Journal of Global Optimization (2005) 32:161–179
DOI 10.1007/s10898-004-2700-0

Local Optimization Method with Global Multidimensional Search

ADIL M. BAGIROV, ALEXANDER M. RUBINOV and JIAPU ZHANG
Centre for Informatics and Applied Optimization, School of Information Technology and Mathematical Sciences, University of Ballarat, Victoria 3353, Australia
(e-mail: a.rubinov@ballarat.edu.au)

(Received 1 April 2004; accepted in revised form 7 April 2004)

Abstract. This paper presents a new method for solving global optimization problems. We use a local technique based on the notion of discrete gradients for finding a cone of descent directions and then we use a global cutting angle algorithm for finding global minimum within the intersection of the cone and the feasible region. We present results of numerical experiments with well-known test problems and with the so-called cluster function. These results confirm that the proposed algorithms allows one to find a global minimizer or at least a deep local minimizer of a function with a huge amount of shallow local minima.

Key words: Derivative-free optimization, Discrete gradient, Global optimization, Lipschitz programming, The cutting angle method

1. Introduction

Numerical methods for global optimization are very time consuming and could not be applied for high-dimensional non-convex optimization problems. This is the reason why many researches try to use various combinations of global and local search techniques. Strategies based on different combinations of global and local searches can be used. In particular, the following two types of such combinations are used:

1) A local technique is used in order to obtain a stationary point (local minimum). Then a global technique should be applied in order to escape from the obtained stationary point and find a new point which can be used as an initial guess for the new round of local search; (see e.g., [8, 11, 19, 28]).
2) Points obtained by a global technique are used as initial points for a local search (see e.g., [21]).

Descent methods of local optimization are based on the following idea. Applying a local approximation of an objective function at the point in hand, we need to find a descent direction and then the step-size along this direction. Local approximation of the first order is given by the gradient or by some of its substitutes. The size of the descent can be found by

different methods, in particular by global one-dimensional minimization. This approach is good enough for local minimization of functions with a few stationary points, however it does not work properly for functions with many shallow local minima. Indeed, for such a function we are mainly interested in a deep enough local minimum, and a local search usually entraps at a shallow local minimizer or even at a stationary point which is not a local minimizer.

Some methods for global optimization are fast enough in small dimensions. This observation gives rise to a completely new combination of local and global technique which is discussed in this paper. Namely, we suggest to apply a global technique for the search for the descent in dimensions higher than one, using a local approximation of the function at the point at hand. *This approach is beneficial for minimization of non-smooth functions with many shallow local minima since it allows one to find a deep enough local minimizer and even a global minimizer*. It can be also used for minimization of smooth functions.

For application of this approach we need to have a good local approximation of an objective function and a fast enough (in small dimensions) method for a global search. Since we are mainly interested in the minimization of *non-smooth function*, we consider a special approximation of Clarke subdifferential and quasidifferential given by discrete gradients [3–5] for a local approximation of the objective function. For a global search we use the cutting angle method [1, 2, 6, 12]). We propose the algorithm for minimization that is based on the use of discrete gradients and the cutting angle method.

The proposed algorithm was applied to two classes of global optimization problems. One of these classes consists of well-known test problems with smooth objective functions (see [23]). The other class consists of problems with the objective function of the form

$$f(x^1, \ldots, x^k) = \sum_{i=1}^{m} \min_{1 \leqslant j \leqslant k} ||x^j - a^i||_p \quad x^j \in \mathbb{R}^n, \quad j = 1, \ldots, k, \tag{1}$$

where $A = \{a^i\}_{i\in 1}^m$ is a finite set of points. Note that f depends on $n \cdot k$ variables. The function f in Equation (1) is called a cluster function (see, [9] and references therein). Such a function is used in cluster analysis. Many location problems can also be reduced to the minimization of a cluster function (see, for example, [15]). It is well-known that the cluster function has very many shallow local minima. We suggest a special method for the minimization of a cluster function which allows one to find a good initial point. In order to find such an initial point we need to solve an optimization problem of dimension n with non-smooth and non-convex objective function. We used the proposed algorithm for both the search of an initial point and the minimization of a cluster function.

The results of numerical experiments confirm that the proposed algorithm can be successfully applied for solving many problems of global optimization.

The structure of this paper is as follows. In Section 2 we briefly recall the notion of discrete gradients which we use for a local approximation of a function and also local discrete gradient method and global cutting angle method. Section 3 provides a description of the main algorithm which we propose. In Section 4 we discuss the results of numerical experiments with well-known test problems. Section 5 contains an algorithm for the minimization of cluster functions. Results of numerical experiments with the minimization of a cluster function are presented in Section 6. Section 7 contains concluding remarks.

2. Preliminaries

In this section we recall some known methods for local approximation and global optimization which will be used in this paper.

2.1. THE DISCRETE GRADIENT METHOD

In this subsection we will briefly describe a local approximation of a locally Lipschitz function by means of discrete gradients. We also describe a known algorithm for calculation of descent directions based on discrete gradients and the discrete gradient method for the local minimization. We start with the definition of the discrete gradient.

2.1.1. *Definition of the discrete gradient*

Let φ be a locally Lipschitz continuous function defined on $\mathbb{R}^n$. Let

$$S_1 = \{g \in \mathbb{R}^n : ||g|| = 1\}, G = \{e \in \mathbb{R}^n : e = (e_1, \ldots, e_n), |e_j| = 1, \quad j = 1, \ldots, n\},$$

$$P = \{z(\lambda) : z(\lambda) \in \mathbb{R}^1, \quad z(\lambda) > 0, \ \lambda > 0, \ \lambda^{-1} z(\lambda) \to 0, \ \lambda \to 0\},$$

$$I(g, \alpha) = \{i \in \{1, \ldots, n\} : |g_i| \geqslant \alpha\},$$

where $\alpha \in (0, n^{-1/2}]$ is a fixed number.

Here S_1 is the unit sphere, G is the set of vertices of the unit hypercube in $\mathbb{R}^n$ and P is the set of univariate positive infinitesimal functions.

We define operators $H_i^j : \mathbb{R}^n \to \mathbb{R}^n$ for $i = 1, \ldots, n$, $j = 0, \ldots, n$ by the formula

$$H_i^j g = \begin{cases} (g_1, \ldots, g_j, 0, \ldots, 0) & \text{if } j < i, \\ (g_1, \ldots, g_{i-1}, 0, g_{i+1}, \ldots, g_j, 0, \ldots, 0) & \text{if } j \geqslant i. \end{cases} \tag{2}$$

We can see that

$$H_i^j g - H_i^{j-1} g = \begin{cases} (0, \ldots, 0, g_j, 0, \ldots, 0) & \text{if } j = 1, \ldots, n, \ j \neq i, \\ 0 & \text{if } j = i. \end{cases} \tag{3}$$

Let $e(\beta) = (\beta e_1, \beta^2 e_2, \ldots, \beta^n e_n)$, where $\beta \in (0, 1]$. For $x \in \mathbb{R}^n$ we consider vectors

$$x_i^j = x_i^j(g, e, z, \lambda, \beta) = x + \lambda g - z(\lambda) H_i^j e(\beta), \tag{4}$$

where $g \in S_1$, $e \in G$, $i \in I(g, \alpha)$, $z \in P$, $\lambda > 0$, $j = 0, \ldots, n$, $j \neq i$.

It follows from Equation (3) that

$$x_i^{j-1} - x_i^j = \begin{cases} (0, \ldots, 0, z(\lambda) e_j(\beta), 0, \ldots, 0) & \text{if } j = 1, \ldots, n, \ j \neq i, \\ 0 & \text{if } j = i. \end{cases} \tag{5}$$

It is clear that $H_i^0 g = 0$ and $x_i^0(g, e, z, \lambda, \beta) = x + \lambda g$ for all $i \in I(g, \alpha)$.

DEFINITION 1. (see [5]) The discrete gradient of the function φ at the point $x \in \mathbb{R}^n$ is the vector $\Gamma^i(x, g, e, z, \lambda, \beta) = (\Gamma_1^i, \ldots, \Gamma_n^i) \in \mathbb{R}^n$, $g \in S_1$, $i \in I(g, \alpha)$, with the following coordinates:

$$\Gamma_j^i = [z(\lambda) e_j(\beta)]^{-1} \left[\varphi(x_i^{j-1}(g, e, z, \lambda, \beta)) - \varphi(x_i^j(g, e, z, \lambda, \beta)) \right],$$
$$j = 1, \ldots, n, \ j \neq i,$$

$$\Gamma_i^i = z(\lambda g_i)^{-1} \left[\varphi(x_i^n(g, e, z, \lambda, \beta)) - \varphi(x) - \sum_{j=1, j \neq i}^{n} \Gamma_j^i (\lambda g_i - z(\lambda) e_j(\beta)) \right].$$

A more detailed description of the discrete gradient and examples can be found in [3]. The discrete gradient is an approximation to a subgradient of a locally Lipschitz function [5].

REMARK 1. It follows from Definition 1 that for the calculation of the discrete gradient $\Gamma^i(x, g, e, z, \lambda, \beta)$, $i \in I(g, \alpha)$ we define a sequence of points

$$x_i^0, \ldots, x_i^{i-1}, x_i^{i+1}, \ldots, x_i^n.$$

For the calculation of the discrete gradient it is sufficient to evaluate the function φ at each point of this sequence.

REMARK 2. The discrete gradient is defined with respect to a given direction $g \in S_1$. We can see that for the calculation of one discrete gradient we have to calculate $(n + 1)$ values of the function φ: at the point x and at the points $x_i^j(g, e, z, \lambda, \beta)$, $j = 0, \ldots, n$, $j \neq i$. For the calculation of another discrete gradient at the same point with respect to any other direction $g^1 \in S_1$

we have to calculate this function n times, because we have already calculated φ at the point x.

2.1.2. *The method*

We consider the following unconstrained minimization problem:

$$\text{minimize } \varphi(x) \text{ subject to } x \in \mathbb{R}^n, \tag{6}$$

where the function φ is assumed to be semismooth (for the definition of semismooth functions see [22]). We consider the following algorithm for solving this problem. An important step in this algorithm is the calculation of a descent direction of the objective function φ. Therefore first, we describe an algorithm for the computation of this descent direction.

Let $z \in P$, $\lambda > 0$, $\beta \in (0, 1]$, the number $c \in (0, 1)$ and a small enough number $\delta > 0$ be given.

ALGORITHM 1. An algorithm for the computation of the descent direction.

Step 1. Choose any $g^1 \in S_1$, $e \in G$, $i \in I(g^1, \alpha)$ and compute a discrete gradient $v^1 = \Gamma^i(x, g^1, e, z, \lambda, \beta)$. Set $\overline{D}_1(x) = \{v^1\}$ and $k = 1$.

Step 2. Calculate the vector $||w^k|| = \min\{||w|| : w \in \overline{D}_k(x)\}$. If

$$||w^k|| \leqslant \delta, \tag{7}$$

then stop. Otherwise go to Step 3.

Step 3. Calculate the search direction by $g^{k+1} = -||w^k||^{-1} w^k$.

Step 4. If

$$\varphi(x + \lambda g^{k+1}) - \varphi(x) \leqslant -c\lambda ||w^k||, \tag{8}$$

then stop. Otherwise go to Step 5.

Step 5. Calculate a discrete gradient

$$v^{k+1} = \Gamma^i(x, g^{k+1}, e, z, \lambda, \beta), \quad i \in I(g^{k+1}, \alpha),$$

construct the set $\overline{D}_{k+1}(x) = \text{co}\{\overline{D}_k(x) \bigcup \{v^{k+1}\}\}$, set $k = k + 1$ and go to Step 2.

Algorithm 1 contains some steps which deserve an explanation. In Step 1 we calculate the first discrete gradient. The distance between the convex hull of all calculated discrete gradients and the origin is calculated in Step 2. If this distance is less than the tolerance $\delta > 0$ then we accept the point x as an approximate stationary point (Step 2), otherwise we calculate the search direction in Step 3. In Step 4 we check whether this direction is a descent direction. If it is we stop and the descent direction has been

calculated, otherwise we calculate another discrete gradient with respect to this direction in Step 5 and add it to the set $\overline{D}_k$.

It is proved that Algorithm 1 is terminating (see [3, 4]).

Let sequences $\delta_k > 0$, $z_k \in P$, $\lambda_k > 0$, $\beta_k \in (0, 1]$, $\delta_k \to +0$, $z_k \to +0$, $\lambda_k \to +0$, $\beta_k \to +0$, $k \to +\infty$ and numbers $c_1 \in (0, 1)$, $c_2 \in (0, c_1]$ be given.

ALGORITHM 2. Discrete gradient method

Step 1. Choose any starting point $x^0 \in \mathbb{R}^n$ and set $k = 0$.

Step 2. Set $s = 0$ and $x_s^k = x^k$.

Step 3. Apply Algorithm 1 for the calculation of the descent direction at $x = x_s^k$, $\delta = \delta_k$, $z = z_k$, $\lambda = \lambda_k$, $\beta = \beta_k$, $c = c_1$. This algorithm terminates after a finite number of iterations $m > 0$. As a result we get the set $\overline{D}_m(x_s^k)$ and an element v_s^k such that

$$||v_s^k|| = \min\{||v|| : v \in \overline{D}_m(x_s^k)\}.$$

Furthermore either $||v_s^k|| \leqslant \delta_k$ or for the search direction $g_s^k = -||v_s^k||^{-1} v_s^k$

$$\varphi(x_s^k + \lambda_k g_s^k) - \varphi(x_s^k) \leqslant -c_1 \lambda_k ||v_s^k||. \tag{9}$$

Step 4. If

$$||v_s^k|| \leqslant \delta_k \tag{10}$$

then set $x^{k+1} = x_s^k$, $k = k + 1$ and go to Step 2. Otherwise go to Step 5.

Step 5. Construct the following iteration $x_{s+1}^k = x_s^k + \sigma_s g_s^k$, where σ_s is defined as follows

$$\sigma_s = \arg\max\{\sigma \geqslant 0 : \varphi(x_s^k + \sigma g_s^k) - \varphi(x_s^k) \leqslant -c_2 \sigma ||v_s^k||\}.$$

Step 6. Set $s = s + 1$ and go to Step 3.

For the point $x^0 \in \mathbb{R}^n$ we consider the set $M(x^0) = \{x \in \mathbb{R}^n : \varphi(x) \leqslant \varphi(x^0)\}$.

THEOREM 1. ([3]) *Assume that the set $M(x^0)$ is bounded for starting points $x^0 \in \mathbb{R}^n$. Then every accumulation point of $\{x^k\}$ belongs to the set $X^0 = \{x \in \mathbb{R}^n : 0 \in \partial\varphi(x)\}$.*

2.2. CUTTING ANGLE METHOD

In this subsection we consider the following problem of global optimization:

$$\text{minimize } f(x) \text{ subject to } x \in S, \tag{11}$$

where the objective function f is an increasing positively homogeneous (IPH) of degree one and the set S is the unit simplex in $\mathbb{R}^n$:

$$S = \left\{ x \in \mathbb{R}^n_+ : \sum_{i=1}^{n} x_i = 1 \right\}.$$

Here $\mathbb{R}^n_+ = \{x \in \mathbb{R}^n : x_i \geqslant 0, i = 1, \ldots, n\}$.

Recall that a function f defined on $\mathbb{R}^n_+$ is called increasing if $x \geqslant y$ implies $f(x) \geqslant f(y)$; the function f is positively homogeneous of degree one if $f(\lambda x) = \lambda f(x)$ for all $x \in \mathbb{R}^n_+$ and $\lambda > 0$.

For a given vector $l \in \mathbb{R}^n_+$, $l \neq 0$ we put $I(l) = \{i = 1, \ldots, n : l_i > 0\}$. We use the following notation for $c \in \mathbb{R}$ and $l \in \mathbb{R}^n_+$:

$$(c/l)_i = \begin{cases} c/l_i & \text{if } i \in I(l), \\ 0 & \text{if } i \notin I(l). \end{cases}$$

Note that an IPH function is non-negative on $\mathbb{R}^n_+$. We assume that $f(x) > 0$ for all $x \in S$. It follows from positiveness of f that $I(l) = I(x)$ for all $x \in S$ and $l = f(x)/x$. Let e^k be the k-th orthant vector, $k = 1, \ldots, n$. Now we describe the cutting angle method for solving problem (11).

ALGORITHM 3. The cutting angle method.

Step 0. (Initialization) Take points $x^k \in S$, $k = 1, \ldots, m$, where $m \geqslant n$, $x^k = e^k$ for $k = 1, \ldots, n$ and $x^k_j > 0$ for $k = n+1, \ldots, m$, $j = 1, \ldots, n$. Let $l^k = f(x^k)/x^k$, $k = 1, \ldots, m$. Define the function h_m:

$$h_m(x) = \max_{k \leqslant m} \min_{i \in I(l^k)} l^k_i x_i = \max \left\{ \max_{k \leqslant n} l^k_k x_k, \max_{n+1 \leqslant k \leqslant m} \min_{i \in I(l^k)} l^k_i x_i \right\} \tag{12}$$

and set $j = m$.

Step 1. Find a solution x^* of the problem

$$\text{minimize } h_j(x) \text{ subject to } x \in S. \tag{13}$$

Step 2. Set $j = j + 1$ and $x^j = x^*$.

Step 3. Compute $l^j = f(x^j)/x^j$, define the function

$$h_j(x) = \max\{h_{j-1}(x), \min_{i \in I(l^j)} l^j_i x_i\} \equiv \max_{k \leqslant j} \min_{i \in I(l^k)} l^j_i x_i \tag{14}$$

and go to Step 1.

A more detailed description of Algorithm 3 with necessary explanations can be found in [6, 7, 26]. This algorithm can be considered as a version of the cutting angle method ([1, 2]). The cutting angle method provides a sequence of lower estimates for the global minimum f_* of Equation (11)

with an IPH objective function, which converges to f_*. Theoretically this sequence can be used for establishment of a stopping criterion (see [26] for details). Let

$$\lambda_j = \min_{x \in S} h_j(x) = h_j(x^{j+1}) \tag{15}$$

be the value of the problem (13). λ_j is a lower estimate of the global minimum f_*. It is known (see, for example, [26]), that λ_j is an increasing sequence and $\lambda_j \to f_*$ as $j \to +\infty$.

The cutting angle method constructs the sequence $\{f(x^j)\}$, which is not necessarily decreasing: it is possible that $f(x^{j+1}) > f(x^j)$ for some j.

The most difficult and time-consuming part of the cutting angle method is solving the auxiliary problem (13). An algorithm for the solution of this problem was proposed in [6]. Some modifications of this algorithm (and corresponding modifications of the cutting angle method) are discussed in [7] and [12].

Only one value of the objective function is used at each iteration of the cutting angle method. Some modifications of this method require to evaluate a few values of the objective function at each iteration.

2.2.1. *Global minimization of Lipschitz functions*

Now we consider the following problem of global optimization:

$$\text{minimize } f(x) \text{ subject to } x \in S, \tag{16}$$

where the function f is Lipschitz continuous on S. This problem can be reduced to the global minimization of a certain IPH function over S. The following theorem has been established in [27] (see [26]).

THEOREM 2. *Let $f: S \to \mathbb{R}$ be a Lipschitz function and let*

$$L = \sup_{x,y \in S, x \neq y} \frac{|f(x) - f(y)|}{||x - y||_1} \tag{17}$$

be the least Lipschitz constant of f in $||\cdot||_1$-norm, where $||x||_1 = \sum_{i=1}^n |x_i|$. Consider a positively homogeneous function

$$\varphi(x) = \begin{cases} ||x|| f\left(\frac{x}{||x||}\right) & \text{if } x \neq 0, \\ 0 & \text{if } x = 0 \end{cases}$$

defined on $\mathbb{R}^n_+$. If $\min_{x \in S} f(x) \geqslant 2L$ then φ is an IPH function and $\varphi(x) = f(x)$ for all $x \in S$.

Let

$$c \geqslant 2L - \min_{x \in S} f(x), \tag{18}$$

where L is defined by Equation (17). Let $f_1(x) = f(x) + c$. Theorem 2 implies that the function f_1 can be extended to an IPH function φ. Since

$f_1(x) = \varphi(x)$ for all $x \in S$ the global minimization of φ over S is equivalent to the following global optimization problem:

$$\text{minimize } f_1(x) \text{ subject to } x \in S \tag{19}$$

and consequently the cutting angle method can be applied to solve this problem. On the other hand the functions f and f_1 have the same minimizers on the simplex S and if the constant c is known the problem (16) can be solved by the cutting angle method. In order to estimate c we need to know an upper estimation of the Lipschitz constant L and a lower estimation of the desired global minimum of the function f. We will assume that c is a sufficiently large number. However it should be noted that for increasing values of c the cutting angle method works less efficiently.

3. The Main Algorithm

In this section we consider the following global optimization problem:

$$\text{minimize } f(x) \text{ subject to } x \in D \tag{20}$$

where

$$D = \{x \in \mathbb{R}^n : a_i \leqslant x_i \leqslant b_i, \quad i = 1, \ldots, n\}.$$

We propose the following algorithm for solving problem (20).

Let sequences $\delta_k > 0$, $z_k \in P$, $\lambda_k > 0$, $\beta_k \in (0, 1]$, $\delta_k \to +0$, $z_k \to +0$, $\lambda_k \to +0$, $\beta_k \to +0$, $k \to +\infty$, numbers $c_1 \in (0, 1)$, $c_2 \in (0, c_1]$ and $c_3 > 0$ be given.

ALGORITHM 4. The discrete gradient method with global search.

Step 1. Choose any starting point $x^0 \in D$ and set $k = 0$.

Step 2. Set $s = 0$ and $x_s^k = x^k$.

Step 3. Apply Algorithm 1 for the calculation of the descent direction at $x = x_s^k$, $\delta = \delta_k$, $z = z_k$, $\lambda = \lambda_k$, $\beta = \beta_k$, $c = c_1$. This algorithm terminates after a finite number of iterations $m > 0$. As a result we get the set $\overline{D}_m(x_s^k)$ and an element v_s^k such that

$$||v_s^k|| = \min\{||v|| : v \in \overline{D}_m(x_s^k)\}.$$

Furthermore either $||v_s^k|| \leqslant \delta_k$ or for the search direction $g_s^k = -||v_s^k||^{-1} v_s^k$

$$f(x_s^k + \lambda_k g_s^k) - f(x_s^k) \leqslant -c_1 \lambda_k ||v_s^k||. \tag{21}$$

Step 4. If

$$||v_s^k|| \leqslant \delta_k \tag{22}$$

and $m > 2$ then set $x_{s+1}^k = x_s^k$ and go to Step 7. If Equation (22) satisfies and $m \leqslant 2$ then set $x^{k+1} = x_s^k$, $k = k + 1$ and go to Step 2. Otherwise go to Step 5.

Step 5. Construct the following iteration $x_{s+1}^k = x_s^k + \sigma_s g_s^k$, where σ_s is defined as follows

$$\sigma_s = \arg\max\{\sigma \geqslant 0 : x_s^k + \sigma g_s^k \in D, \quad f(x_s^k + \sigma g_s^k) - f(x_s^k) \leqslant -c_2\sigma||v_s^k||\}.$$

Step 6. If $m = 1$ then set $s = s + 1$ and go to Step 3. Otherwise go to Step 7.

Step 7. Calculate two discrete gradients w_1, $w_2 \in \overline{D}_m(x_s^k)$ such that

$$\frac{\langle w^1, w^2\rangle}{||w^1||||w^2||} = \min\left\{\frac{\langle v^1, v^2\rangle}{||v^1||||v^2||} : v^1, v^2 \in \overline{D}_m(x_s^k), \quad ||v^1||, ||v^2|| \neq 0\right\}$$

Step 8. Set $y^1 = x_{s+1}^k$, $g^1 = -w^1$, $g^2 = -w^2$ and $I(g^t) = \{i \in \{1, \ldots, n\} : g_i^t \neq 0\}$, $t = 1, 2$. For $t = 1, 2$ calculate the maximum step-sizes along the directions $g^t, t = 1, 2$:

$$s_t^1 = \min\left\{\frac{a_i - y_i^1}{g_i^t}, i \in I(g^t), \quad g_i^t < 0\right\},$$

$$s_t^2 = \min\left\{\frac{b_i - y_i^1}{g_i^t}, i \in I(g^t), \quad g_i^t > 0\right\},$$

$$s_t = \min\{s_t^1, s_t^2\}, \quad t = 1, 2.$$

Set $\bar{s} = \max\{s_1, s_2\}$.

Step 9. Calculate the points y^2 and y^3 as follows:

$$y^2 = y^1 + s_1 g^1, \quad y^3 = y^1 + s_2 g^2$$

and construct the following set:

$$\overline{S} = \{v \in \mathbb{R}^n : v = \alpha_1 y^1 + \alpha_2 y^2 + \alpha_3 y^3, \quad \alpha_1 + \alpha_2 + \alpha_3 = 1, \quad \alpha_1, \alpha_2, \alpha_3 \geqslant 0\}.$$

Step 10. Apply the cutting angle method to solve the following global optimization problem:

$$\text{minimize } f(x) \text{ subject to } x \in \overline{S}. \tag{23}$$

Step 11. Let $x^* \in \overline{S}$ be a solution to the problem (23). If

$$f(x^*) - f(x_{s+1}^k) \leqslant -c_3\bar{s}$$

then set $x_{s+1}^k = x^*$, $s = s+1$ and go to Step 3. Otherwise go to Step 12.

Step 12. If (21) satisfies then set $s = s+1$ and go to Step 3. Otherwise set $x^{k+1} = x_{s+1}^k$, $k = k+1$ and go to Step 2.

Now we give some explanations to Algorithm 4. In Step 1 we select any starting point and then apply the discrete gradient method with the starting values of the parameters in the definition of the discrete gradient. In Step 3 Algorithm 1 is applied to calculate a descent direction or to determine that the current point x_s^k is an approximate stationary point. If we calculate a descent direction in Step 3, that is the condition (21) is satisfied then we carry out a line search along this direction at the point x_s^k in Step 5 and calculate a new point x_{s+1}^k. It should be noted that the step-size σ_s in Step 5 is computed approximately using Armijo type line search. If the number of the calculated discrete gradients is more than two (this allows us to construct two-dimensional set for a global search) we carry out a global search in Steps 7–11.

If x_s^k is the approximate stationary point and the number of the calculated discrete gradients at this point is more than two then we carry out global search in Steps 7–11 trying to escape from x_s^k, otherwise we change the parameters for the calculation of the discrete gradient to get a better approximation to the subdifferential.

Two-dimensional global search is carried out in Steps 7–11. In Step 7 we calculate two discrete gradients from the set of discrete gradients $\overline{D}_m^k$ with largest angle between them. These two discrete gradients give us two directions g^1 and g^2 (Step 8), respectively. Then we calculate two points (y^2 and y^3 in Step 9) where the rays from the current point $y^1 = x_{s+1}^k$ meet the boundary of the feasible region. Using these two points and the point y^1 we construct the set $\overline{S}$ for the global search by the cutting angle method (Steps 9 and 10).

The problem (23) can be rewritten as follows:

$$\text{minimize } \psi(\alpha) \text{ subject to } \alpha \in S \tag{24}$$

where

$$S = \{\alpha = (\alpha_1, \alpha_2, \alpha_3) : \alpha_1 + \alpha_2 + \alpha_3 = 1, \quad \alpha_1, \alpha_2, \alpha_3 \geqslant 0\}$$

and

$$\psi(\alpha_1, \alpha_2, \alpha_3) = f(\alpha_1 y^1 + \alpha_2 y^2 + \alpha_3 y^3).$$

It follows from the results stated in Subsection 2.2 that the problem (24) can be reduced to the global minimization of a certain IPH function over the unit simplex and the cutting angle method can be applied for its solution.

Step 11 checks whether the global search achieves the guaranteed decrease of the objective function. If it is we do not change the parameters

of the discrete gradients in Step 3. If both the line search and global search do not achieve the guaranteed decrease of the objective we change the parameters of the discrete gradient in Step 3 to get a better approximation to the subdifferential.

It is clear that all accumulation points of the sequence generated by Algorithm 4 are stationary points of problem (20).

4. Results of Numerical Experiments

In this section we report the results of numerical experiments for some known test problems with smooth objective functions involved. The following test problems of global optimization were used in numerical experiments: Ackleys function (A1) , Branin function (B1), Camel function (C1), two Levy functions (L2 and L3), Rastrigin function (R1) and three Shubert functions (Sh1, Sh2 and Sh3). We consider the problem of global minimization of these functions subject to box-constraints. The description of functions and the corresponding box-constraints can be found, for example, in [23].

In numerical experiments 50 initial points were randomly generated from the feasible region. Thus 50 results were obtained by the proposed algorithm starting from 50 different initial points. In all problems we used two-dimensional search by the cutting angle method. At each global search the number of iterations by the cutting angle method was restricted to 100.

The codes of algorithms have been written in Fortran 95 and numerical experiments have been carried out in VPAC's (Victorian Partnership for Advanced Computing), supercomputer in Melbourne using one processor with 833 MHz.

Results of numerical experiments are presented in Table 1. The following notations are used in this table. In column "known gl. min" we present the known global minimum of the corresponding problem. In columns "mean" and "st. dev." mean values and standard deviation of all results obtained by the discrete gradient and DG + CAM methods are presented, respectively.

Results presented in Table 1 show that in all problems except Levy function No. 2, the DG + CAM method improved the results obtained by the discrete gradient method. In many cases such an improvement is significant. In particular, DG + CAM method achieved much better results for Ackleys function ($n = 2$), Branin function, Camel function, Levy function L3 ($n = 4, 5$), Rastrigin function ($n = 2, 5$) and all Shubert functions. Values of standard deviations for DG and DG + CAM methods show that the latter method allows one to get better and more "stable" results.

Table 1. Results of numerical experiments

Prob.	n	Known gl. min.	Discrete gradient			+CAM		
			f_{best}	Mean	st. dev.	f_{best}	mean	st.dev.
A1	2	0.00	0.2801	4.5869	2.3187	0.0000	1.8253	1.4097
A1	10	0.00	3.9491	3.9491	0.0000	2.398	3.8179	0.4027
A1	30	0.00	3.9296	3.9477	0.0082	2.1155	3.6924	0.5969
B1	2	0.00	0.0000	2.3792	3.2117	0.0000	0.1240	0.5413
C1	2	−1.0316	−1.0316	−0.7868	0.3778	−1.0316	−0.9990	0.1616
L2	2	0.00	0.0000	0.8708	1.5442	0.0000	0.8708	1.5442
L2	10	0.00	0.0000	0.0187	0.0746	0.0000	0.0187	0.0746
L2	20	0.00	0.0000	0.0933	0.0770	0.0000	0.0933	0.0770
L3	4	−21.5024	−21.5024	−4.3183	15.6069	−21.5024	−18.3553	9.6436
L3	5	−11.5044	−11.5044	−6.7422	4.2938	−11.5044	−11.2685	0.8825
R1	2	0.00	0.0000	13.1931	11.2383	0.0000	4.9151	9.3581
R1	5	0.00	7.9597	23.1427	9.7087	0.9950	14.9641	8.4591
R1	10	0.00	36.8740	40.6341	3.5355	24.8740	38.3855	4.6715
R1	20	0.00	79.5966	82.9795	2.3442	30.8437	81.2681	8.4698
R1	30	0.00	119.3949	127.99131	5.41521	9.9496	98.9386	41.6100
Sh1	2	−186.7309	−186.7309	−93.1254	73.5074	−186.7309	−113.0273	26.6690
Sh2	2	−186.7309	−186.7309	−81.6387	67.8182	−186.7309	−118.6854	17.0499
Sh3	2	−24.0625	−24.0625	−16.5940	4.6777	−24.0625	−21.5355	3.8992

Moreover, in the case of Branin, Camel, Levy function L2 ($n = 10, 20$), Levy function L3 ($n = 5$) and Shubert function Sh3 the mean value of all results obtained by DG+CAM method is quite close to the value of the global minimum.

In numerical experiments we restricted the number of iterations generated by the cutting angle method by 100. Otherwise CPU time required by the cutting angle method could be large which makes the proposed algorithm ineffective. The results confirm that DG+CAM requires in average two times more CPU time than the discrete gradient method. Thus we can say that DG+CAM is a local optimization method with better global search properties.

5. Minimization of Cluster Functions

In this section we consider the application of the proposed algorithm for solving cluster analysis problem.

Assume that we are given a finite points set A in n-dimensional space $\mathbb{R}^n : A = \{a_1, \ldots, a^m\}, a^i \in \mathbb{R}^n$. Then the problem of finding centers $x^1, \ldots, x^k$ of k clusters in this set is reduced to the following optimization problem:

$$\text{minimize } f_k(x^1, \ldots, x^k) \text{ subject to } x^j \in \mathbb{R}^n, \quad j = 1, \ldots, k \tag{25}$$

where

$$f_k(x^1, \ldots, x^k) = \sum_{i=1}^{m} \min_{1 \leqslant j \leqslant k} ||x^j - a^i||_p \tag{26}$$

and $||\cdot||_p$ is the p-norm in $\mathbb{R}^n$:

$$||x||_p = \left(\sum_{l=1}^{n} |x_l|^p \right)^{1/p}, \quad x \in \mathbb{R}^n.$$

Solution $\bar{x}^j \in \mathbb{R}^n$, $j = 1, \ldots, k$ of Equation (25) can be considered as centers of clusters.

The function f_k in Equation (26) is called a cluster function (see [9] and references therein). The cluster function is a typical example of the so-called sum–min function, that is the function of the form

$$F(x^1, \ldots, x^k) = \sum_{a^i \in A} \min(\varphi_1(x^1, a^i), \varphi_2(x^2, a^i), \ldots, \varphi_k(x^k, a^i)),$$

where $x \mapsto \varphi_i$ (x, a) is a convex function defined on $\mathbb{R}^m (i = 1, \ldots, k, a \in A)$. Another example of sum–min function can be found in [14].

The problem (25) is a problem of non-smooth and non-convex optimization. The number of variables n in this problem can be very large in many practical applications including the problem of the cluster analysis. Therefore the global optimization techniques as a rule fail to solve this problem.

We propose the following algorithm for solving problem (25). This algorithm is some modification of the algorithm proposed in [10] (see, also, [9]).

ALGORITHM 5. An algorithm for minimization of a cluster function.

Step 1. (Initialization). Select a starting point $x^0 \in \mathbb{R}^n$ and solve the following minimization problem:

$$\text{minimize } f_1(x) \text{ subject } to \ x \in \mathbb{R}^n. \tag{27}$$

Let $x^{1*} \in \mathbb{R}^n$ be a solution to this problem. Set q = 1.

Step 2. (Computation of the next cluster center). Select a starting point $y^0 \in \mathbb{R}^n$ and solve the following minimization problem:

$$\text{minimize } f^{\bar{q}}(y) \text{ subject to } y \in \mathbb{R}^n, \tag{28}$$

where

$$f^{\bar{q}}(y) = \sum_{i=1}^{m} \min\{\|x^{1*} - a^i\|_p, \ldots, \|x^{q*} - a^i\|_p, \|y - a^i\|_p\}. \tag{29}$$

Step 3. (Refinement of all cluster centers). Let y^* be a solution to problem (28). Take $x^{q+1,0} = (x^{1*}, \ldots, x^{q*}, y^*)$ as a new starting point and

solve the problem (25) for $k = q + 1$. Let $x^{q+1,*}$ be a solution to the problem (25) for $k = q + 1$.

Step 4. (Stopping criterion). If $q < k - 1$ then set $q = q + 1$ and go to Step 2. Otherwise stop.

Algorithm 5 contains some steps which need to be explained. In Step 1 the center of the first cluster is calculated. The problem (27) is a convex programming problem. In Step 2 we calculate a center of the next $(q + 1)$-st cluster, assuming the previous q cluster centers to be known and fixed. It should be noted that the number of variables in problem (28) is n which is substantially less than if we calculate all cluster centers simultaneously. In Step 3 the refinement of all $q + 1$ cluster centers is carried out. One can expect that the starting point $x^{q+1,0}$ calculated in Step 2 is not far from the solution to the problem (28). Such an approach allows one to significantly reduce the computational time for solving problem (25).

Algorithm 4 is applied to solve problems (25) and (28). Let

$$\bar{a_i} = \min\{a_i^j, j = 1, \ldots, m\},$$

and

$$\bar{b_i} = \max\{a_i^j, j = 1, \ldots, m\}.$$

Then the problems (28) and (25) can be replaced by the following problems, respectively:

$$\text{minimize } f^{\bar{q}}(y) \text{ subject to } y \in D \tag{30}$$

$$\text{minimize } f_k(x^1, \ldots, x^k) \text{ subject to } x^j \in D, \quad j = 1, \ldots, k, \tag{31}$$

where

$$D = \{x \in \mathbb{R}^n : \bar{a_i} \leqslant x_i \leqslant \bar{b_i}, \quad i = 1, \ldots, n\}.$$

6. Results of Numerical Experiments with Cluster Functions

We tested the proposed algorithm for solving the cluster analysis problem using real-world datasets. Three test sets which have been used for testing of the proposed algorithm are: (i) 1060 and (ii) 3068 points in the plane, taken from the TSP-LIB data base ([25]), (iii) 19-dimensional image segmentation data ([13]).

In numerical experiments we consider Euclidean norm, that is $p = 2$. In this case the clustering problem is also known as minimum sum-of-squares-clustering.

The results of numerical experiments are presented in Table 2. In this table we report best known value for the global minimum from [16, 18], as well as results obtained by three most effective heuristic algorithms for

Table 2. Results for the clustering problem.

Datasets	Number of clusters	Best known value	K–M	J–M +	VNS	DG + CAM
TSPLIB1060	10	1.75484×10^9	0.03	0.19	0.04	0.00
	20	7.91794×10^8	3.96	0.04	0.83	0.00
	30	4.81251×10^8	10.51	1.82	0.42	0.53
	50	2.55509×10^8	16.58	3.84	1.70	0.70
TSPLIB3038	2	0.31688×10^{10}	0.00	0.00	0.00	0.00
	3	0.21763×10^{10}	1.55	1.29	1.37	0.00
	4	0.14790×10^{10}	0.03	0.00	0.00	0.00
	5	0.11982×10^{10}	0.12	0.11	0.10	0.00
	6	0.96918×10^9	1.22	1.98	0.01	0.00
	7	0.83966×10^9	1.65	1.48	0.73	1.73
	8	0.73475×10^9	1.90	1.48	0.62	0.00
	9	0.64477×10^9	1.47	0.99	0.11	0.00
	10	0.56025×10^9	2.44	1.81	0.06	0.00
	20	0.26681×10^9	3.16	2.60	0.09	0.14
	30	0.17557×10^9	4.04	2.89	0.91	0.03
	40	0.12548×10^9	6.21	3.49	0.93	−0.38
	50	0.98400×10^8	6.79	3.51	0.33	0.11
Image seg.	2	0.35606×10^8	0.17	0.00	0.00	−0.01
	3	0.27416×10^8	1.38	0.37	0.46	−0.02
	4	0.19456×10^8	25.32	0.00	0.00	−0.03
	5	0.17143×10^8	23.48	0.81	0.10	−0.03
	6	0.15209×10^8	16.41	2.47	2.87	0.78
	7	0.13404×10^8	12.49	5.39	2.53	0.49
	8	0.12030×10^8	10.94	6.29	0.16	0.55
	9	0.10784×10^8	13.58	6.92	0.32	0.61
	10	0.97952×10^7	15.83	3.65	0.18	1.71
	20	0.51283×10^7	34.63	3.46	0.97	0.35
	30	0.35076×10^7	44.68	5.62	0.40	−0.02
	40	0.27398×10^7	53.43	4.90	0.43	0.73
	50	0.22249×10^7	58.51	4.32	0.35	0.44

finding of minimum of sum-of-squares clustering problems: $k-m$, $j-m+$ and Variable Neighborhood Search (VNS) algorithms (see [16–18]). We present the average value of results obtained by these three algorithms by 10 restarts. The % error E reported in the table are calculated as

$$E = \frac{(\bar{f} - f_{\text{opt}})}{f_{\text{opt}}} \cdot 100,$$

where $\bar{f}$ and f_{opt} denote the solution found by the algorithm and the best known solution. Negative % errors mean that the algorithm has improved the best known solution.

Since in the proposed algorithm the starting points are updated by the algorithm itself we report only one value obtained by this algorithm.

Results presented in Table 2 show that in 6 cases the proposed algorithm improved the best known solution. In many cases this algorithm calculated

the best known solution and in all other cases its results are comparable with results obtained by other algorithms. Results from this table confirm the proposed algorithm is quite powerful algorithm for solving clustering problems in large datasets.

7. Conclusion

In this paper the algorithm for solving global optimization problems has been proposed. This algorithm is based on the discrete gradient method where line search procedure is partly replaced by the multidimensional global search. The global search is carried out by the cutting angle method. Results of numerical experiments are presented which demonstrate that this method allows one to improve results obtained by the discrete gradient method.

The proposed algorithm has been applied for solving the cluster analysis problems in three datasets from the literature. The use of the algorithm allows one in many cases to get best known values for global minimum. Moreover, in six cases the new best values have been found. In all other cases the results achieved by the proposed algorithm are better or comparable with the results obtained by other most effective algorithms of the cluster analysis.

Results of numerical experiments confirm that the proposed algorithm is effective for finding a good local minimum at least for the problems considered in this paper.

Acknowledgement

The authors would like to thank an anonymous referee whose comments have improved this paper.

References

1. Andramonov, M.Y., Rubinov, A.M. and Glover, B.M. (1997), Cutting angle method for minimizing increasing convex-along-rays functions, Research Report 97/7, SITMS, University of Ballarat.
2. Andramonov, M.Y., Rubinov, A.M. and Glover, B.M. (1999), Cutting angle method in global optimization, *Applied Mathematics Letters* 12, 95–100.
3. Bagirov, A.M. (1999), Derivative-free methods for unconstrained nonsmooth optimization and its numerical analysis, *Investigacao Operacional* 19, 75–93.
4. Bagirov, A.M. (2002), A method for minimization of quasidifferentiable functions, *Optimization Methods and Software* 17(1), 31–60.
5. Bagirov, A.M. and Gasanov, A.A. (1995), A method of approximating a quasidifferential, *Russian Journal of Computational Mathematics and Mathematical Physics* 35(4), 403–409.

6. Bagirov, A.M. and Rubinov, A.M. (2000), Global minimization of increasing positively homogeneous functions over the unix simplex, *Annals of Operations Research* 98, 171–187.
7. Bagirov, A.M. and Rubinov, A.M. (2001), Modified versions of the cutting angle method. In: Hadjisavvas, N. and Pardalos, P.M. (eds.), *Advances in Convex Analysis and Global Optimization*, Kluwer Academic Publishers, Dordrecht, 245–268.
8. Bagirov, A.M. and Rubinov, A.M. (2003), Cutting angle method and a local search, *Journal of Global Optimization* 27, 193–213.
9. Bagirov, A.M., Rubinov, A.M., Soukhoroukova, N.V. and Yearwood J. (2003), Unsupervised and supervised data classification via nonsmooth and global optimization, *TOP (Journal of Spanish Operations Research Society)* 11, 1–75.
10. Bagirov, A.M., Rubinov, A.M., Soukhoroukova, N.V. and Yearwood, J. (2003), An algorithm for clustering based on nonsmooth optimization techniques, *International Transactions in Operational Research* 10, 611–617.
11. Bagirov, A.M. and Zhang, J. (2003), Comparative analysis of the cutting angle method and simulated annealing methods in global optimization, *Optimization* 52, 363–378.
12. Batten, L.M. and Beliakov, G. (2002), Fast algorithm for the cutting angle method of global optimization, *Journal of Global Optimization* 24(2), 149–161.
13. Blake, C.L. and Merz, C.J. (1998), UCI Repository of machine learning databases, Department of Information and Computer Science, University of Calirfornia, Irvine. Also available at: http://www.ics.uci.edu/mlearn/MLRepository.html.
14. Bradley, P.S. and Mangasarian, O.L. (2000), *k*-Plane clustering, *Journal of Global Optimization* 16, 23–32.
15. Brimberg, J., Love, R.F. and Mehrez, A. (2002), Location/Allocation of queuing facilities in continuous space using minsum and minmax criteria. In: Pardalos, P., Migdalas, A. and Burkard, R. (eds.), *Combinatorial and Global Optimization*, World Scientific.
16. Hansen, P. and Mladenovic, N. (2001), *j*-means: a new local search heuristic for minimum sum-of-squares clustering, *Pattern Recognition* 34(2), 405–413.
17. Hansen, P. and Mladenovic, N. (2001), Variable neighborhood search: principles and applications, *European Journal of Operational Research* 130, 449–467.
18. Hansen, P., Ngai, E., Cheung, B.K. and Mladenovic, N. (August, 2002), Analysis of global *k*-means, an incremental heuristic for minimum sum-of-squares clustering, *Les Cahiers du GERAD*, G-2002-43.
19. Hedar, A.R. and Fukushima, M. (2002), Hybrid simulated annealing and direct search method for nonlinear unconstrained global optimization, *Optimization Methods and Software* 17(5), 891–912.
20. Horst, R., Pardalos, P.M. and Thoai, N.V. (1995), *Introduction to Global Optimization, Nonconvex Optimization and Its Applications*, Vol. 3, Kluwer Academic Publishers, Dordrecht.
21. Lim, K.F., Beliakov, G. and Batten, L.M. (2003), Predicting molecular structures: application of the cutting angle method. *Physical Chemistry Chemical Physics* 5, 3884–3890.
22. Mifflin, R. (1977), Semismooth and semiconvex functions in constrained optimization, *SIAM Journal on Control and Optimization* 15(6), 959–972.
23. Neumaier, A., Optimization test problems. Available in: solon.cma.univie.ac.at/neum/glopt.html.
24. Pinter, J. (1996), *Global Optimization in Action*, Kluwer Academic Publishers, Dordrecht.
25. Reinelt, G. (1991), TSP-LIB-A Traveling Salesman Library, *ORSA Journal of Computing* 3, 376–384.
26. Rubinov, A.M. (2000), *Abstract Convexity and Global Optimization*, Kluwer Academic Publishers, Dordrecht.

27. Rubinov, A.M. and Andramonov, M. (1999), Lipschitz programming via increasing convex along-rays functions, *Optimization Methods and Software* 10, 763–781.
28. Yiu, K.F.C., Liu, Y. and Teo, K.L. (2004), A hybrid descent method for global optimization, *Global Optimization* 28(2), 229–238.