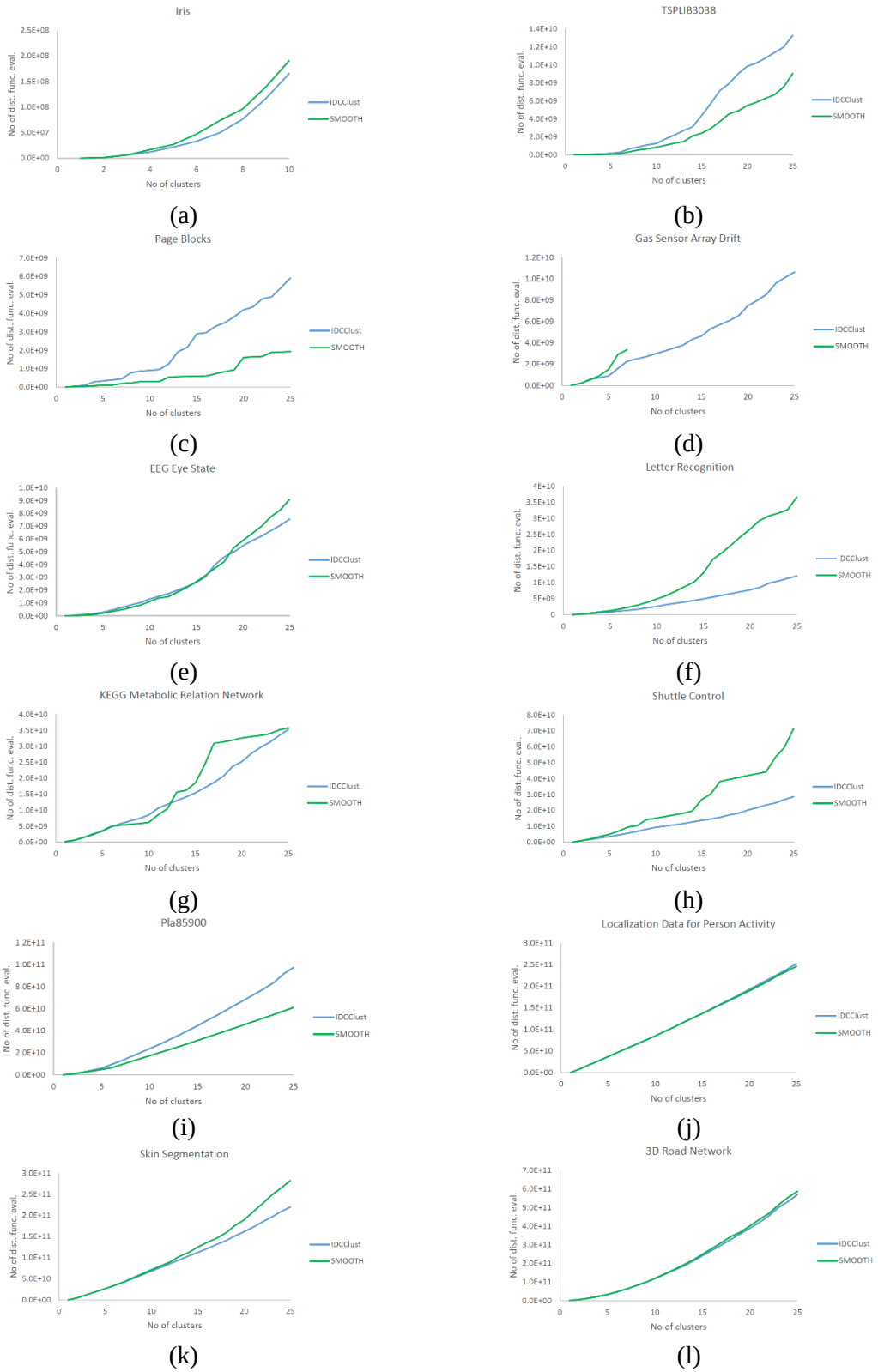

Figure 1. Number of distance function evaluations versus number of clusters.

[DOI: 10.29252/iors.8.2.2]

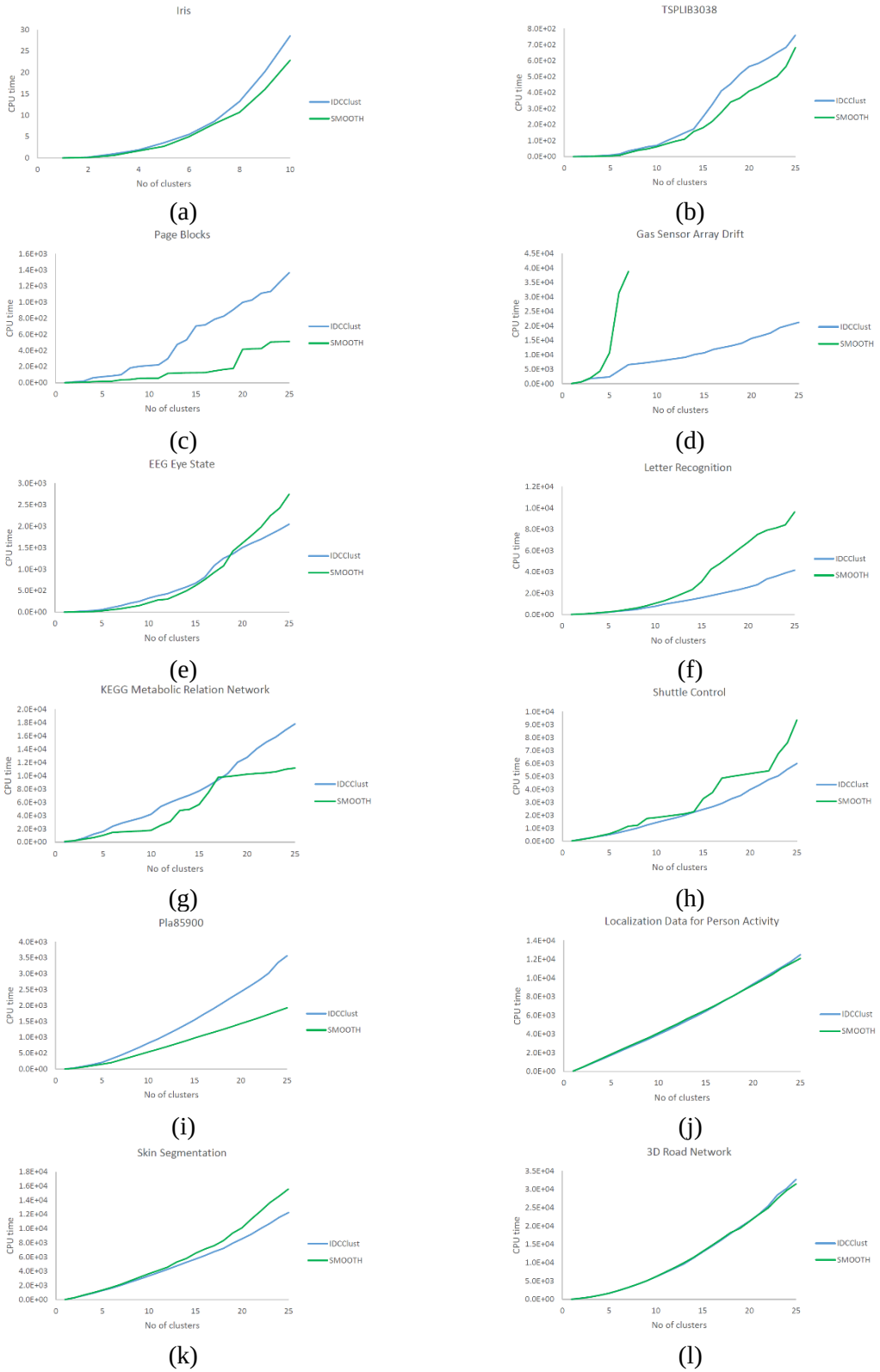

Figure 2. CPU times versus number of clusters.

[DOI: 10.29252/iors.8.2.2]

6. Conclusion

An algorithm for solving clustering problems with the similarity measure defined by the L_1-norm was introduced. This algorithm was based on an explicit DC representation of the clustering functions. These functions were complex nonsmooth nonconvex whose smoothing might require a large number of parameters. However, in the DC representations of clustering functions first the DC components were simple convex nonsmooth functions which could easily be smoothed using only one smoothing parameter. Such an approach allowed one to define a partial smoothing of the clustering functions whose Clarke subgradients could be efficiently computed.

An optimization algorithm for finding Clarke stationary points of clustering problems was designed and its convergence behavior was studied. An incremental clustering algorithm was developed using the optimization algorithm. The proposed clustering algorithm was tested and compared with other clustering algorithms using several real world data sets including those with large number of data points. Two algorithms, used for comparison, were the multi-start k-medians algorithm and an algorithm based on smoothing of the clustering functions without using its DC decomposition. Results demonstrated that the proposed algorithm significantly outperformed the multi-start k-medians algorithm on large data sets although on small data sets their performances were similar. The proposed algorithm outperformed the second algorithm on large data sets as well as on data sets with large number of attributes.

Numerical results demonstrate that the proposed algorithm was efficient for clustering on data sets containing hundreds of thousands of data points. However, it also had some limitations. The algorithm was time consuming on data sets having large number of attributes (hundreds and more) and/or with large number of data points (millions and more). The algorithm for minimization of clustering functions converged to Clarke stationary points of these functions. The study of algorithms that can guarantee convergence to inf-stationary points of these functions will be subject of future research. Calculation of such stationary points may improve the quality of the solution obtained by the incremental algorithm.

The Fortran source code of the proposed algorithm is available by request from the authors.

Acknowledgement

This research was supported under Australian Research Council's Discovery Projects funding scheme (Project No. DP140103213).

References

[1] Aggarwal, C., Hinneburg, A., and Keim D. (2001), On the surprising behavior of distance metrics in high dimensional space, *In ICDT '01 Proceedings of the 8th International Conf. on Database Theory*, 420 – 434.

[2] An, L.T.H., Minh, L.H., and Tao, P.D. (2014), New and efficient DCA based algorithms for minimum sum-of-squares clustering, *Pattern Recognition*, 47, 388–401.

[3] An, L.T.H., and Tao, P.D. (2005), The DC (difference of convex functions) programming and DCA revisited with DC models of real world nonconvex optimization problems, *Annals of Operations Research*, 133, 23–46.

[DOI: 10.29252/iors.8.2.2]

[4] Bagirov, A.M. (2008), Modified global k-means algorithm for minimum sum-ofsquares clustering problems, *Pattern Recognition*, 41(10), 3192–3199.
[5] Bagirov, A.M., Nuaimat, A. Al, and Sultanova, N. (2013), Hyperbolic smoothing function method for minimax problems, *Optimization*, 62(6), 759–782.
[6] Bagirov, A.M., and Mohebi, E. (2015), Nonsmooth optimization based algorithms in cluster analysis, In E. Celebi, editor, *Partitional Clustering Algorithms*, pages 99–146. Springer International Publishing.
[7] Bagirov, A.M., and Mohebi E. (2016), An algorithm for clustering using l_1-norm based on hyperbolic smoothing technique, *Computational Intelligence*, 32(3), 439–457.
[8] Bagirov, A.M., Ordin, B., Ozturk, G., and Xavier, A.E. (2015), An incremental clustering algorithm based on hyperbolic smoothing, *Comp. Opt. and Appl.*, 61(1), 219–241.
[9] Bagirov, A.M., Rubinov, A.M., Soukhoroukova, N.V., and Yearwood, J. (2003), Unsupervised and supervised data classification via nonsmooth and global optimization, *Top*, 11, 1–93.
[10] Bagirov, A.M., Taheri, S., and Ugon, J. (2016), Nonsmooth DC programming approach to the minimum sum-of-squares clustering problems, *Pattern Recognition*, 53, 12–24.
[11] Bagirov, A.M., and Yearwood, J. (2006), A new nonsmooth optimization algorithm for minimum sum-of-squares clustering problems, *European Journal of Operational Research*, 170(2), 578–596.
[12] Bezdek, J.C. (1981), Pattern Recognition with Fuzzy Objective Function Algorithms, Plenum Press, New York.
[13] Carmichael, J.W., and Sneath, P.H.A. (1969), Taxometric maps, *Systematic Zoology*, 18, 402–415.
[14] Celebi, M.E., Kingravi, H.A., and Vela, P.A. (2013), A comparative study of efficient initialization methods for the k-means clustering algorithm, *Expert Systems with Applications*, 40, 200–210.
[15] Clarke, F.H. (1983), Optimization and Nonsmooth Analysis, Canadian Mathematical Society series of monographs and advanced texts, Wiley.
[16] Horst, R., and Thoai, N.V. (1999), DC programming: Overview, *Journal of Optimization Theory and Applications*, 103(1), 1–43.
[17] Jain, A.K. (2010), Data clustering: 50 years beyond k-means, *Pattern Recognition Letters*, 31(8), 651–666.
[18] Jain, A.K., Murty, M.N., and Flynn, P.J. (1999), Data clustering: A review, *ACM Comput. Surv.*, 31(3), 264–323.
[19] Jajuga, K. (1987), A clustering method based on the l_1-norm, *Computational Statistics and Data Analysis*, 5, 357–371.
[20] Kaufman, L., and Rousseeuw, P.J. (1990), Finding Groups in Data: An Introduction to Cluster Analysis, Wiley Series in Probability and Statistics. Wiley.
[21] Lai, J.Z.C. and Huang, T.-J. (2010), Fast global k-means clustering using cluster membership and inequality, Pattern Recognition, 43(5), 1954–1963.
[22] Lichman, M. (2013), UCI machine learning repository, http://archive.ics.uci.edu/ml, University of California, Irvine, School of Information and Computer Sciences.
[23] Likas, A., Vlassis, N., and Verbeek, J. (2003), The global k-means clustering algorithm, *Pattern Recognition*, 36(2), 451–461.
[24] Ordin, B., and Bagirov, A.M. (2015), A heuristic algorithm for solving the minimum sum-of-squares clustering problems, *Journal of Global Optimization*, 61(2), 341–361.
[25] Pelleg, D., and Moore, A.W. (2000), x-means: Extending k-means with efficient estimation of the number of clusters, In P. Langley, editor, *ICML'00 Proceedings of the 17-th International Conf. on Machine Learning*, pages 727–734. Morgan Kaufmann Publishers Inc.

[DOI: 10.29252/iors.8.2.2]

[26] Reinelt, G. (1991), TSP-LIB-A traveling salesman library. *ORSA J. Comput.*, 3, 319–350.
[27] Sabo, K. (2014), Center-based l_1-clustering method, *International Journal of Applied Mathematics and Computer Science*, 24(1), 151–163.
[28] Späth, H. (1976), Algorithm 30: l_1 cluster analysis, *Computing*, 16(4), 379–387.
[29] Tao, P.D., and An, L.T.H. (1997), Convex analysis approach to DC programming: theory, algorithms and applications, *Acta Mathematica Vietnamica*, 22(1), 289–355.
[30] Teboulle, M. (2007), A unified continuous optimization framework for center-based clustering methods, *The J. of Machine Learning Research*, 8,65–102.
[31] Tuy, H. (1998), Convex Analysis and Global Optimization, Nonconvex Optimization and Its Applications, Vol. 22. Kluwer, 1998.
[32] Xavier, A.E. (2010), The hyperbolic smoothing clustering method. *Pattern Recognition*, 43, 731–737.
[33] Xavier, A.E., and Oliveira, A.A.F.D. (2005), Optimal covering of plane domains by circles via hyperbolic smoothing, *Journal of Global Optimization*, 31(3), 493–504.
[34] Zhang, J., Peng, L., Zhao, X., and Kuruoglu, E.E. (2012), Robust data clustering by learning multi-metric lq-norm distances, *Expert Systems with Applications*, 39, 335–349.

[DOI: 10.29252/iors.8.2.2]