Minimization of the sum of minima of convex functions and its application to clustering

A.M. Rubinov, N.V. Soukhoroukova and J. Ugon

CIAO and School of Information Technology and Mathematical Sciences
University of Ballarat

Abstract

We study functions that can be represented as the sum of minima of convex functions. Minimization of such functions can be used for approximation of finite sets and their clustering. We suggest to use the local discrete gradient (DG) method [1] and the hybrid method between the cutting angle method and the discrete gradient method (DG+CAM) [5] for the minimization of these functions. We report and analyze the results of numerical experiments.

Key words: sum-min function, cluster function, skeleton, discrete gradient method, cutting angle method

1 Introduction

In this paper we introduce and study a class of sum-min functions. This class $\mathcal{F}$ consists of functions of the form

$$F(x_1, \dots, x_k) = \sum_{a \in A} \min(\varphi_1(x_1, a), \varphi_2(x_2, a), \dots \varphi_k(x_k, a)),$$

where A is a finite subset of a finite dimensional space and the function $x \mapsto \varphi_i(x, a)$ is convex for each i and $a \in A$. In particular, the cluster function (see, for example, [3]) and Bradley-Mangasarian function [8] belong to $\mathcal{F}$. We also introduce the notion of a skeleton of the set A, which is a version of Bradley-Mangasarian approximation of a finite set. The search for skeletons can be carried out by a constrained minimization of a certain function belonging to $\mathcal{F}$.

We point out some properties of functions $F \in \mathcal{F}$. In particular we show that these functions are DC (difference of convex) functions.

Functions $F \in \mathcal{F}$ are nonsmooth and nonconvex. If the set A is large enough then these functions have a large number of shallow local minima. Some functions $F \in \mathcal{F}$ (in particular, cluster functions) have a saw-tooth form. The minimization of these functions is a challenging problem. We consider both local and global minimization of functions $F \in \mathcal{F}$. We suggest to use the derivative-free discrete gradient (DG) method [1] for local minimization of these functions. For global minimization we use the hybrid method between DG and the cutting angle method

(DG+CAM)[4, 5] and the commercial software GAMS (LGO solver), see [19] and [20] for more information.

These methods were applied to the minimization of two types of functions from $\mathcal{F}$: cluster functions C_k (generalized cluster functions $\tilde{C}_k$) and skeleton functions L_k (generalized skeleton functions $\tilde{L}_k$). These functions are used for finding clusters in datasets (unsupervised classification).

The notion of clustering is relatively flexible (see [14] and [2] for more information). The goal of clustering is to group points in a dataset in a way that representatives of the same group (the same cluster) are similar to each other. There are different notions of similarity. Very often it is assumed that similar points have similar coordinates because each coordinate represents measurements of the same characteristic. The functions C_k, $\tilde{C}_k$, L_k, $\tilde{L}_k$ can be used to represent the dissimilarity of obtained systems of clusters. Therefore, a clustering system which gives a minimum of a chosen dissimilarity function is considered as a desired clustering system. Different dissimilarity functions lead to different approaches to clustering, therefore different clustering results can be obtained by the minimization of functions $F \in \mathcal{F}$.

We report results of numerical experiments and analyze these results.

2 A class of sum-min functions

2.1 Functions represented as the sum of minima of convex functions

Consider finite dimensional vector space $\mathbb{R}^n$ and $\mathbb{R}^m$. Let $A \subset \mathbb{R}^n$ be a finite set and let k be a positive integer. Consider a function F defined on $(\mathbb{R}^m)^k$ by

$$F(x_1, \dots, x_k) = \sum_{a \in A} \min(\varphi_1(x_1, a), \varphi_2(x_2, a), \dots \varphi_k(x_k, a)), \tag{1}$$

where $x \mapsto \varphi_i(x, a)$ is a convex function defined on $\mathbb{R}^m$ ($i = 1, \dots, k$, $a \in A$). We do not assume that this function is smooth. We denote the class of functions of the form (1) by $\mathcal{F}$.

The search for some geometric characteristics of a finite set can be accomplished by minimization (either unconstrained or constrained) of functions from $\mathcal{F}$, (see, for example [3, 8]). Location problems (see, for example, [9]) also can be reduced to the minimization of functions from $\mathcal{F}$.

The minimization of function $F \in \mathcal{F}$ is a *min-sum-min* problem. We also can consider *min-max-min* problems with the objective function

$$\tilde{F}(x_1, \dots, x_k) = \max_{a \in A} \min(\varphi_1(x_1, a), \varphi_2(x_2, a), \dots \varphi_k(x_k, a)).$$

Using sum-min function F we take into account the contribution of each point $a \in A$ to a characteristic of the set A, which is described by means of functions $\varphi_i(x, a)$. This is not true if we consider $\tilde{F}$. From this point of view, the minimization of sum-min functions is preferable for examination of many characteristics of finite sets.

2.2 Some properties of functions belonging to $\mathcal{F}$.

Let $F \in \mathcal{F}$, that is
$$F(x_1, \dots, x_k) = \sum_{a \in A} \min_{i=1,\dots,k} \varphi_i(x_i, a),$$
where $x \mapsto \varphi_i(x_i, a)$ is a convex function. Then F enjoys the following properties:

1. F is quasidifferentiable ([11]). Moreover, F is DC (the difference of convex functions). Indeed, we have (see for example [11], p.108):
$$F(x) = f_1(x) - f_2(x), \qquad x = (x_1, \dots, x_k),$$
where
$$f_1(x) = \sum_{a \in A} \sum_{i=1}^{k} \varphi_i(x_i, a)$$
$$f_2(x) = \sum_{a \in A} \max_{i=1,\dots,k} \sum_{j \neq i} \varphi_j(x_j, a).$$
Both f_1 and f_2 are convex functions. The pair $DF(x) = (\partial f_1(x), -\partial f_2(x))$ is a quasidifferential [11] of F at a point x. Here ∂f stands for the convex subdifferential of a convex function f.

2. Since F is DC, it follows that this function is locally Lipschitz.

3. Since F is DC it follows that this function is semi-smooth.

We can use quasidifferentials of a function $F \in \mathcal{F}$ for a local approximation of this function near a point x. Clarke subdifferential also can be used for local approximation of F, since F is locally Lipschitz.

3 Examples

We now give some examples of functions belonging to class $\mathcal{F}$. In all the examples, datasets are denoted as finite sets $A \subset \mathbb{R}^n$, that is as sets of n-dimensional points (also denoted observations).

3.1 Cluster functions and generalized cluster functions

Assume that a finite set $A \subset \mathbb{R}^n$ consists of k clusters. Let $X = \{x_1, \dots, x_k\} \subset (\mathbb{R}^n)^k$. Consider the distance $d(X, a) = \min\{\|x_1 - a\|, \dots \|x_k - a\|\}$ between the set X and a point (*observation*) $a \in A$. (It is assumed that $\mathbb{R}^n$ is equipped with a norm $\| \cdot \|$.) The deviation of X from A is the quantity $d(X, A) = \sum_{a \in A} d(X, a)$. Let $\bar{X} = \{\bar{x}_1, \dots \bar{x}_k\}$ be a solution to the problem:
$$\min_{x_1, \dots, x_k \in \mathbb{R}^n} \sum_{a \in A} \min\{\|x_1 - a\|, \dots \|x_k - a\|\}.$$

Then $\bar{x}_1, \dots, \bar{x}_k$ can be considered as the centres of required clusters. (It is implicitly assumed that these are point-centred clusters.) *If the cluster centres are known each point is assigned to the cluster with the nearest centre.* Assume that N is the cardinality of set A. The function

$$C_k(x_1, \dots, x_k) \equiv \frac{1}{N} d(X, A) = \frac{1}{N} \sum_{a \in A} \min(\|x_1 - a\|, \dots, \|x_k - a\|) \tag{2}$$

is called a cluster function. This function has the form (1) with $\varphi_i(x, a) = \|x - a\|$ for each $a \in A$ and $i = 1, \dots, k$. The cluster function was examined in [3]. Some numerical methods for its minimization were suggested in [3].

The cluster function has a saw-tooth form and the number of teeth drastically increases as the number of addends in (2) increases. This leads to the increase of the number of shallow local minima and saddle points. If the norm $\|\cdot\|$ is a polyhedral one, say $\|\cdot\| = \|\cdot\|_1$, then the cluster function is piece-wise linear with a very large number of different linear pieces. The restriction of the cluster function to a one-dimensional line has the form of a saw with a huge amount of teeth of different size but of the same slope.

Let $(m_a)_{a \in A}$ be a family of positive numbers. Function

$$\tilde{C}_k(x_1, \dots, x_k) = \frac{1}{N} \sum_{a \in A} m_a \min(\|x_1 - a\|, \dots, \|x_k - a\|) \tag{3}$$

is called a *generalized cluster function*. Clearly $\tilde{C}_k$ has the form (1). The structure of this function is similar to the structure of cluster function, however different teeth of generalized cluster function can have different slopes.

Clusters constructed according to centres, obtained as a result of the cluster function minimization are called *centre-based clusters*.

3.2 Bradley-Mangasarian approximation of a finite set

If a finite set A consists of flat parts it can be approximated by a collection of hyperplanes. Such kind of approximation was suggested by P.S. Bradley and O.L. Mangasarian [8]. Assume that we are looking for a collection of k hyperplanes $H_i = \{x : [l_i, x] = c_i\}$ approximating the set A. (Here $[l, x]$ stands for the inner product of vectors l and x.) The following optimization problem was considered in [8]:

$$\text{minimize } \sum_{a \in A} \min_{i=1,\dots,k} ([l_i, a] - c_i)^2 \text{ subject to } \|l_i\|_2 = 1, \quad i = 1, \dots, k. \tag{4}$$

Here $\min_{i=1,\dots,k}([l_i, a] - c_i)^2$ is the square of 2-norm distance between a point a and the nearest hyperplane from the given collection. Function

$$G((l_1, c_1), \dots, (l_k, c_k)) = \sum_{a \in A} \min_{i=1,\dots,k} ([l_i, a] - c_i)^2$$

can be represented in the form (1):

$$G((l_1,c_1),\dots,(l_k,c_k)) = \sum_{a\in A} \min_{i=1\dots,k} \varphi((l_i,c_i),a),$$

where $\varphi((l,c),a) = ([l,a]-c)^2$.

3.3 Skeleton of a finite set of points

We now consider a version of Bradley-Mangasarian definition, where the distances to hyperplanes are used instead of the squares of these distances. Assume that $\mathbb{R}^n$ is equipped with a norm $\|\cdot\|$. Let A be a finite set of points. Consider vectors $l_1,\dots,l_k$ with $\|l_i\|_* \equiv \max_{\|x\|=1}[l,x] = 1$ and numbers c_i $(i=1,\dots,k)$. Let $H_i = \{x : [l_i,x] = c_i\}$ and $H = \cup_i H_i$. Then the distance between the set H_i and a point a is $d(a,H_i) = |[l_i,a]-c_i|$ and the distance between the set H and a is

$$d(a,H) = \min_i |[l_i,a]-c_i|. \tag{5}$$

The deviation of X from A is

$$\sum_{a\in A} d(a,H) \equiv \sum_{a\in A} \min_i |[l_i,a]-c_i|.$$

The function

$$L_k((l_1,c_1),\dots,(l_k,c_k)) = \sum_{a\in A} \min_i |[l_i,a]-c_i| \tag{6}$$

is of the form (1). Consider the following constrained *min-sum-min* problem

$$\min \sum_{a\in A} \min_i |[l_i,a]-c_i| \text{ subject to } \|l_j\| = 1,\ \ c_j \in \mathbb{R}\ (j=1,\dots,k) \tag{7}$$

A solution of this problem will be called a *k-skeleton* of the set A. The function in (7) is called the *skeleton function.*

More precisely, k-skeleton is the union of k hyperplanes $\{x : [l_i,x] = c_i\}$, where $((l_1,c_1),\dots,(l_k,c_k))$ is a solution of (7). *If the skeletons are known, each point is assigned to the cluster with the nearest skeleton.* It is difficult to find a global minimizer of (7), so sometimes we can consider the union of hyperplanes that is formed by a local solution of (7) as a skeleton.

Clusters constructed according to skeletons, obtained as a result of the skeleton function minimization are called *skeleton-based clusters.*

The concept of *shape* of a finite set of points was introduced and studied in [17]. By definition, the shape is a minimal (in a certain sense) ellipsoid, which contains the given set. A technique to find an ellipsoidal shape is then proposed in the same paper. In many instances the geometric characterization of a set A can be viewed as the intersection between its shape, describing its external boundary, and its skeleton, describing its internal aspect.

A comparative study of Bradley-Mangasarian approximation and skeletons was undertaken in [12]. It was shown there that skeletons are quite different from Bradley-Mangasarian approximation, even for simple sets.

3.4 Illustrative examples

We now give two illustrative examples.

Example 3.1 Consider the set depicted in Fig. 1

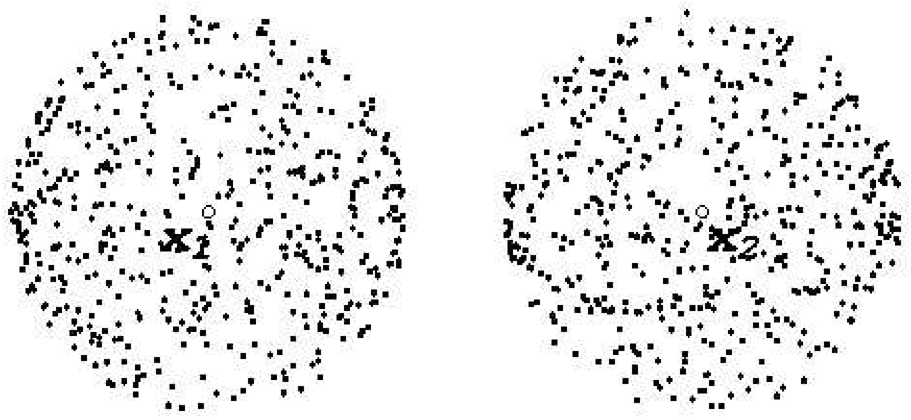

Figure 1: Clusters based on centres

Clearly this set consists of two clusters, the centers of these clusters (points x_1 and x_2) can be found by the minimization of the cluster function. The skeleton of this set hardly depends on the number k of hyperplanes (straight lines). For each k this skeleton cannot give a clear presentation on the structure of the set.

Example 3.2 Consider now the set depicted in Fig. 2.

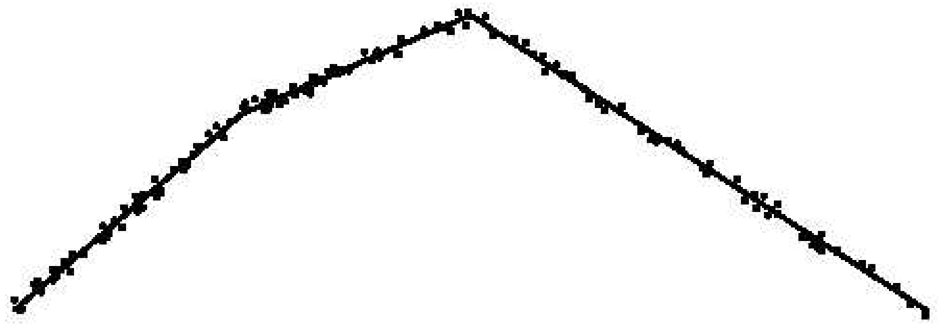

Figure 2: Clusters based on skeletons

It is difficult to say how many point-centred clusters has this set. Its description by means of such clusters cannot clarify its structure. At the same time this structure can be described by the intersection of its skeleton consisting on three straight lines and its shape. It does not make sense to consider k-skeletons of the given set with $k > 3$.

4 Minimization of sum-min functions belonging to class $\mathcal{F}$

Consider function F defined by (1):

$$F(x_1,\dots,x_k)=\frac{1}{N}\sum_{a\in A}\min(\varphi_1(x_1,a),\varphi_2(x_2,a),\dots\varphi_k(x_k,a)),\quad x_i\in\mathbb{R}^n,\ i=1,\dots,k.$$

where $A\subset\mathbb{R}^n$ is a finite set. This function depends on $n\times k$ variables. In real-world applications $n\times k$ is a large enough number and the set A contains some hundreds or thousands points. In such a case function F has a huge amount of shallow local minimizers that are very close to each other. The minimization of such functions is a challenging problem.

In this paper we consider both local and global minimization of sum-min functions from $\mathcal{F}$. First we discuss possible local techniques for the minimization.

The calculation of even one of the Clarke subgradients and/or a quasidifferential of function (1) is a difficult task, so methods of nonsmooth optimization based on subgradient information (quasidifferential information) at each iteration are not effective for the minimization of F. It seems that derivative-free methods are more effective for this purpose.

For the local minimization of functions (1) we propose to use the so-called discrete gradient (DG) method, which was introduced and studied by Adil Bagirov (see for example, [1]). A discrete gradient is a certain finite difference approximated the Clarke subgradient or a quasidifferential. In contrast with many other finite differences, the discrete gradient is defined with respect to a given direction. This leads to a good enough approximation of Clarke subgradients (quasidifferentials). DG calculates discrete gradients step-by-step; if a current point in hands is not an approximate stationary point then after a finite number of iterations the algorithm calculates a descent direction. Armijo's method is used in DG for a line search.

The calculation of discrete gradients is much easier if the number of addends in (1) is not very large. The decrease of the number of addends leads also to a drastic diminishing of the number of shallow local minima. Since the number of addends is equal to the number of points in the dataset, we conclude that the results of the application of DG for minimization of (1) significantly depend on the size of the set A.

The discrete gradient method is a local method, which may terminate in a local minimum. In order to ascertain the quality of the solution reached, it is necessary to apply global methods. Here we call global method a method that does not get trapped on stationary points, and can leave local minima to a better solution.

Various combinations between local and global techniques have recently been studied (see, for example [13, 18]). We use a combination of the DG and the cutting angle method (DG+CAM) in our experiments. We call this method the *hybrid global method*.

These two techniques (DG and DG+CAM) have been included in a new optimization software (CIAO-GO) created recently at the Centre for Informatics and Applied Optimization (CIAO) at the University of Ballarat, see [22] for more information. This version of the CIAO-GO software (Centre for Informatics and Applied Optimization-Global Optimization) allows one to use four different solvers

1. DG,
2. DG multi start,
3. DG+CAM,
4. DG+CAM multi start.

Working with this software users have to input

- an objective function (for minimization),
- an initial point for optimization,
- upper and lower bounds for variables,
- constraints and a penalty constant (in the case of constrained optimization), constraints can be represented as equalities and inequalities,
- maximal running time,
- maximal number of iterations.

"Multi start" option in CIAO-GO means that the program starts from the initial point chosen by a user and also generates 4 additional random initial points. The final result is the best obtained result. The additional initial points are generated by CIAO-GO from the corresponding feasible region (or close to the feasible region).

As a global optimization technique we use the General Algebraic Modeling System (GAMS), see [19] for more information. We use the Lipschitz global optimizer (LGO) solver [20] from Pinter Consulting Services [21].

5 Minimization of generalized cluster function

In this section we discuss applications of DG, DG+CAM and the LGO solver for minimization of generalized cluster functions. We propose several approaches for selecting initial points.

5.1 Construction of generalized cluster functions

Consider a set $A \subset \mathbb{R}^n$ that contains N points. Choose $\varepsilon > 0$. Then choose a random vector $b^1 \in A$ and consider subset $A_{b^1} = \{a \in A : \|a - b^1\| < \varepsilon\}$ of the set A. Take randomly a point $b^2 \in A_1 = A \setminus A_{b^1}$. Let $A_{b^2} = \{a \in A_1 : \|a - b^2\| < \varepsilon\}$ and $A_2 = A_1 \setminus A_{b^2}$. If the set A_{j-1} is known, take randomly $b^j \in A_{j-1}$, define set A_{b^j} as $\{a \in A_{j-1} : \|a - b^j\| < \varepsilon\}$ and define set A_j as $A_{j-1} \setminus A_{b^j}$. The result of the described procedure is the set $B = \{b^j\}_{j=1}^{N_B}$, which is a subset of the original dataset A. The vector b^j is a representative for the whole group of vectors, removed on the step j.

If m_j is the cardinality of A_{b^j} then the generalized cluster function corresponding to B

$$\tilde{C}_k(x^1, \dots, x^k) = \frac{1}{N} \sum_j m_j \min(\|x^1 - b^j\|, \dots, \|x^k - b^j\|)$$

can be used for finding centers of clusters of the set A.

The size of the dataset B obtained as the result of the described procedure is the most important parameter, so we shall use this parameter for characterization of B.

It can be proved (see [2]) that this function does not differ by more than ε from the original cluster function.

Remark 5.1 *We can use the same idea to construct the generalized skeleton function.*

Remark 5.2 *Unfortunately, it is very difficult to know a priori the value for ε which allows one to remove a certain proportion of observations. In our experiments we had to try several values for ε before we found suitable ones.*

5.2 Initial points

Most methods of local optimization are very sensitive to the choice of an initial point. In this section we suggest a choice of initial points which can be used for the minimization of cluster functions and generalized cluster functions.

Consider a set $A \subset \mathbb{R}^n$ that contains N points. Assume that we want to find k clusters in A. In this case an initial point is a vector $x \in \mathbb{R}^{n \times k}$. The structure of the problem under consideration leads to different approaches to the choice of initial points. We suggest the following four approaches.

k-meansL_1 initial point The k-meansL_1 method is a version of the well-known k-means method (see, for example, [16]), where $\| \cdot \|_1$ is used instead of $\| \cdot \|_2$. (We use $\| \cdot \|_1$ in numerical experiments, this is the reason for consideration of k-meansL_1 instead of k-means.) We use the following procedure in order to sort N observations into k clusters:

1. Take any k observations as the centres of the first k clusters.
2. Assign the remaining $N-k$ observations to one of the k clusters on the basis of the shortest distance (in the sense of $\| \cdot \|_1$ norm) between an observation and the mean of the cluster.
3. After each observation has been assigned to one of the k clusters, the means are recomputed (updated).

Stopping criterion: there is no observation, which moves from one cluster to another.

Note that results of this procedure depend on the choice of an initial observation.

We apply this algorithm for original dataset A and then the result point $x \in \mathbb{R}^{n\times k}$ is considered as an initial point for minimization of generalized cluster function generated by the dataset B.

Uniform initial point The application of optimization methods to clustering requires a certain data processing. In particular, a scaling procedure should be applied. In our experiments we convert a given dataset to a dataset with the mean-value 1 for each feature (coordinate). In such a case we can choose the point $x = (1, 1, \ldots, 1) \in \mathbb{R}^{n\times k}$ as initial one. We shall call it the *uniform initial point.*

Ordered initial point Recall that m_j indicates the cardinality of the set of points $A_{b^j} \in A$, which are represented by a point $b^j \in B$. It is natural to consider the collection of the heaviest k points as an initial vector for the minimization of generalized cluster function $\tilde{C}$. To formalize this, we rearrange the points so that the numbers $m_j, j = 1, \ldots, N_B$ decrease and take the first k points from this rearranged dataset. Thus, in order to construct an initial point we choose the k observations with the largest values for weights m_j from the dataset B.

Uniform-ordered initial point This initial point is a hybrid between the Uniform and the Ordered initial points. It contains the heaviest $k - 1$ observations and the barycentre (each coordinate is 1).

6 Numerical experiments with generalized cluster function

For numerical experiments we use two types of datasets, namely the original dataset A and a small dataset B obtained by the procedure described in Subsection 5.1. We compare results obtained for B with the results obtained for the entire original dataset A.

6.1 Datasets

We carried out numerical experiments with two well-known test datasets (see [16]):

- Letters dataset (20000 observations, 26 classes, 16 features). This dataset consists of samples of 26 capital letters, printed in different fonts; 20 different fonts were considered and the location of the samples was distributed randomly within the dataset.
- Pendigits dataset (10992 observations, 10 classes, 16 features). This dataset was created by collecting 250 samples from 44 writers. These writers are asked to write 250 digits in random order inside boxes of 500 by 500 tablet pixel resolution.

Both Letters and Pendigit datasets have been used for testing different methods of supervised classification (see [16] for details). Since we use these datasets only for construction of generalized cluster function, we consider them as datasets with unknown classes.

6.2 Numerical experiments: description

We are looking for three and four clusters in both Letters and Pendigits datasets. Dimension of optimization problems is equal to 48 in the case of 3 clusters and 64 in the case of 4 clusters. We consider two small sub-databases of the Letters dataset (Let1, 353 points, approximately 2% of the original dataset; and Let2, 810 points, approximately 4% of the original dataset) and two small sub-sets of the Pendigits dataset (Pen1, 216 points, approximately 2% of the original dataset; and Pen2, 426 points, approximately 4% of the original dataset).

We apply local techniques (discrete gradient method) and global techniques (a combination between discrete gradient and cutting angle method and the LGO solver) to the minimization of the generalized cluster function. Then we need to estimate the results obtained. We can use different approaches for this estimation. One of them is based on comparison of values of cluster function C_k constructed with respect to the centers obtained in the original dataset A and with respect to the centers obtained in its small sub-dataset B. We compare the cluster function values, started from different initial points in original datasets and their approximations. We use the following procedure.

Let A be an original dataset and B be its small sub-dataset. First, the centres of clusters in B should be found by an optimization technique. Then we evaluate the cluster function values in A using obtained points as the centers of clusters in A. Using this approach we can find out how the results of the minimization depend on initial points and how far we can go in the process of dataset reduction.

In our research we use 4 types of initial points, described in section 5.2. These initial points have been carefully chosen and the results obtained starting from these initial points are better than the results obtained starting from random initial points. Therefore, we present the results obtained for these 4 types of initial points rather than the results obtained starting from random initial points generated, for example, by ”multi start” option.

6.3 Results of numerical experiments

6.3.1 Local optimization

First of all we have to point out that we have two groups of initial points

- Group 1: Uniform initial point and k-meansL_1 initial point,
- Group 2: Ordered initial point and Uniform-ordered initial point.

Initial points from Group 1 are the same for an original dataset and for all its reduced versions. Initial points from Group 2 are constructed according to their weights. Points in original datasets have the same weights which are equal to 1.

Dataset	Size	Cluster function value	Generalized cluster function value	Cluster function value	Generalized cluster function value
		3 clusters		4 clusters	
Pen1	216	6.4225	5.5547	5.7962	4.8362
Pen2	426	6.3844	5.8132	5.7725	5.0931
Pendigits	10992	6.3426	6.3426	5.7218	5.7218
Let1	353	4.3059	3.3859	4.1200	3.1611
Let2	810	4.2826	3.7065	4.0906	3.5040
Letters	20000	4.2494	4.2494	4.0695	4.0695

Table 1: Cluster function and generalized cluster function: DG, Uniform initial point

Remark 6.1 *Because the weights can vary for different reductions of the dataset, the Ordered initial points for Let1 and Let2 do not necessarily coincide. The same is true for the Uniform-ordered initial points. The same observation applies to the Pendigits dataset and its reduced versions Pen1 and Pen2.*

Our next step is to compare results obtained starting from different initial points in the original datasets and in their approximations. In our experiments we use two different kinds of function: the cluster function and the generalized cluster function. Values for the cluster function and the generalized cluster function are the same for original datasets because each point has the same weight which is equal to 1. In the case of reduced datasets we produce our numerical experiments in corresponding approximations of original datasets and calculate two different value: the cluster function value and the generalized function value. The **cluster function value** is the value of the cluster function calculated in the corresponding original dataset according to the centres found in the reduced dataset. The **generalized cluster function value** is the value of the generalized cluster function calculated in the reduced dataset according to the centres found in the same reduced dataset. Normally a cluster function value (calculated according to the centres found reduced datasets) is larger than a generalized cluster function value calculated according to the same centres and the corresponding weights, because optimization techniques have been actually applied to minimize the generalized cluster in the corresponding reduced dataset. In Table 1-Table 2 we present the results of our numerical experiments obtained for DG and DG+CA starting from the Uniform initial point.

It is also very important to remember that *a better result in a reduced dataset is not necessarily better for the original one.* For example, in the case of the Pen1 dataset, 3 clusters, the Uniform initial point the generalized function value is lower for DG+CAM than for DG, however the cluster function value is lower for DG than for DG+CAM. We observe the same situation in some other examples.

Our actual goal is to find clusters in the original datasets, therefore it is important to compare *cluster function values calculated in original datasets* according to obtained centres. Centres can be obtained from our numerical experiments with both types of datasets: original datasets and reduced datasets. It is one of the possible ways to test the efficiency of the proposed approach:

Dataset	Size	Cluster function value	Generalized cluster function value	Cluster function value	Generalized cluster function value
		3 clusters		4 clusters	
Pen1	216	6.4254	5.5546	5.7943	4.8353
Pen2	426	6.3843	5.8131	5.7718	5.0931
Pendigits	10992	6.3426	6.3426	5.7218	5.7218
Let1	353	4.3059	3.3859	4.1208	3.1600
Let2	810	4.2828	3.7061	4.0909	3.5020
Letters	20000	4.2494	4.2494	4.0695	4.0695

Table 2: Cluster function and generalized cluster function: DG+CAM, Uniform initial point

Dataset	Size	Cluster function value 3 clusters	Cluster function value 4 clusters
Pen1	216	6.4272	5.8063
Pen2	426	6.3840	5.7704
Pendigits	10992	6.3409	5.7217
Let1	353	4.3087	4.1241
Let2	810	4.2816	4.1013
Letters	20000	4.2495	4.0726

Table 3: Cluster function: DG, k-meansL_1 initial point

substitution of original datasets by their smaller approximations.

Table 3-8 represent cluster function values obtained in our numerical experiments starting from the k-meansL_1, Ordered and Uniform-ordered initial point. We do not present the obtained generalized function values because this function can not be used as a measure of the quality of clustering.

Recall that reduced datasets are approximations of corresponding original datasets. Decreasing the number of observations we reduce the complexity of our optimization problems but the obtain less precise approximations. Therefore, our goal is to find some balance between the reduction of the complexity of optimization problems and the quality of obtained results. In some cases (mostly initial point from Group 2, see Remark 6.1 for more information) the results obtained on larger approximations of original datasets (more precise approximations) are worse than the results obtained on smaller approximations of original datasets (less precise approximations). For example, Pen1 and Pen2 for initial point from Group 2 (3 and 4 clusters).

Remark 6.2 *In the original datasets, it is not relevant to consider the Ordered and Uniform-ordered initial points, because all the points have the same weight.*

Summarizing the results of the numerical experiments (cluster function, local and hybrid global techniques, 4 special kinds of initial points) we can draw out the following conclusions:

Dataset	Size	Cluster function value 3 clusters	Cluster function value 4 clusters
Pen1	216	6.4278	5.8063
Pen2	426	6.3841	5.7723
Pendigits	10992	6.3409	5.7217
Let1	353	4.3087	4.1262
Let2	810	4.2824	4.1014
Letters	20000	4.2495	4.0726

Table 4: Cluster function: DG+CAM, k-meansL_1 initial point

Dataset	Size	Cluster function value 3 clusters	Cluster function value 4 clusters
Pen1	216	6.4188	5.8226
Pen2	426	6.6534	5.9047
Let1	353	4.3228	4.2049
Let2	810	4.3843	4.1112

Table 5: Cluster function: DG, Ordered initial point

Dataset	Size	Cluster function value 3 clusters	Cluster function value 4 clusters
Pen1	216	6.4171	5.8201
Pen2	426	6.6536	5.9047
Let1	353	4.3228	4.2045
Let2	810	4.3843	4.1107

Table 6: Cluster function: DG+CAM, Ordered initial point

Dataset	Size	Cluster function value 3 clusters	Cluster function value 4 clusters
Pen1	216	6.4188	5.7921
Pen2	426	6.6514	5.8718
Let1	353	4.2910	4.1225
Let2	810	4.2828	4.1129

Table 7: Cluster function: DG, Uniform-ordered initial point

Dataset	Size	Cluster function value 3 clusters	Cluster function value 4 clusters
Pen1	216	6.4171	5.7945
Pen2	426	6.6492	5.8715
Let1	353	4.2905	4.1233
Let2	810	4.2828	4.1130

Table 8: Cluster function: DG+CAM, Uniform-ordered initial point

1. DG and DG+CAM applied to the same datasets produce almost identical results if initial points are the same,
2. DG and DG+CAM applied to the same datasets starting from different initial points (4 proposed initial points) produce very similar results in most of the examples,
3. in some cases the results obtained on smaller approximations of original datasets are better than the results obtained on larger approximations of original datasets.

6.3.2 Global optimization: LGO solver

First we present the results obtained by the LGO solver (global optimization). We use the Uniform initial point. The results are in Table 9.

In almost all the cases (except Pendigits 3 clusters) the results for reduced datasets are better than for original datasets. It means that the cluster function is too complicate for the solver as an objective function and it is more efficient to use generalized cluster functions generated on reduced datasets. It is beneficial to use reduced datasets in the case of the LGO solver from two points of view

1. computations with reduced datasets allow one to reach a better minimizer;
2. computational time is significantly less for reduced datasets than for original datasets.

It is also obvious that the software failed to reach a global minimum. We suggest that the LGO solver has been developed for a broad class of optimization problems. However, the solvers included in CIAO-GO are more efficient for minimization of the sum of minima of convex functions, especially if the number of components in sums is large.

Remark 6.3 *The LGO solver was not used in the experiments on skeletons.*

Dataset	Size	Cluster function value 3 clusters	Cluster function value 4 clusters
Pen1	216	6.4370	5.8029
Pen2	426	6.4122	5.7800
Pendigits	10992	6.3426	7.1859
Let1	353	4.3076	4.1426
Let2	810	4.2829	4.1191
Letters	20000	5.8638	4.2064

Table 9: Cluster function: LGO solver

7 Skeletons

7.1 Introduction

The problem of grouping (clustering) points by means of skeletons is not so widely studied as it is in the case of cluster function based models. Therefore, we would like to start with some examples produced in not very large datasets (no more than 1000 observations). In this subsection we formulate the problems of finding skeletons mathematically, discuss applications of DG and DG+SA to finding skeletons with respect to $\|\cdot\|_1$ and and give graphical implementation to obtained results (for examples with no more than 3 features).

The search for skeletons can be done by solving constrained minimization problem (7).

Both algorithms are designed for unconstrained problems so we use a penalty function in order to convert problem (7) to the unconstrained minimization. The corresponding unconstrained problem has the form:

$$\min_{(l_1,b_1)\in\mathbb{R}^{n+1},\dots,(l_k,b_k)\in\mathbb{R}^{n+1}} \sum_{q\in Q} \min_i |[l_i, a^q] - b_i| + R_p \sum_{i=1}^{k} |\|l_i\|_1 - 1|, \tag{8}$$

where R_p is a penalty parameter.

Finally, the algorithms were applied starting from 3 different initial points, and the best solution found was selected. The 3 different points used in the example are:

- $P_1 = \begin{cases} l_i = \frac{1}{\sqrt{N}}(1,\dots,1) \\ b_i = 1 \end{cases}$
- $P_2 = \begin{cases} l_i = (1,0,..,0) \\ b_i = 1 \end{cases}$
- $P_3 = \begin{cases} l_i = \frac{1}{\sqrt{N-1}}(0,1\dots,1) \\ b_i = 1 \end{cases}$

The problem has been solved for different sets of points, selected from 3 different well known datasets: the Heart disease database (13 features, 2 classes: 160 observations are from the first class and 197 observations are from the second class), the Diabetes database (8 features, 2 classes: 500 observations are from the first class and 268 observations are from the second class) and the Australian credit cards database (14 features, 2 classes: 383 observations are from the first class and 307 observations are from the second class), see also [16] and references therein. Each of these datasets was submitted first to the feature selection method described in [3].

The value of the objective function was considerably decreased by both methods. However, the discrete gradient method often gives a local solution which is very close to the initial point, while the hybrid gives a solution which is further and better. In the tables the distance considered is the Euclidean distance between the solution obtained and the initial solution, and the value considered is the value of the objective function at this solution.

		DG method		hybrid method	
	Initial point	value	distance	value	distance
Class 1	1	22.9804	10.668	6.11298	7.98738
	2	25.5102	2.81543	13.2263	5.91397
	3	6.10334	4.40741	6.10334	4.40741
Class 2	1	0.473317	5.00549	0.473317	5.00549
	2	3.029	2.14784	0.222154	2.13944
	3	6.87897	6.06736	4.73828	6.74424
computation time		54 sec		664 sec	

Table 10: Australian credit card database with 2 hyperplanes skeletons

		DG method		hybrid method	
	Initial point	value	distance	value	distance
Class 1	1	28.5856	6.78624	28.1024	6.79326
	2	39.3925	11.4668	28.2417	11.7711
	3	33.2006	3.09434	31.4624	2.31922
Class 2	1	22.2806	2.3755	22.2806	2.3755
	2	30.346	56.7222	19.5574	8.76914
	3	23.0529	1.61649	22.9495	1.76052
computation time		212 sec		1521 sec	

Table 11: Diabetes database with 3 hyperplanes skeletons

The different examples show that although sometimes the hybrid method does not improve the result obtained with the discrete gradient method, in some other cases the result obtained is much better than when the discrete gradient method is used. However the computations times it induces are much greater than the simple use of the discrete gradient method. The diabetes dataset has 3 features, after feature selection (see [3]). This allows us to plot graphically some of the results obtained during the computations.

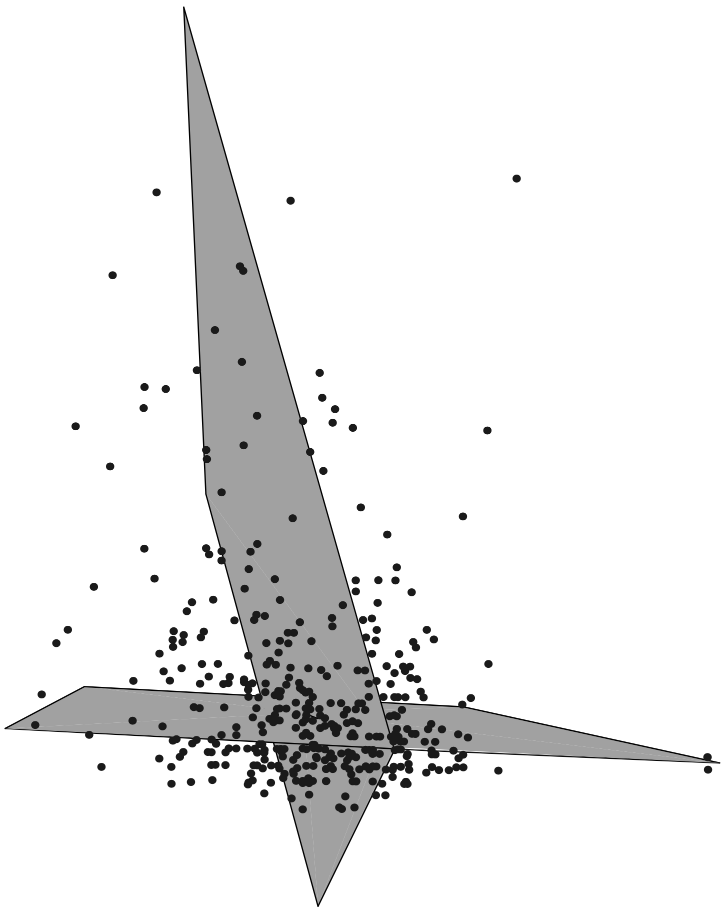

Figure 3: 2^{nd} class for the diabetes database, with 2 hyperplanes

We can observe that the hybrid method does not necessarily give an optimal solution. Even with the hybrid method the initial point is very important. The figure 3 however, confirms that the solutions obtained are usually very good, and represent correctly the set of points. The set of points studied here is constituted by a big mass of points, and some other points spread around. It is interesting to remark that the hyperplanes intersect around the same place - where the big mass is situated - and take different directions, to be the closer possible to the spread points.

The figure 4 shows the complexity of the diabetes dataset.

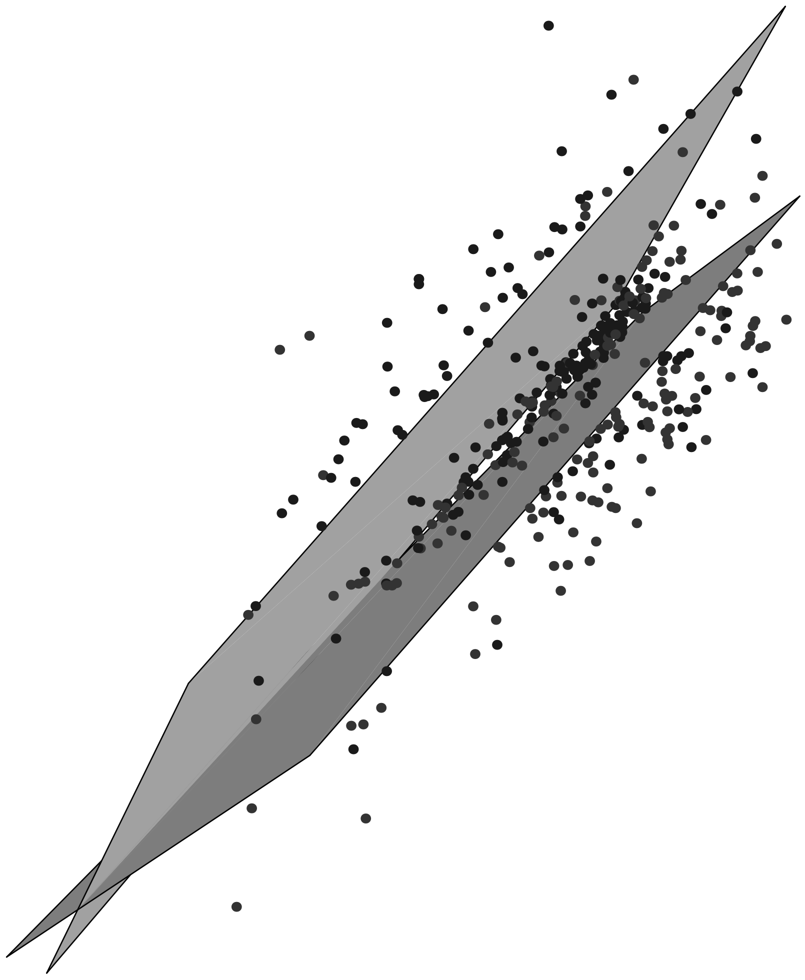

Figure 4: Diabetes database, with 1 hyperplane per class

Number of seletons	Dataset	Size	Skeleton function values			
			DG	DGMULT	DG+CAM	DG+CAMMULT
3	Pen1	216	2137.00	1287.58	1832.97	1320.00
	Pen2	426	735.00	735.47	735.47	735.47
	Pendigits	10992	567.20	567.20	567.20	566.55
4	Pen1	216	1223.16	1315.68	1194.65	1180.79
	Pen2	426	1360.16	946.74	1322.46	946.74
	Pendigits	10992	905.56	905.56	905.56	661.84

Table 12: Skeleton function: Pendigits

7.2 Numerical experiments: description

We are looking for three and four clusters in both Letters and Pendigits datasets. Dimension of optimization problems is equal to 51 in the case of 3 skeletons and 68 in the case of 4 skeletons. We use the same sub-datasets as in section 6 (Pen1, Pen2, Let1, Let2).

We apply local techniques (DG and DG+CAM) for minimization of the generalized skeleton function. Then we use a procedure which is similar to the one we use for the cluster function to estimate the obtained results. First, we find skeletons in original datasets (or in reduced datasets). Then we evaluate the skeleton function values in original datasets using the obtained skeletons.

For the skeleton function the problem of constructing a good initial point has not been studied yet. Therefore, in our numerical experiments as an initial point we choose a feasible point. We also use ”multi start” option to compare results obtained starting from different initial points.

7.3 Numerical experiments: results

In this subsection we present the results obtained for the skeleton function. Our goal is to find the centres in original datasets, therefore *we do not present the generalized skeleton function values*. Table 12 and Table 13 present the values of the skeleton function evaluated in the corresponding *original datasets* (Pendigits and Letters respectively) according to the skeletons obtained as optimization results reached in datasets from the first column of the tables. We use two different optimization methods: DG and DG+CAM and two different types of initial points: ”single start” (DG or DG+CAM) and ”multi start” (DGMULT or DG+CAMMULT).

The most important conclusion to the results is that in the case of the skeleton function the best optimization results (the lowest value of the skeleton function) *have been reached in the experiments with the original datasets*. It means that the *proposed cleaning procedure is not as efficient in the case of skeleton function as it is in the case of the clustering function*. However, in the case of the clustering function the initial points for the optimization methods have been chosen after some preliminary study. It can happen that an efficient choice of initial points leads to better optimization results for both kinds of datasets: original and reduced.

Number of seletons	Dataset	Size	Skeleton function values			
			DG	DGMULT	DG+CAM	DG+CAMMULT
3	Let1	353	1548.30	1548.30	1545.58	1545.58
	Let2	810	2201.75	1475.77	2171.01	1608.14
	Letters	20000	1904.71	1904.71	1904.71	964.37
4	Let1	353	1566.69	1566.69	1531.99	1531.99
	Let2	810	2030.20	2030.20	1892.31	1892.31
	Letters	20000	964.37	850.14	850.14	850.14

Table 13: Skeleton function: Letters

Recall that (7) is a constrained optimization problem with equality constraints. This problem is equivalent to the following constrained optimization problem with inequality constraints

$$\min \sum_{a \in A} \min_i |[l_i, a] - c_i| \text{ subject to } \|l_j\| \geq 1, \;\; c_j \in \mathbb{R} \; (j = 1, \ldots, k). \tag{9}$$

In our numerical experiments we use both formulations (7) and (9). In most of the experiments the results obtained for (7) are better than for (9) but computational time is much higher for (7) than for (9). It is recommended, however, to use the formulation (9) if, for example, experiments with (7) produce empty skeletons.

7.4 Other experiments

Another set of numerical experiments has been carried out on the both objective functions. Although of little interest from the point of view of the optimization itself, to the authors' opinion it may bring some more light on the clustering part.

The objective functions (2) and (7) has been minimized using two different methods: the discrete gradient method described above, and a hybrid method between the DG method and the well known simulated annealing method. This command is described with details in [6].

The basic idea of the hybrid method is to alternate the descent method to obtain a local minima and the simulated annealing method to escape this minimum. This reduces drastically the dependency of the local method on an initial point, and ensures that the method reaches a "good" minimum.

Numerical experiments were carried out on the Pendigit and Letters datasets for the generalized cluster function using different size dataset approximations. The results have shown that the hybrid method reached a sensibly comparable value as the other methods, although the algorithm had to leave up to 50 local minima. This can be explained by the large number of local minima in the objective function, each close to one another.

The skeleton function was minimized for the Heart Disease and the Diabetes datasets. The same behaviour can be observed. As the results of these experiments were not drawing any major conclusion, they are not shown here.

Numerical experiments have shown that while considerably faster than the simulated annealing method, the hybrid method is still fairly slow to converge.

8 Conclusions

8.1 Optimization

In this paper, a particular type of optimization problems has been presented. The objective function of these problems is the sum of mins of convex functions. This type of problems appears quite often in the area of data analysis, and two examples have been solved.

The generalized cluster function has been minimized for two datasets, using three different methods: the LGO global optimization software included in GAMS, the discrete gradient method and a combination between this method and the cutting angle method.

The last two methods, have been started from carefully selected initial points and from a random initial point.

The LGO software failed most of the time to reach even a good solution. This is due to the fact that the objective function has a very complex structure. This method was limited in time, and may have reached the global solution, had it been given a limitless amount of time.

Similarly, the local methods failed to reach the solution when started from a random point. The reason is the large amount of local minima in the objective function which prevent local methods to reach a good solution.

However the discrete gradient method, for all the examples, reached a good solution for at least one of the initial point. The combination reached a good solution for all of the initial points.

This shows that for such types of functions, presenting a complex structure and many local minima, most global methods will fail. However, well chosen initial points will lead to a deep local minimum. Because the local methods are much faster than global ones, it is more advantageous to start the local method from a set of carefully chosen initial points to reach a global minimum.

The application of the combination between the discrete gradient and the cutting angle methods appears to be a good alternative, as it is not very dependant on the initial point, while reaching a good solution in a limited time.

The second set of experiments was carried out over the hyperplanes function. This function having been less studied in the literature, it is harder to draw definite conclusions. However, the experiments show very clearly that the local methods once again strongly depend on the initial point. Unfortunately it is harder to devise a good initial point for this objective function.

8.2 Clustering

From the clustering point of view, two different similarity functions have been minimized. The first one is a variation of the widely studied cluster function, where the points are weighted. The second one is a variation of the Bradley-Mangasarian function, where distances from the hyperplanes are taken instead of their square.

A method for reducing the size of the dataset, ε-cleaning, has been devised and applied. Different values for epsilon lead to different sizes of datasets. Numerical experiments have been carried out for different values of epsilon, leading to very small (2% and 4%) datasets.

For the generalized cluster function, this method proves to be very successful: even for very small datasets, the function value obtained is very satisfactory. When the method was solved using the global method LGO, the results obtained for the reduced dataset were almost always better than those obtained for the original dataset. The reason is that the larger the dataset, the larger number of local minima for the objective function. When the dataset is reduced, what is lost in measurement quality is gained by the strong simplification of the function. Because each point in the reduced dataset acts already as a centre for its neighbourhood, minimizing the generalized cluster function is equivalent to group these “mini” clusters into larger clusters.

It has to be noted that there is not a monotone correspondence between the value of the generalized cluster function for the reduced and the original dataset. It may happen that a given solution is better than another one for the reduced dataset, and worse for the original. Thus we cannot conclude that the solution can be reached for the reduced dataset. However, the experiments show that the solution found for the reduced dataset is always good.

For the skeletons function, however, this method is not so successful. Although this has to be taken with precautions, as the initial points for this function could not be devised so carefully as for the cluster function, one can expect such behavior: the reduced dataset is actually a set of cluster centres. The skeleton approach is based on the assumption that the clusters in the dataset can be represented by hyperplanes, while the cluster approach assumes that the clusters are represented by centres.

The experiments show the significance of the choice of the initial point to reach good clusters. While random points did not allow any method to reach a good solution, all initial points selected upon the structure of the dataset lead the combination DG-CAM to the solution.

Since for the cluster function we are able to provide some good initial points, but not for the skeleton function, unless the structure of the dataset is known to correspond to some skeletons, we would recommend to use the centre approach.

Finally, the comparison between the results obtained by the two different methods has to be relativized: experiments having shown the importance of initial points, it is difficult to draw definitive conclusions from the results obtained for the skeleton approach.

However, there seems to be a relationship between the classes and the clusters obtained by both approaches, some classes being almost absent from certain clusters. Further investigations should be carried out in this direction, and classification processes based on these approaches could be proposed.

Acknowledgements

The authors are very thankful to Dr. Adil Bagirov for his valuable comments.

References

[1] A.M. Bagirov, Derivative-free methods for unconstrained nonsmooth optimization and its numerical analysis, *Investigacao Operacional* **19,** 75-93, 1999.

[2] A.M. Bagirov, A. M. Rubinov, N. Soukhoroukova, J. Yearwood, Unsupervised and Supervised Data Classification Via Nonsmooth and Global Optimization, Sociedad de Estadistica e Investigacion Operativa, Top 2003, Vol. 11, N1, pp. 1-93.

[3] A. M. Bagirov, A.M. Rubinov and J. Yearwood, A global optimization approach to classification, *Optimization and Engineering* **3**, 129-155, 2002

[4] A. Bagirov, A. Rubinov and Jiapu Zhang, Local Optimization Method with Global Multidimensional Search for descent, accepted for Journal of Global Optimization, alternatively, can be downloaded
http://www.optimization-online.org/DB_FILE/2004/01/808.pdf

[5] A. Bagirov, A. Rubinov and J. Zhang, A new multidimensional descent method for global optimization, submitted to Computational Optimization and Applications.

[6] A. M. Bagirov and J. Zhang, Hybrid simulating annealing method and discrete gradient method for global optimization, Proceedings of Industrial Mathematics Symposium, Perth, 2003

[7] G. Beliakov, A. Bagirov and J. E. Monsalve, Parallelization of the discrete gradient method of non-smooth optimization and its applications, *Proceedings of the 3rd International Conference on Computational Science*, Springer-Verlag, Heidelberg, vol. 3, 592-601, 2003.

[8] P.S. Bradley and O.L. Mangasarian, k-Plane clustering, Journal of Global Optimization, vol. 16, 23-32, 2000.

[9] J. Brimberg, R.F. Love and A. Mehrez, Location/Allocation of queuing facilities in continuous space using minsum and minmax criteria, in *Combinatorial and Global Optimization*, P.Pardalos, A. Migdalas and R. Burkard *eds*, World Scientific, 2002

[10] Clarke, F., *Nonsmooth Analysis and Optimization*, Wiley, New York, 1983.

[11] V. Demyanov and A. Rubinov, Constructive Nonsmooth Analysis, Peter Lang, 1995

[12] R. Ghosh, A.M. Rubinov and J.Zhang, Optimisation approach for clustering datasets with weights, Optimization Methods and Software, Vol. 20, 2006.,

[13] A.-R. Hedar and M. Fukushima, Hybrid simulated annealing and direct search method for nonlinear unconstrained global optimization, Optimization Methods and Software, Vol. 17, No. 5, 2002, pp. 891-912.

[14] A.K. Jain, M.N. Murty and P.J. Flynn,*Data Clustering: A review* ACM Computing Surveys, Vol 31, No 3, pp 264-323, 1999.

[15] C.T. Kelly, Detection and remediatio of stagnation in the Nelder-Mead algorithm using a sufficient decreasing condition, SIAM J. Optimization, 10, 43-55.

[16] D. Michie, D.J. Spiegelhalter and C.C. Taylor (eds.), *Machine Learning, Neural and Statistical Classification.* Ellis Horwood Series in Artificial Intelligence, London, 1994.

[17] N. Soukhoroukova, J. Ugon, A new algorithm to find a shape of a finite set of points submitted to *Proceedings of Conference on Industrial Optimization, Perth, Australia.*

[18] K.F.C. Yiu, Y. Liu and K.L. Teo, A Hybrid Descent Method for Global Optimization, Journal of Global Optimization, Vol. 11, 2003, to appear.

[19] http://www.gams.com/

[20] http://www.gams.com/solvers/lgo.pdf

[21] http://www.dal.ca/ jdpinter/

[22] http://www.ciao-go.com.au/index.php