# A new global optimization method for a symmetric Lipschitz continuous function and the application to searching for a globally optimal partition of the set $\mathcal{A} \subset \mathbb{R}$

*Rudolf Scitovski*[1]
*Department of Mathematics, University of Osijek*
*Trg Ljudevita Gaja 6, HR – 31 000 Osijek, Croatia*
*e-mail:* `scitowsk@mathos.hr`

**Abstract.** In this paper, we consider a global optimization problem for a symmetric Lipschitz continuous function $g\colon [a,b]^k \to \mathbb{R}$, whose domain $[a,b]^k \subset \mathbb{R}^k$ consists of $k!$ hypertetrahedrons of the same size and shape in which function $g$ attains equal values. A global minimum can therefore be searched for in one hypertetrahedron only, but then this becomes a global optimization problem with linear constraints. Apart from that, some known global optimization algorithms in standard form cannot be applied to solve the problem. In this paper, it is shown how this global optimization problem with linear constraints can easily be transformed into a global optimization problem on hypercube $[0,1]^k$ for the solving of which an applied `DIRECT` algorithm in standard form is possible. This approach has a somewhat lower efficiency than known global optimization methods for symmetric Lipschitz continuous functions (such as `SymDIRECT` or `DISIMPL`), but, on the other hand, this method allows for the use of publicly available and well developed computer codes for solving a global optimization problem on hypercube $[0,1]^k$ (e.g. `the DIRECT` algorithm). The method is illustrated and tested on standard symmetric functions and very demanding center-based clustering problems for the data that have only one feature. An application to the image segmentation problem is also shown.

**Key words:** symmetric function; Lipschitz continuous function; global optimization; DIRECT; SymDIRECT; DISIMPL; center-based clustering

**MSC2010:** 65K05, 90C26, 90C27, 90C56, 90C57, 05E05

## 1 Introduction

A real symmetric function $g\colon [a,b]^k \to \mathbb{R}$, of $k$ variables is the one whose value at any $k$-tuple of arguments is the same as its value at any permutation of that $k$-tuple. These functions are often the subject of research in different applications [5, 11]. In this paper, we shall especially consider symmetric Lipschitz continuous functions that occur naturally, for example, in center-based clustering problems (see, e.g., [13, 16, 31]), whereby special

[1]Rudolf Scitovski, e-mail: `scitowsk@mathos.hr`, telephone number: ++385-31-224-800, fax number: ++385-31-224-801

importance is attached to searching for a globally optimal partition of the data that have only one feature.

If the function $g$ attains its global minimum on $[a,b]^k$, then generally there exist at least $k!$ points from $[a,b]^k$, where this global minimum is attained [11, 41]. However, instead of solving a global optimization problem (GOP) for the function $g$, without loss of generality, a simpler GOP for the function $f\colon [0,1]^k \to \mathbb{R}$

$$\underset{(c_1,\ldots,c_k)\in[0,1]^k}{\operatorname{argmin}} f(c_1,\ldots,c_k), \tag{1}$$

can be considered, where $f = g\circ\kappa$ and $\kappa\colon [0,1]^k \to [a,b]^k$, $\kappa(x) = Dx + u$, $D = \operatorname{diag}(b-a,\ldots,b-a) \in \mathbb{R}^{k\times k}$, $u = (a,\ldots,a) \in \mathbb{R}^k$.

The domain of a symmetric Lipschitz continuous function $f\colon [0,1]^k \to \mathbb{R}$ is the hypercube $[0,1]^k$, which consists of $k!$ hypertetrahedrons of the same size and shape. On each of them, function $f$ attains equal values. Therefore, it is sufficient to solve only GOP (1) on one of these hypertetrahedrons, e.g. it is sufficient to find the solution of the GOP

$$\underset{(c_1,\ldots,c_k)\in\Delta}{\operatorname{argmin}} f(c_1,\ldots,c_k), \tag{2}$$

where hypertetrahedron $\Delta$ is given by

$$\Delta = \{(c_1,\ldots,c_k) \in [0,1]^k\colon 1 \ge c_1 \ge \cdots \ge c_k \ge 0\}. \tag{3}$$

In this paper, we propose how GOPs (2)–(3) can be transformed into a GOP on $[0,1]^k$. After that, the application of some well-known global optimization method in standard form [6, 7, 12, 24, 32] (e.g. DIRECT) becomes possible.

The paper is organized as follows. A new method for solving a GOP for a symmetric Lipschitz continuous function is described in detail in Section 2. The method is illustrated and tested on a center-based clustering problem for the data $\mathcal{A} \subset [0,1]$ that have only one feature. In Section 3, it is shown that, with natural conditions on the data, this problem always has a solution which is attained in each of $k!$ hypertetrahedrons contained in hypercube $[0,1]^k$. Several numerical experiments are given in Section 4. Also, the method is illustrated and tested on the image segmentation problem. Finally, some conclusions are discussed in Section 5.

# 2 Lipschitz global optimization for a symmetric function

A direct application of known methods for solving GOP (1) for complex problems [6, 9, 17–19, 23, 36**?** –39] is difficult especially if the number of independent variables is somewhat greater. Because of the symmetry property of the minimizing function, it is sufficient to consider only GOPs (2)–(3). In this way, the area in which GOP (1) is solved has been reduced by $k!$ times, but it has now become a nonlinear GOP with linear constraints.

From the geometrical point of view, the domain of minimizing function in this case becomes a hypertetrahedron, and therefore known global optimization methods (e.g. `the DIRECT` (DIviding RECTangles) algorithm [15]) cannot be applied in standard form. In [11], an efficient modification of the well-known `DIRECT` method called `SymDIRECT` is proposed for solving this problem. The modification implies that, in the dividing process, only those hyperrectangles are taken into consideration that are completely or only partially contained in the region $\Delta$ given by (3). A still more effective method for solving a GOP for a symmetric Lipschitz continuous function are `DISIMPL` (DIviding SIMPLices) algorithms proposed in [24–27].

Naturally, one could question how `GOPs` (2)–(3) can be transformed to a `GOP` on the set $[0,1]^k$. In that case, for solving this problem, some known global optimization methods could be used in standard form.

## 2.1 An example

The concept of the method proposed in this paper will be illustrated on the following simple example.

**Example 1.** *Let $\mathcal{A}$ be the set obtained by Wolfram* Mathematica *[45]:*

```
In[1]:= m = 20; SeedRandom[13];
        A = RandomReal[{0, 1}, m]

Out[1]={0.456535, 0.868230, 0.704274, 0.795001, 0.040520, 0.957827, 0.008372, 0.251257,
        0.014313, 0.743946, 0.066294, 0.783009, 0.907372, 0.081007, 0.486618, 0.824774,
        0.684515, 0.063848, 0.086283, 0.658425}
```

*Elements of the set $\mathcal{A}$ will be denoted by $a_i$, $i = 1, \ldots, m$. The set $\mathcal{A}$ should be partitioned into two nonempty disjoint subsets $\pi_1, \pi_2$ by using least squares distance-like function $d(x,y) = (x-y)^2$ and by solving the center-based clustering problem (see e.g. [14, 16, 31, 34]*

$$\underset{(c_1,c_2)\in[0,1]^2}{\operatorname{argmin}} F(c_1,c_2), \qquad F(c_1,c_2) = \sum_{i=1}^{m} \min\{d(c_1,a_i), d(c_2,a_i)\}. \tag{4}$$

The function $F$ is a symmetric Lipschitz continuous function [11, 28, 29, 31], it does not have to be either convex or differentiable and it attains its global minimum in at least two different points

$$\underset{(c_1,c_2)\in[0,1]^2}{\operatorname{argmin}} F(c_1,c_2) = \{(c_1^\star, c_2^\star), (c_2^\star, c_1^\star)\},$$

where $c_1^\star = 0.0765$, $c_2^\star = 0.7392$, $F(c_1^\star, c_2^\star) = 0.3003$. Therefore, `GOP` (4) can be reduced to a `GOP` on triangle $\triangle(OAB) = \{(c_1,c_2) \in [0,1]^2 \colon 1 \geq c_1 \geq c_2 \geq 0\}$ (see Fig. 1a). Furthermore, by means of appropriate transformation, this problem can be transformed to a `GOP` on $[0,1]^2$ for a new function $\Phi$ defined on $[0,1]^2$.

For this purpose, let us define a linear operator $\mathcal{T} \colon \mathbb{R}^2 \to \mathbb{R}^2$ by $\mathcal{T}(e_1) = e_1$, $\mathcal{T}(e_2) = e_1 + e_2$, where $\{e_1, e_2\}$ is a standard orthonormal basis in $\mathbb{R}^2$. Operator $\mathcal{T}$ maps the square $\square(OABC)$ to rhombus $\mathfrak{R}(OAB'C')$ (see Fig. 1b).

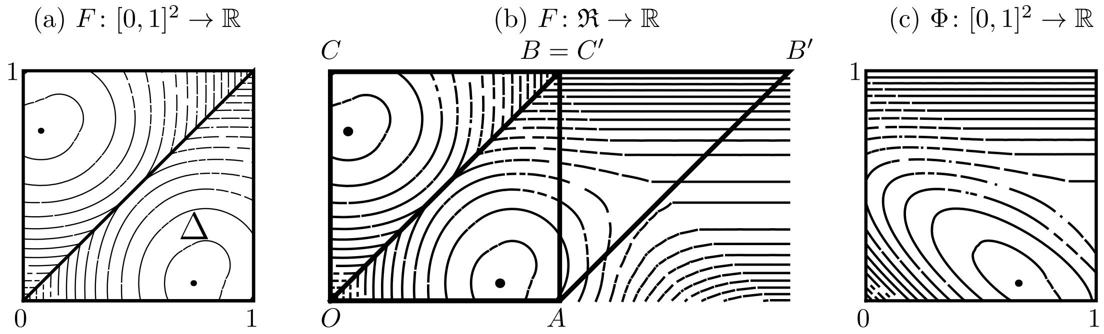


Figure 1: `ContourPlot`s of minimizing functions

Let $\Phi\colon [0,1]^2 \to \mathbb{R}$, $\Phi(\zeta_1,\zeta_2) := F(\mathcal{T}(\zeta_1,\zeta_2)) = F(\zeta_1+\zeta_2,\zeta_2)$ be a new function. The domain of the function $\Phi$ is a square $[0,1]^2$ and their `ContourPlot` can be seen in Fig. 1c. There holds (a more general assertion is shown in Section 3.1)

$$\min_{(c_1,c_2)\in\Delta} F(c_1,c_2) = \min_{(c_1,c_2)\in\mathfrak{R}} F(c_1,c_2) = \min_{(\zeta_1,\zeta_2)\in[0,1]^2} \Phi(\zeta_1,\zeta_2),$$

where $(c_1,c_2) = \mathcal{T}(\zeta_1,\zeta_2)$.

For the data set $\mathcal{A}$ from Example 1 by using the `DIRECT` algorithm in standard form, we obtain (see `http://www.mathos.unios.hr/images/homepages/scitowsk/DIRECT-2.nb`)

$$\begin{aligned} &(\zeta_1^\star,\zeta_2^\star) \in \operatorname*{argmin}_{(\zeta_1,\zeta_2)\in[0,1]^2} \Phi(\zeta_1,\zeta_2), \quad \text{where} \\ &\zeta_1^\star = 0.6627, \quad \zeta_2^\star = 0.0765, \qquad \Phi(\zeta_1^\star,\zeta_2^\star) = 0.3003, \end{aligned}$$

and finally

$$(c_1^\star,c_2^\star) = \mathcal{T}(\zeta_1^\star,\zeta_2^\star) = (\zeta_1^\star+\zeta_2^\star,\zeta_2^\star) = (0.7392, 0.0765).$$

## 2.2 A new method

The concept of the method proposed in this paper implies that, by virtue of a simple transformation, `GOP` (2) is observed on the hypercube $[0,1]^k$ instead of the set $\Delta$.

The following lemma shows how a hypertetrahedron $\Delta$ given by (3) can be obtained from the hyperrectangle $[0,1]^k$.

**Lemma 1.** *Let $\mathcal{T}\colon \mathbb{R}^k \to \mathbb{R}^k$ be a linear operator given by $\mathcal{T}(e_i) = \sum\limits_{j=1}^{i} e_j$, where $\{e_1,\dots,e_k\}$ is a standard orthonormal basis in $\mathbb{R}^k$. Then, there holds*

$$c \in \Delta \quad \Leftrightarrow \quad c\in[0,1]^k \quad \textit{and} \quad \mathcal{T}^{-1}(c)\in[0,1]^k \quad .$$

*Proof.* It is easy to verify that $\mathcal{T}^{-1}\colon \mathbb{R}^k \to \mathbb{R}^k$ is given by

$$\mathcal{T}^{-1}(e_1) = e_1, \quad \mathcal{T}^{-1}(e_i) = e_i - e_{i-1},\ i = 2,\dots,k, \tag{5}$$

and $\mathcal{T}^{-1}(c) = (c_1 - c_2, \ldots, c_{k-1} - c_k, c_k)$. Therefore, for all $c \in [0,1]^k$, such that $\mathcal{T}^{-1}(c) \in [0,1]^k$, there holds

$$\begin{aligned} &0 \le c_i \le 1, \quad i = 1, \ldots, k, \\ &0 \le c_i - c_{i+1} \le 1, \quad i = 1, \ldots, k-1, \end{aligned}$$

which is equivalent to

$$\begin{aligned} &0 \le c_i \le 1, \quad i = 1, \ldots, k, \\ &c_i \ge c_{i+1}, \quad i = 1, \ldots, k-1, \end{aligned}$$

i.e. to

$$1 \ge c_1 \ge c_2 \ge \cdots \ge c_k \ge 0. \qquad \square$$

The following theorem shows how `GOP` (2) can be transformed into a corresponding `GOP` on the hypercube $[0,1]^k$.

**Theorem 1.** *Let $f\colon [0,1]^k \to \mathbb{R}$ be a symmetric Lipschitz continuous function which attains its global minimum at the point $c^\star \in \Delta \subset [0,1]^k$, and let $\Phi\colon [0,1]^k \to \mathbb{R}$, $\Phi(\zeta) = f(\mathcal{T}(\zeta))$, where $\mathcal{T}\colon \mathbb{R}^k \to \mathbb{R}^k$ is a linear operator given by $\mathcal{T}(e_i) = \sum_{j=1}^{i} e_j$, where $\{e_1, \ldots, e_k\}$ is a standard orthonormal basis in $\mathbb{R}^k$. Then, there exists $\zeta^\star := \mathcal{T}^{-1}(c^\star) \in [0,1]^k$, in which the function $\Phi$ attains its global minimum and there holds*

$$\Phi(\zeta^\star) := \min_{\zeta \in [0,1]^k} \Phi(\zeta) = \min_{c \in \Delta} f(c) =: f(c^\star), \tag{6}$$

*where $c = \mathcal{T}(\zeta)$.*

*Proof.* For $c^\star = (c_1^\star, \ldots, c_k^\star) \in \Delta$, there holds

$$0 \le c_k^\star \le 1 \quad \text{and} \quad 0 \le c_i^\star - c_{i+1}^\star \le c_i^\star \le 1, \quad i = 1, \ldots, k-1,$$

and hence $\zeta^\star = \mathcal{T}^{-1}(c^\star) \in [0,1]^k$. Due to linearity of the operator $\mathcal{T}$, the function $\Phi$ at the point $\zeta^\star$ attains its global minimum. Furthermore, it follows that

$$\Phi(\zeta^\star) = \min_{\zeta \in [0,1]^k} \Phi(\zeta) = \min_{\zeta \in [0,1]^k} f(\mathcal{T}(\zeta)) = \min_{\mathcal{T}^{-1}(c) \in [0,1]^k} f(c).$$

Since the function $f$ attains its global minimum on the set $\Delta \subset [0,1]^k$, according to Lemma 1, there holds

$$\min_{\mathcal{T}^{-1}(c) \in [0,1]^k} f(c) = \min_{c \in \Delta} f(c) = f(c^\star). \qquad \square$$

Since from the domain $[0,1]^k$ of the minimizing symmetric function $f$ only one of $k!$ hypertetrahedrons is separated and the point of global minimum is searched for on this hypertetrahedron, the new global optimization method for solving `GOP` (1) for a symmetric Lipschitz continuous function $f$ will be called "*Separation Method*", and the corresponding algorithm will be called "`SepDIRECT`".

**Algorithm 1.** (`SepDIRECT`)

Step 1: For a symmetric Lipschitz continuous function $g\colon [a,b]^k \to \mathbb{R}$, define function $f\colon [0,1]^k \to \mathbb{R}$, such that $f = g \circ \kappa$, where $\kappa\colon [0,1]^k \to [a,b]^k$, $\kappa(x) = Dx + u$, $D = \operatorname{diag}(b-a,\ldots,b-a) \in \mathbb{R}^{k\times k}$, $u = (a,\ldots,a) \in \mathbb{R}^k$;

Step 2: For a symmetric Lipschitz continuous function $f\colon [0,1]^k \to \mathbb{R}$, define a new function $\Phi\colon [0,1]^k \to \mathbb{R}$, $\Phi(\zeta) := f(\mathcal{T}(\zeta))$, where $\mathcal{T}\colon \mathbb{R}^k \to \mathbb{R}^k$ is a linear operator given by $\mathcal{T}(e_i) = \sum\limits_{j=1}^{i} e_j$, where $\{e_1,\ldots,e_k\}$ is a standard orthonormal basis in $\mathbb{R}^k$;

Step 3: By using some global optimization method, solve `GOP`: $\operatorname*{argmin}\limits_{\zeta\in[0,1]^k} \Phi(\zeta) =: \zeta^\star$;

Step 4: A solution of the `GOP` $\operatorname*{argmin}\limits_{c\in[0,1]^k} f(c)$ is given by: $\hat{c} := \mathcal{T}(\zeta^\star)$ and a solution of $\operatorname*{argmin}\limits_{c\in[a,b]^k} g(c)$ is given by $c^\star = \kappa(\hat{c})$.

# 3 An application: A center-based clustering problem for the set $\mathcal{A} \subset \mathbb{R}$

An important and complex problem, which can be considered as a `GOP`, is a data clustering problem (see e.g. [1–3, 16, 21, 22, 30, 33, 35, 43, 44]). A partition of the set $\mathcal{A} = \{a_i \in \mathbb{R}^n\colon i = 1,\ldots,m\}$ into $1 \le k \le m$ disjoint subsets $\pi_1,\ldots,\pi_k$, such that

$$\bigcup_{i=1}^{k} \pi_i = \mathcal{A}, \qquad \pi_r \cap \pi_s = \emptyset, \quad r \neq s, \qquad |\pi_j| \ge 1, \quad j = 1,\ldots,k, \tag{7}$$

will be denoted by $\Pi = \{\pi_1,\ldots,\pi_k\}$ and the set of all such partitions by $\mathcal{P}(\mathcal{A},k)$.

Because of simplicity, this problem will be considered for the data that have only one feature, i.e., a center-based clustering problem for the set $\mathcal{A} \subset \mathbb{R}$ will be considered. Let us mention only two applications of this problem, i.e., the problem of determining spatial clusters of accidents along a continuous highway [13, 31] and the application to the image segmentation problem (see, e.g., [2, 4, 42]).

Furthermore, without loss of generality, let us suppose that $\mathcal{A} \subset [0,1] \subset \mathbb{R}$, $|\mathcal{A}| = m$ is a finite subset of real numbers.

If some distance-like function $d\colon \mathbb{R}\times\mathbb{R} \to \mathbb{R}_+$ is introduced [16, 30], then to each cluster $\pi_j \in \Pi$ we can associate its center $c_j^\star$, defined by

$$c_j^\star = c^\star(\pi_j) := \operatorname*{argmin}_{x\in[0,1]} \sum_{a_i\in\pi_j} d(x,a_i). \tag{8}$$

It is said that the partition $\Pi^\star \in \mathcal{P}(\mathcal{A},k)$ is a globally optimal $k$-partition if

$$\Pi^\star = \operatorname*{argmin}_{\Pi\in\mathcal{P}(\mathcal{A},k)} \mathcal{F}(\Pi), \qquad \mathcal{F}(\Pi) = \sum_{j=1}^{k} \sum_{a_i\in\pi_j} d(c_j,a_i), \tag{9}$$

where $\mathcal{F}\colon \mathcal{P}(\mathcal{A},k) \to \mathbb{R}_+$ is the objective function.

This problem can be formulated as a center-based clustering problem [14, 20, 34]

$$\underset{(c_1,\dots,c_k)\in\mathbb{R}^k}{\operatorname{argmin}} F(c_1,\dots,c_k), \qquad F(c_1,\dots,c_k) = \sum_{i=1}^{m} \min_{1\le j\le k} d(c_j,a_i). \tag{10}$$

The solutions of optimization problems (9) and (10) coincide [40].

## 3.1 Existence and solution localization

For a general problem (10), it will be shown that the objective function $F$ attains its global minimum on a unit hypercube $[0,1]^k$. It is sufficient to search for the function $F$ on the set $\Delta$ given by (3) due to its symmetry property.

First, the following theorem shows that the function $F$ given by (10) outside the hypercube $[0,1]^k$ does not achieve a lower value.

**Theorem 2.** *Let $\mathcal{A} \subset [0,1] \subset \mathbb{R}$ be a finite set of real numbers which should be grouped into $k$ disjoint nonempty subset-clusters. Then for each $(c_1,\dots,c_k) \in \mathbb{R}^k$ there exists $(\bar{c}_1,\dots,\bar{c}_k) \in [0,1]^k$, such that $F(\bar{c}_1,\dots,\bar{c}_k) \le F(c_1,\dots,c_k)$.*

*Proof.* The case when $(c_1,\dots,c_k) \in [0,1]^k$ is trivial. Therefore, let us assume that there exists at least one $c_j \notin [0,1]$. Let $(\bar{c}_1,\dots,\bar{c}_k) \in [0,1]^k$ be a point defined by

$$\bar{c}_j = \begin{cases} c_j, & \text{if } c_j \in [0,1] \\ 1, & \text{if } c_j > 1 \\ 0, & \text{if } c_j < 0. \end{cases}$$

There holds

$$d(\bar{c}_j,a) \le d(c_j,a), \quad \forall a \in \mathcal{A} \text{ and } \forall j = 1,\dots,k,$$

i.e., more precisely

$$\begin{aligned} d(\bar{c}_j,a) &= d(c_j,a), \quad \forall a \in \mathcal{A} \quad \& \quad c_j \in [0,1], \\ d(\bar{c}_j,a) &< d(c_j,a), \quad \forall a \in \mathcal{A} \quad \& \quad c_j \notin [0,1], \end{aligned}$$

from where it follows

$$\min_{1\le j\le k} d(\bar{c}_j,a) \le \min_{1\le j\le k} d(c_j,a), \quad \forall a \in \mathcal{A},$$

i.e., $F(\bar{c}_1,\dots,\bar{c}_k) \le F(c_1,\dots,c_k)$. □

The following theorem shows that, with natural conditions on the data, the objective function $F$ defined by (10) outside the hypercube $[0,1]^k$ always attains a strictly higher value than on the set $\Delta \subset [0,1]^k$.

**Theorem 3.** *Let $\mathcal{A} \subset [0,1] \subset \mathbb{R}$ be a finite set of real numbers which has at least $k$ mutually different elements, and which should be grouped into $k$ disjoint nonempty subset-clusters.*

*Then for each $(c_1,\dots,c_k) \in \mathbb{R}^k \setminus [0,1]^k$ there exists $(\hat{c}_1,\dots,\hat{c}_k) \in \Delta$, such that $F(\hat{c}_1,\dots,\hat{c}_k) < F(c_1,\dots,c_k)$.*

*Proof.* Let $(c_1,\dots,c_k) \in \mathbb{R}^k \setminus [0,1]^k$ be an arbitrary point. This means that there exists $s \in \{1,\dots,k\}$ such that $c_s \in \mathbb{R} \setminus [0,1]$. Let us define $(\hat{c}_1,\dots,\hat{c}_k) \in [0,1]^k$ in the following way

$$\hat{c}_j = \begin{cases} c_j, & \text{if } c_j \in [0,1] \\ a_{\min} = \min\{a \in \mathcal{A} \colon a \notin \{c_1,\dots,c_k\}\}, & \text{if } c_j < 0 \\ a_{\max} = \max\{a \in \mathcal{A} \colon a \notin \{c_1,\dots,c_k\}\}, & \text{if } c_j > 1 \end{cases}$$

Without loss of generality, let us suppose that $c_s > 1$ (as in Figure 2). Then

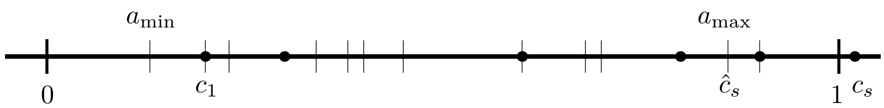


Figure 2: Data set $\mathcal{A}$ (dashes) and centers $\{c_1,\dots,c_k\}$ (dots)

(i) for $a = a_{\max}$ there holds $0 = d(\hat{c}_s,a) < d(c_j,a), \forall j = 1,\dots,k$ and therefore, $\min\limits_{1\le j\le k} d(\hat{c}_j,a) < \min\limits_{1\le j\le k} d(c_j,a)$;

(ii) for $a_{\min} \le a < a_{\max}$ there holds $d(\hat{c}_s,a) < d(c_s,a)$ and therefore, $\min\limits_{1\le j\le k} d(\hat{c}_j,a) \le \min\limits_{1\le j\le k} d(c_j,a)$;

(iii) for $a > a_{\max}$ or $a < a_{\min}$ there holds $\min\limits_{1\le j\le k} d(\hat{c}_j,a) = \min\limits_{1\le j\le k} d(c_j,a) = 0$.

Therefore, $F(\hat{c}_1,\dots,\hat{c}_k) < F(c_1,\dots,c_k)$ for each permutation of coordinates of the point $(\hat{c}_1,\dots,\hat{c}_k) \in [0,1]^k$, and consequently also for $1 \ge \hat{c}_{i_1} \ge \cdots \ge \hat{c}_{i_k} \ge 0$, i.e., for $(\hat{c}_{i_1},\dots\hat{c}_{i_k}) \in \Delta$. □

So, instead of solving a nonlinear `GOP` (10), it is sufficient to solve the following nonlinear `GOP` with linear constraints

$$\operatorname*{argmin}_{(c_1,\dots,c_k)\in\Delta} F(c_1,\dots,c_k), \qquad F(c_1,\dots,c_k) = \sum_{i=1}^{m} \min_{1\le j\le k} d(c_j,a_i), \tag{11}$$

where the set $\Delta$ is given by (3).

## 3.2 Searching for the globally optimal partition of the set $\mathcal{A} \subset \mathbb{R}$

Function $F$ defined in (11) is a symmetric Lipschitz continuous function, which, according to Theorem 2 and Theorem 3, attains its global minimum on the set $\Delta$ given by (3). Therefore, Theorem 1 can be applied.

**Corollary 1.** *Let $\mathcal{A} \subset \mathbb{R}$ be the set of real numbers and $d\colon \mathbb{R}\times\mathbb{R} \to \mathbb{R}_+$ some distance-like function and let the function $F$ given by* (11) *attain its global minimum at the point $c^\star \in \Delta$. Then it holds*

$$F(c^\star) = \min_{\zeta\in[0,1]^k} \Phi(\zeta) =: \Phi(\zeta^\star),$$

*where $\Phi(\zeta) = F(\mathcal{T}(\zeta))$, $\zeta^\star = \mathcal{T}^{-1}(c^\star)$ and $\mathcal{T}\colon \mathbb{R}^{k\times k} \to \mathbb{R}^{k\times k}$ is a linear operator given by $\mathcal{T}(e_i) = \sum\limits_{j=1}^{i} e_j$, where $\{e_1,\dots,e_k\}$ is a standard orthonormal basis in $\mathbb{R}^k$.*

Therefore, instead of solving the nonlinear `GOP` with linear constraints (11), the following `GOP`

$$\operatorname*{argmin}_{\zeta\in[0,1]^k} \Phi(\zeta), \qquad \Phi(\zeta) = F(\mathcal{T}(\zeta)) \tag{12}$$

can be solved on the hyperrectangle $[0,1]^k$.

The problem will be illustrated on Example 4 from paper [11]. In this example, in the corresponding set $\Delta$ there are at least two local minima of the function $F$.

**Example 2.** *The set of $m = 10$ uniform distributed random numbers from* $[0,1]$

$$\mathcal{A} = \{0.008,\ 0.014,\ 0.041,\ 0.251,\ 0.457,\ 0.704,\ 0.744,\ 0.795,\ 0.868,\ 0.958\}.$$

*should be partitioned into 3 clusters.*

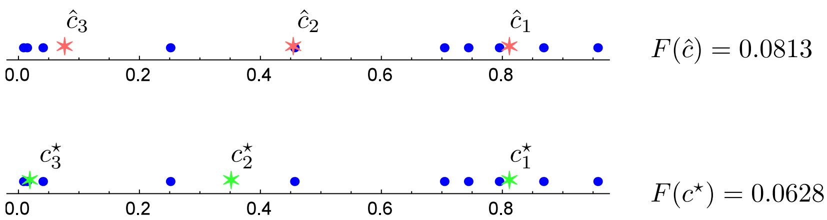


Figure 3: The data and the centers

In this case, the minimizing function $F$ attains local minima at points $\hat{c}$ and $c^\star$ (see also Fig. 3).

$$\begin{aligned}
&\hat{c} = (0.8138,\ 0.4570,\ 0.0785) \in \Delta, \qquad F(\hat{c}) = 0.0813,\\
&c^\star = (0.8138,\ 0.3540,\ 0.0210) \in \Delta, \qquad F(c^\star) = 0.0628,\\
&\text{where} \quad \Delta = \{(c_1,c_2,c_3) \in [0,1]^3 \colon 1 \ge c_1 \ge c_2 \ge c_3 \ge 0\}.
\end{aligned}$$

These points are obtained by applying the $k$-means algorithm and choosing appropriate initial approximations.

By applying the `DIRECT` algorithm to solving the corresponding `GOP` (11), we obtain the point $c^\star$. Application of the `DIRECT` algorithm to the corresponding problem (12) yields $\zeta^\star = (0.45991,\ 0.33280,\ 0.02111)$. It is easy to verify that $c^\star = \mathcal{T}(\zeta^\star)$.

# 4 Numerical experiments

The new `Separation Method` described in Section 2.2 searches for a solution of a `GOP` for symmetric Lipschitz continuous function and shows significantly better characteristics than known global optimization algorithms in standard form. At the same time, it is not expected that this method will show better performance than some special methods for solving a `GOP` for symmetric Lipschitz continuous function (`SymDIRECT` [15], `DISIMPL` [25]).

It should be emphasized that a direct application of the `DIRECT` algorithm for solving a GOP for a symmetric Lipschitz continuous function $F$ leads to searching for all $k!$ points of global minima of the symmetric function $F$. For example, in case of searching for a globally optimal $k$-partition of the set $\mathcal{A} \subset \mathbb{R}$, for a somewhat larger $k$, a standard `DIRECT` algorithm is very inefficient because the number of potentially optimal rectangles in the implementation of the algorithm becomes too large and the method is too slow. On the other hand, a direct application of the `DIRECT` algorithm to the hypertetrahedron $\Delta$ given by (3) is not possible.

Searching for a solution of the corresponding GOP in the hypertetrahedron $\Delta$ reduces the number of points searched for by $k!$ times. By applying the linear operator $\mathcal{T}$ given by (5), the hypertetrahedron $\Delta$ is transformed on the hypercube $[0, 1]^k$, and this allows for the direct use of known global optimization algorithms in standard form (for example, the `DIRECT` algorithm), for which there are publicly available and well elaborated computer codes (see, e.g., [7, 8, 10]). This is the main advantage of the new `Separation Method`.

Therefore, the new `Separation Method` described in Section 2.2 will be illustrated and compared with the `DIRECT` algorithm and the `SymDIRECT` algorithm on several examples based on the problem described in Section 3.1. The well known `DIRECT` algorithm is described well in [8, 15].

`SymDIRECT` algorithm [11] is a modification of the `DIRECT` algorithm for a symmetric Lipschitz continuous function. This means that in the procedure of dividing some hyperrectangle attention should be paid to the part of the region $[0, 1]^k$ it appears in. The following situations might occur:

(*i*) If the whole hyperrectangle is located in the region $\Delta$, all hyperrectangles that emerge by its division will also be contained in the region $\Delta$. All of them are also liable to further division.

(*ii*) If a hyperrectangle appears outside the region $\Delta$, the point of the global minimum we search for cannot appear therein. Hence such hyperrectangles shall not be divided further.

(*iii*) If a hyperrectangle lies in the region $\Delta$ only partially, some hyperrectangles that come into existence by its division can be fully contained in the region $\Delta$ (classified under case (i)), some might be completely outside the region $\Delta$ (classified under case (ii)), and some might lie only partially in the region $\Delta$ (again classified under case (iii)).

Sufficient conditions by which some hyperrectangle completely or partially lies in the region $\Delta$ are given in [11]. These conditions should be determined for all given cases, and they should be tested in the iterative procedure.

In this paper, all evaluations were carried out in *Mathematica*[2] and the new `Separation Method` in the examples listed below is implemented by using the `DIRECT` algorithm, although some other algorithms for global optimization might be used for that purpose.

**Example 3.** *Efficiency of all three aforementioned methods will be compared in terms of the necessary CPU-time and the number of function evaluations on the basis of synthetic data generated similarly to [43].*

*Ten experiments have been conducted for each* $k \in \{2, \ldots, 10\}$ *as follows. First,* $k$ *points* $z_1, \ldots, z_k$ *uniformly distributed in the interval are chosen, such that they mutually differ for at least* $\frac{1}{2k}$*. After that, in the neighborhood of each point* $z_j$ $\lfloor \frac{m}{k} \rfloor \pm \lfloor \frac{m}{4k} \rfloor$*, random real numbers from* $\mathcal{N}(z_j, \frac{1}{4k})$ *are generated, where* $m = 200$*. For the data point sets obtained in such way, we searched for a globally optimal partition*

*(i) by applying the* `DIRECT` *algorithm for solving* `GOP` (10)*;*

*(ii) by applying the* `SymDIRECT` *algorithm for solving* `GOP` (11)*;*

*(iii) by applying the* `SepDIRECT` *algorithm for solving* `GOP` (12) *by using the* `DIRECT` *algorithm.*

Average CPU-times and the corresponding number of function evaluations ($N_f$) for all three methods are shown in Table 1. Application of the `DIRECT` algorithm for solving `GOP` (10) searches all $k!$ points of global minima of the function $F$ and therefore, in the optimizing process, the number of potentially optimal rectangles becomes too large and the method too slow. Therefore, for $k > 7$, the method has not been tested.

| | $k=2$ | $k=3$ | $k=4$ | $k=5$ | $k=6$ | $k=7$ | $k=8$ | $k=9$ | $k=10$ |
|---|---|---|---|---|---|---|---|---|---|
| DIRECT($N_f$) | 100 | 320 | 847 | 4224 | 18338 | 130867 | - | - | - |
| SymDIRECT($N_f$) | 65 | 116 | 387 | 401 | 451 | 493 | 584 | 950 | 683 |
| SepDIRECT($N_f$) | 55 | 126 | 362 | 398 | 428 | 523 | 1574 | 2118 | 1831 |
| DIRECT(CPU) | 0.020 | 0.072 | 0.215 | 1.662 | 13.115 | 413.220 | - | - | - |
| SymDIRECT(CPU) | 0.017 | 0.039 | 0.062 | 0.131 | 0.184 | 0.264 | 0.317 | 2.765 | 4.248 |
| SepDIRECT(CPU) | 0.012 | 0.036 | 0.056 | 0.093 | 0.168 | 0.290 | 1.765 | 6.548 | 9.522 |

Table 1: Average CPU-time (in sec) and the number of function evaluations $N_f$.

Both the `SymDIRECT` algorithm and the new `SepDIRECT` algorithm require significantly shorter CPU-time and a smaller number of function evaluations than the application of

[2]Corresponding *Mathematica*–modules and testing of algorihms were done by Ivan Vazler, Department of Mathematics, University of Osijek, on the computer with a `2.66 GHz Intel(R) Core(TM)i5 CPU` with `4GB` of `RAM`. *Mathematica*–modules for `DIRECT` and `SymDIRECT` can be found at: `http://www.mathos.unios.hr/images/homepages/scitowsk/DIRECT-2.nb`

the `DIRECT` algorithm for solving `GOP` (10). By using ordinary *Mathematica*-codes available on the website `http://www.mathos.unios.hr/images/homepages/scitowsk/DIRECT-2.nb`, in case of solving complex problems ($k \geq 7$) the `SymDIRECT` algorithm shows higher efficiency. By using optimized computer codes for the standard `DIRECT` algorithm efficiency improvement of the new algorithm `SepDIRECT` could be expected.

**Example 4.** *Efficiency of all three aforementioned methods will be compared in terms of the necessary CPU-time and the number of function evaluations on the standard symmetric functions used in [11]. Corresponding* Mathematica-*codes can be found on the previously mentioned website.*

| Method | Grbić | | Alolyan | | Easom | | Rastrigin | | Shubert | |
|---|---|---|---|---|---|---|---|---|---|---|
| | $N_f$ | CPU | $N_f$ | CPU | $N_f$ | CPU | $N_f$ | CPU | $N_f$ | CPU |
| `DIRECT` | 321 | 0.078 | 249 | 0.063 | 865 | 0.594 | 257 | 0.031 | - | - |
| `SymDIRECT` | 173 | 0.047 | 141 | 0.047 | 521 | 0.328 | 177 | 0.047 | - | - |
| `SepDIRECT` | 130 | 0.047 | 228 | 0.047 | 840 | 0.328 | 242 | 0.047 | 116 | 0.046 |

Table 2: CPU-time (in sec) and the number of function evaluations $N_f$. `DIRECT` and `SymDIRECT` algorithms do not give a global minimum for the Shubert test function.

The results obtained in the previous example are confirmed: both the `SymDIRECT` algorithm and the new `SepDIRECT` algorithm require significantly shorter CPU-time and a smaller number of function evaluations than the application of the `DIRECT` algorithm for solving `GOP` (10). By using optimized computer codes for the standard `DIRECT` algorithm, significant efficiency improvement of the new algorithm `SepDIRECT` could be expected.

## 4.1 Image segmentation problem

The proposed `Separation Method` will be illustrated on the image segmentation problem (see, e.g., [2, 4, 42]). For example, a $512 \times 512$ grayscale test image “Elaine” (see Fig. 4a) will be segmented into $2, 4$ and 8 layers (clusters) based on image gray levels. Let $\mathcal{A} = \{a^i \in \mathbb{R} \colon i = 1, \ldots, 262\,144\}$ be the set (finite sequence) of real numbers which represent gray levels of the points of the $512 \times 512$ image “Elaine”.

The weight $w_i = 1$ is associated to each $a^i \in \mathcal{A}$ . The set $\mathcal{A}$ should be partitioned into $k = 2, 4$ or 8 clusters. To each cluster (which contains points with a similar gray level) we will associate its centroid, and then to all points of this cluster we will associate the gray level of that centroid. An optimal partition in this case can be considered as a compression of the original image. Images Fig. 4b-d represent optimal partitions with 2, 4 and 8 clusters (layers). Above each of those images, the size of their `eps` format is indicated. Under each image, a gray-level histogram with corresponding cluster centers (gray levels) is shown.

For example, if $k = 2$, the set $\mathcal{A}$ is partitioned into two clusters: cluster $\pi_1^\star$ with brighter points and center $c_1^\star = 0.39$, and cluster $\pi_2^\star$ with darker points and center $c_2^\star = 0.69$. After

(a) Original (366 KB)

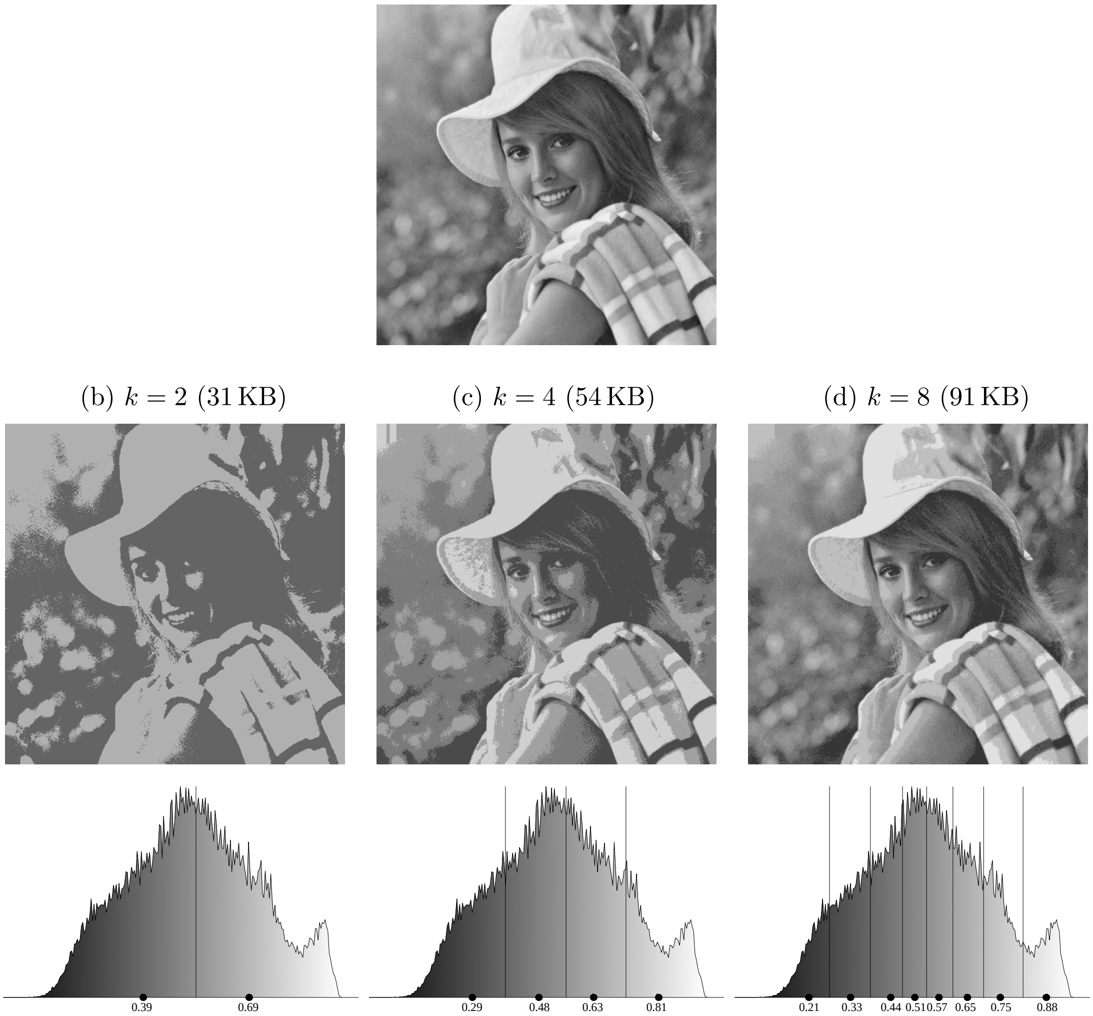


Figure 4: The original image and its segmentation into 2, 4 and 8 clusters

that, to each point of cluster $\pi_1^\star$ and to the each point of cluster $\pi_2^\star$ we will associate the gray level $c_1^\star$ and the gray level $c_2^\star$, respectively. In this way, we obtain Fig. 4b. Note that in this way we interpreted a grayscale image by a finite sequence of 262 144 members which can be only the numbers 0.39 or 0.69.

| | $k=2$ | | $k=4$ | | $k=8$ | |
|---|---|---|---|---|---|---|
| | Time | $N_f$ | Time | $N_f$ | Time | $N_f$ |
| `DIRECT` | 0 | 99 | 0.141 | 1 065 | 2:46:37 | 944 601 |
| `SymDIRECT` | 0.015 | 57 | 0.031 | 137 | 0.156 | 587 |
| `SepDIRECT` | 0.015 | 63 | 0.031 | 173 | 0.359 | 1 259 |

Table 3: CPU-time and the number of function evaluations $N_f$.

The necessary CPU-time for searching for an optimal partition with 2, 4 and 8 clusters by applying the `DIRECT` algorithm, the `SymDIRECT` algorithm and the `SepDIRECT` (by using the `DIRECT` algorithm) is very short (see Table 3).

# 5 Conclusions

Generally, if a symmetric Lipschitz continuous function $f\colon [a,b]^k \to \mathbb{R}$ attains its global minimum, then the set $\operatorname*{argmin}_{x\in[a,b]^k} f(x)$ has at least $k!$ different points. Therefore, a direct application of some of known global optimization methods is very inefficient. If the global minimum is searched for only in one of $k!$ equal hypertetrahedrons contained in the hypercube $[a,b]^k$, the problem is transformed into a `GOP` with linear constraints. Some known global optimization algorithms in standard form (e.g. `DIRECT`) cannot be applied to solving this problem. In this paper, it is shown how this `GOP` can be easily transformed into a `GOP` on a hypercube $[0,1]^k$. In this way, the new Separation Method is constructed and the corresponding minimizing function $\Phi$ has $k!$ times less points of the global minimum, so that the application of known global optimization methods becomes significantly more efficient. However, the new Separation Method is primarily theoretically important, although its numerical performance does not lag significantly behind known global optimization methods for the symmetric Lipschitz continuous function (`SymDIRECT, DISIMPL`).

This is confirmed by numerous numerical experiments carried out when solving a center-based clustering problem with synthetic data. A center-based clustering problem (10) for the set $\mathcal{A} \subset \mathbb{R}^n$ is a very demanding global optimization problem even in the case of data having only one feature. The objective function for this problem is a symmetric Lipschitz continuous function. Also, it is proved that, with natural conditions on the data, this problem always has a solution.

It can be expected that the new Separation Method can be applied to clustering problems in higher dimensions, too. Namely, linear transformation $\mathcal{T}$ given by (5) is very simple, and after that everything depends on the choice of the global optimization method on $[0,1]^k$.

Finally, it should be emphasized that the proposed new `Separation Method` allows for the direct and efficient use of well-known global optimization algorithms, as for example `DIRECT`.

**Acknowledgements.**

The author would like to thank the referees and the journal editors for their careful reading of the paper and insightful comments that helped us to improve the paper. Especially, the author would like to thank Mrs. Ivanka Ferčec. This work was supported by the Croatian Science Foundation through research grants `IP-2016-06-6545` “The optimization and statistical models and methods in recognizing properties of data sets measured with errors” and `IP-2016-06-8350` “Methodological framework for efficient energy management by intelligent data analytics”.

# References


[1] Bagirov, A.M., Ugon, J., 2005. An algorithm for minimizing clustering functions. Optimization 54, 351–368.

[2] Bandyopadhyay, S., Saha, S., 2013. Unsupervised Classification: Similarity Measures, Classical and Metaheuristic Approaches, and Applications. Springer.

[3] Bezdek, J.C., Keller, J., Krisnapuram, R., Pal, N.R., 2005. Fuzzy models and algorithms for pattern recognition and image processing. Springer, New York.

[4] Bishop, C.M., 2006. Pattern Recognition and Machine Learning. Springer.

[5] DiSerafino, D., Liuzzi, G., Piccialli, V., Riccio, F., Toraldo, G., 2011. A modified dividing rectangles algorithm for a problem in astrophysics. J Optim Theory Appl 151, 175–190.

[6] Evtushenko, Y.G., 1985. Numerical Optimization Techniques (Translations Series in Mathematics and Engineering). Springer-Verlag, Berlin.

[7] Finkel, D.E., 2003. DIRECT Optimization Algorithm User Guide. Center for Research in Scientific Computation. North Carolina State University. `http://www4.ncsu.edu/~ctk/Finkel_Direct/DirectUserGuide_pdf.pdf`.

[8] Finkel, D.E., Kelley, C.T., 2006. Additive scaling and the `DIRECT` algorithm. Journal of Global Optimization 36, 597–608.

[9] Floudas, C.A., Gounaris, C.E., 2009. A review of recent advances in global optimization. Journal of Global Optimization 45, 3–38.

[10] Gablonsky, J.M., 2001. DIRECT Version 2.0. Technical Report. Center for Research in Scientific Computation. North Carolina State University.

[11] Grbić, R., Nyarko, E.K., Scitovski, R., 2013. A modification of the DIRECT method for Lipschitz global optimization for a symmetric function. Journal of Global Optimization 57, 1193–1212.

[12] Hansen, E., Walster, G.W., 2004. Global optimization using interval analysis. Marcel Dekker, New York. 2nd edition.

[13] Iyigun, C., 2007. Probabilistic Distance Clustering. Ph.D. thesis. Graduate School – New Brunswick, Rutgers.

[14] Iyigun, C., Ben-Israel, A., 2010. A generalized Weiszfeld method for the multi-facility location problem. Operations Research Letters 38, 207–214.

[15] Jones, D.R., Perttunen, C.D., Stuckman, B.E., 1993. Lipschitzian optimization without the Lipschitz constant. Journal of Optimization Theory and Applications 79, 157–181.

[16] Kogan, J., 2007. Introduction to Clustering Large and High-dimensional Data. Cambridge University Press, New York.

[17] Kvasov, D.E., Sergeyev, Y.D., 2009. A univariate global search working with a set of Lipschitz constants for the first derivative. Optimization Letters 3, 303 − 318.

[18] Kvasov, D.E., Sergeyev, Y.D., 2012a. Lipschitz gradients for global optimization in a one-point-based partitioning scheme. Journal of Computational and Applied Mathematics 236, 4042 − 4054.

[19] Kvasov, D.E., Sergeyev, Y.D., 2012b. Univariate geometric Lipschitz global optimisation algorithms. Numerical Algebra, Control and Optimization 2, 69–90.

[20] Leisch, F., 2006. A toolbox for k-centroids cluster analysis. Computational Statistics & Data Analysis 51, 526–544.

[21] Marošević, T., Scitovski, R., 2015. Multiple ellipse fitting by center-based clustering. Croatian Operational Research Review 6, 43–53.

[22] Morales-Esteban, A., Martínez-Álvarez, F., Scitovski, S., Scitovski, R., 2014. A fast partitioning algorithm using adaptive Mahalanobis clustering with application to seismic zoning. Computers & Geosciences 73, 132–141.

[23] Neumaier, A., 2004. Complete search in continuous global optimization and constraint satisfaction. Acta Numerica 13, 271–369.

[24] Paulavičius, R., Sergeyev, Y., Kvasov, D., Žilinskas, J., 2014. Globally-biased DISIMPL algorithm for expensive global optimization. Journal of Global Optimization 59, 545–567.

[25] Paulavičius, R., Žilinskas, J., 2014a. Simplicial Global Optimization. Springer.

[26] Paulavičius, R., Žilinskas, J., 2014b. Simplicial Lipschitz optimization without Lipschitz constant. Journal of Global Optimization 59, 23–40.

[27] Paulavičius, R., Žilinskas, J., 2016. Advantages of simplicial partitioning for Lipschitz optimization problems with linear constraints. Optimization Letters 10, 237–246.

[28] Pintér, J. (Ed.), 2006. Global Optimization: Scientific and Engineering Case Studies. Springer.

[29] Pintér, J.D., 1996. Global Optimization in Action (Continuous and Lipschitz Optimization: Algorithms, Implementations and Applications). Kluwer Academic Publishers, Dordrecht.

[30] Sabo, K., Scitovski, R., 2015. An approach to cluster separability in a partition. Information Sciences 305, 208–218.

[31] Sabo, K., Scitovski, R., Vazler, I., 2013. One-dimensional center-based $l_1$-clustering method. Optimization Letters 7, 5–22.

[32] Schöbel, A., Scholz, D., 2010. The big cube small cube solution method for multidimensional facility location problems. Computers & Operations Research 37, 115 – 122.

[33] Scitovski, R., Marošević, T., 2014. Multiple circle detection based on center-based clustering. Pattern Recognition Letters 52, 9–16.

[34] Scitovski, R., Sabo, K., 2014. Analysis of the $k$-means algorithm in the case of data points occurring on the border of two or more clusters. Knowledge-Based Systems 57, 1–7.

[35] Scitovski, R., Scitovski, S., 2013. A fast partitioning algorithm and its application to earthquake investigation. Computers & Geosciences 59, 124–131.

[36] Sergeyev, Y.D., Famularo, D., Pugliese, P., 2001. Index branch-and-bound algorithm for Lipschitz univariate global optimization with multiextremal constraints. Journal of Global Optimization 21, 317–341.

[37] Sergeyev, Y.D., Kvasov, D.E., 2006. Global search based on efficient diagonal partitions and a set of Lipschitz constants. SIAM Journal on Optimization 16, 910 – 937.

[38] Sergeyev, Y.D., Kvasov, D.E., 2008. Diagonal Global Optimization Methods. FizMatLit, Moscow. (in Russian).

[39] Sergeyev, Y.D., Kvasov, D.E., 2011. Lipschitz global optimization, in: Cochran, J. (Ed.), Wiley Encyclopedia of Operations Research and Management Science. Wiley, New York. volume 4, pp. 2812–2828.

[40] Späth, H., 1983. Cluster-Formation und Analyse. R. Oldenburg Verlag, München.

[41] Strongin, R.G., Sergeyev, Y.D., 2000. Global Optimization with Non-Convex Constraints: Sequential and Parallel Algorithms. Kluwer Academic Publishers, Dordrecht.

[42] Szeliski, R., 2011. Computer Vision: Algorithms and Applications. Springer.

[43] Theodoridis, S., Koutroumbas, K., 2009. Pattern Recognition. Academic Press, Burlington. $4^{th}$ edition.

[44] Vidović, I., Scitovski, R., 2014. Center-based clustering for line detection and application to crop rows detection. Computers and Electronics in Agriculture 109, 212–220.

[45] Wolfram Research, I., 2016. Mathematica. Wolfram Research, Inc., Champaign, Illinois. version 11.0 edition.