Lipschitz and Hölder global optimization using space-filling curves*

D. Lera†, Ya.D. Sergeyev‡

Abstract

In this paper, the global optimization problem $\min_{y \in S} F(y)$ with $S = [a, b]$, $a, b \in \mathbf{R}^N$, and $F(y)$ satisfying the Lipschitz condition, is considered. To deal with it four algorithms are proposed. All of them use numerical approximations of space-filling curves to reduce the original Lipschitz multidimensional problem to a univariate one satisfying the Hölder condition. The Lipschitz constant is adaptively estimated by the introduced methods during the search. Local tuning on the behavior of the objective function and a newly proposed technique, named *local improvement*, are used in order to accelerate the search. Convergence conditions are given. A theoretical relation between the order of a Hilbert space-filling curve approximation used to reduce the problem dimension and the accuracy of the resulting solution is established, as well. Numerical experiments carried out on several hundreds of test functions show a quite promising performance of the new algorithms.

Key Words. Global optimization, Lipschitz and Hölder functions, local information, space-filling curves approximations, acceleration.

*This research was supported by the following grants: Italian PRIN project 2007-9PLLN7, Russian RFBR grant 07-01-00467-a, the Russian Federal Program "Scientists and Educators in Russia of Innovations", contract number 02.740.11.5018, and the grant 4694.2008.9 for supporting the leading research groups awarded by the President of the Russian Federation.

†Dipartimento di Matematica e Informatica, Università di Cagliari, Cagliari, Italy

‡Corresponding author, Dipartimento di Elettronica, Informatica e Sistemistica, Università della Calabria and the Institute of High Performance Computing and Networking of the National Research Council of Italy, Via Pietro Bucci 42C, 87036 Rende (CS), Italy; and Software Department, University of Nizhni Novgorod, Gagarin Av. 23, Nizhni Novgorod, Russia

1 Introduction

Let us consider the following global optimization problem:

$$\min\{F(y) : \ y \in [a,b]\}, \tag{1.1}$$

where $[a,b]$ is a hypercube in $\mathbf{R}^N$ and F is a multiextremal function that satisfies the Lipschitz condition

$$|F(y') - F(y'')| \le L\|y' - y''\|, \qquad y', y'' \in [a,b], \tag{1.2}$$

with a constant L, $0 < L < \infty$, generally unknown; $\|\cdot\|$ denotes the Euclidean norm. In the literature, there exist many methods for the resolution of the problem (1.1), (1.2) (see [2, 10, 14, 17, 23, 27, 29]). In this paper, we consider an approach based on the reduction of the dimension by using space-filling curves. These curves, first introduced by Peano (1890) and Hilbert (1891) (see [3, 16, 18, 24, 27]), fill in the hypercube $[a,b] \subset \mathbf{R}^N$, i.e., they pass through every point of $[a,b]$, and this gave rise to the term space-filling curves. More precisely, in this work we consider the Hilbert space-filling curves. A Hilbert curve emerges as the limit object generated by an iterative process: it is a fractal constructed using the principle of self-similarity. Examples of construction of these curves are given in Figs. 1–3.

It has been shown (see [3],[24]–[27]) that, by using space filling curves, the multidimensional global minimization problem (1.1), (1.2) is turned into a one-dimensional problem. In particular, Strongin proved in [24] that finding the global minimum of the Lipschitz function $F(y), y \in R^N$, over a hypercube is equivalent to determining the global minimum of the function $f(x)$:

$$f(x) = F(p(x)), \qquad x \in [0,1], \tag{1.3}$$

where $p(x)$ is the Hilbert curve. Moreover, the Hölder condition

$$|f(x') - f(x'')| \le H|x' - x''|^{1/N}, \qquad x', x'' \in [0,1], \tag{1.4}$$

holds for the function f with the constant

$$H = 2L\sqrt{N+3}, \tag{1.5}$$

where L is the Lipschitz constant of the multidimensional function $F(y)$.

Thus, we can solve the problem (1.1), (1.2) by using algorithms proposed for minimizing functions (1.3), (1.4) in one dimension. Naturally, in order to realize the passage from the multi-dimensional problem to the one-dimensional one, computable approximations to the Hilbert curve should be employed in the numerical algorithms. In Fig. 1 and Fig. 2 we can see an approximation of level four in the domain $[-1,1] \times [-1,1] \subset \mathbf{R}^2$ and in Fig. 3 an approximation of level two. Hereinafter we use the designation $p_m(x)$ for the m-level approximation of the Hilbert curve.

In this paper, we generalize the algorithm introduced in [12] for minimizing Hölderian one-dimensional functions. Since the algorithm given in [12] supplies a

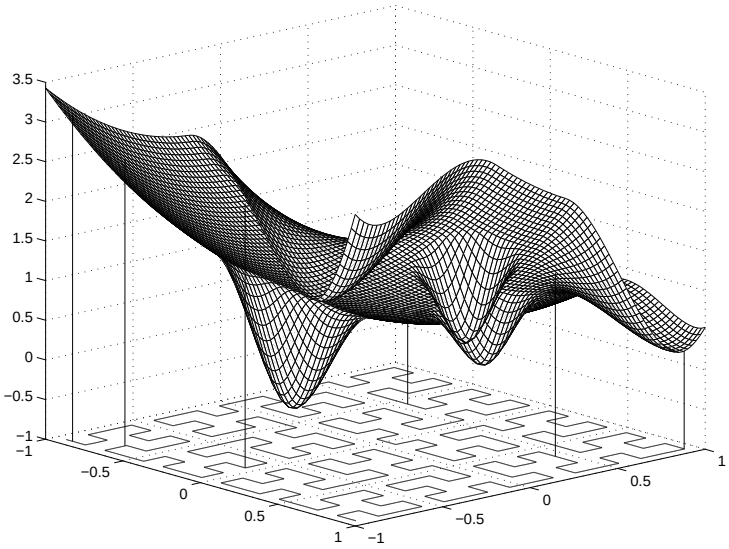

Figure 1: *Usage of the Hilbert curve approximation of level 4 for global optimization. The objective function has been evaluated at six points that belong to the curve and, naturally, to the original two-dimensional search region as well.*

lower bound of the objective function over the one-dimensional interval, by using the m-approximation of the Hilbert curve we can calculate, at each iteration, a lower bound of the function $f(x) = F(p_m(x))$ on the curve $p_m(x)$, $x \in [0, 1]$, if the Hölder constant H is known. However, since we are interested in the behavior of the original function $F(y)$ from (1.1) over the multi-dimensional domain $[a, b]$, the problem of obtaining an estimate of the global minimum in $[a, b]$, given a lower bound over the curve $p_m(x)$, arises. This problem is studied in Section 2.

The second issue considered in the paper is related to the problem of availability of the Lipschitz constant L. Since in real-life problems, the Lipschitz constant very often is not known, in this work we describe two optimization techniques that do not require the a priori knowledge of the constant. The first of them estimates the *global* Hölder constant over the entire Hilbert curve during the search; the second adaptively estimates the *local* Hölder constants in different subintervals of the search region during optimization. This procedure performs a *local tuning* on the behavior of the objective function balancing global and local information obtained during the search.

The local tuning is crucial in the study of the third issue we tackle in the paper – acceleration of the search. It has been shown for different classes of global optimization problems (see [12, 20, 21, 23, 27]) that the local tuning technique can accelerate the search significantly. In the algorithms we propose in this paper, the local tuning is used together with a new tool we introduce to quicken the search. This new technique (called hereinafter *local improvement*) forces the global optimization method to make a local improvement of the best approximation of the global mini-

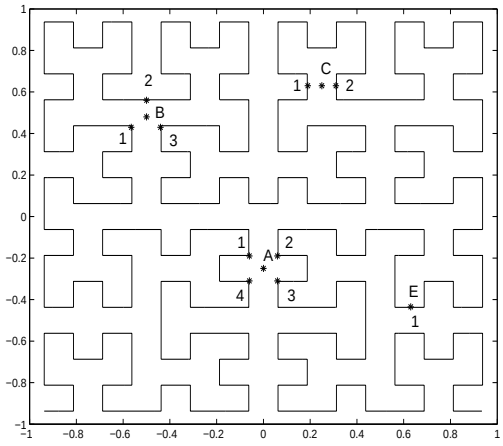

Figure 2: *Approximation of a point*

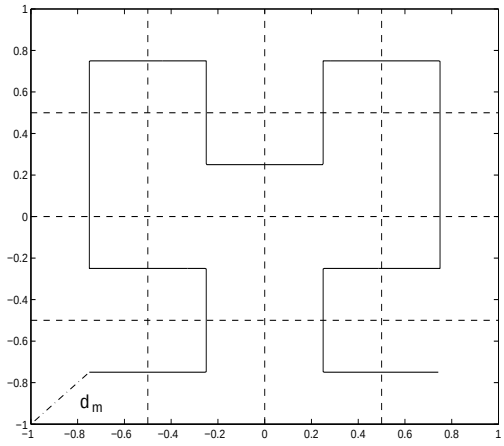

Figure 3: *The distance d_m*

mum immediately after a new approximation better than the current one is found. Obviously (see [10, 14, 15, 24, 27]) Lipschitz methods automatically execute such an improvement. However, we show that it is very prolonged in time and the local improvement quickens the search significantly giving us so the second tool (together with the local tuning) for acceleration of the global optimization process.

This paper has the following structure. In Section 2, a theorem that provides a lower bound for the objective function $F(y)$ in its multi-dimensional domain given a lower bound for the one-dimensional function $F(p_m(x))$ on the curve $p_m(x)$ is presented and a general scheme describing new algorithms for solving the problem (1.1), (1.2) is then described. In Section 3, four specific algorithms belonging to the scheme are introduced and their convergence conditions are established. Section 4 presents results of some numerical experiments on 600 test functions taken from literature. Finally, Section 5 concludes the paper.

2 A lower bound for the multi-dimensional objective function and a general algorithmic scheme

Reduction of the N-dimensional problem (1.1), (1.2) by Hilbert curves to the one-dimensional one allows us to use univariate optimization algorithms for solving the problem (1.3), (1.4). If such a method uses the Hilbert curve $p_m(\cdot)$ of level m and provides a lower bound M_m^* for the one-dimensional function $f(x)$ then this value will be a lower bound for the function $F(y)$ but only along the curve $p_m(\cdot)$. Naturally, the following question becomes very important in this connection: Can we establish a lower bound for the function $F(y)$ over the entire multi-dimensional search region $[a, b]$? The following theorem answers this question (without loss of generality we use $a_i = -1, b_i = 1, 1 \le i \le N$, hereinafter).

Theorem 2.1 *Let M_m^* be a lower bound along the Hilbert curve $p_m(x)$ for a multi-*

dimensional function $F(y)$ satisfying condition (1.2), i.e.,

$$M_m^* \leq F(p_m(x)), \qquad x \in [0,1]. \tag{2.1}$$

Then the value $M = M_m^* - 2^{-(m+1)}L\sqrt{N}$ is a lower bound for $F(y)$ over the entire region $[a,b]$, i.e.,

$$M \leq F(y), \qquad y \in [a,b]. \tag{2.2}$$

Proof. Every point $y \in [a,b]$ is approximated by points (called *images*) $\alpha_i(y), 1 \leq i \leq J$, on the curve minimizing the Euclidean distance from y to the curve

$$\alpha_i(y) = \arg\min\{\|y - \tilde{y}\| : \tilde{y} = p_m(x), x \in [0,1]\}, \quad 1 \leq i \leq J.$$

It has been shown (see [24]–[27]) that the number of images, J, ranges between 1 and 2^N. For example, for $N = 2$ (see Fig. 2), the point A has four images on the curve, B has three images, C has two, and E has only one image.

Let us consider now a point $y \in [a,b]$ and its approximation $\alpha(y)$ on the Hilbert curve. Since the function $F(y)$ satisfies the Lipschitz condition (1.2), we have

$$|F(y) - F(\alpha(y))| \leq L\|y - \alpha(y)\|,$$

$$F(y) \geq F(\alpha(y)) - L\|y - \alpha(y)\|.$$

The point $\alpha(y)$ belongs to the Hilbert curve and M_m^* is a lower bound for $F(y)$ along the curve. Thus, it follows from (2.1) that $F(\alpha(y)) \geq M_m^*$ and then

$$F(y) \geq M_m^* - L\|y - \alpha(y)\| \geq M_m^* - Ld_m, \tag{2.3}$$

where the designation

$$d_m = \max_{y \in [a,b]} \|y - \alpha(y)\|$$

has been used. It is easy to understand how the distance d_m can be calculated. The Hilbert curves establish a correspondence between subintervals of the curve and N-dimensional sub-cubes of $[a,b] \subset \mathbf{R}^N$ (these sub-cubes for $N = 2$ are shown in Fig. 3) with the side equal to 2^{-m} (see [24],[27]). Thus, d_m is equal to the distance between the center of a sub-cube and one of its vertex, i.e., $d_m = 2^{-(m+1)}\sqrt{N}$. From this result and (2.3) we obtain the final estimate

$$F(y) \geq M_m^* - L2^{-(m+1)}\sqrt{N} = M$$

that concludes the proof. □

Thus, one-dimensional methods from [12] constructing at each iteration auxiliary functions providing a lower bound of the univariate objective function can be used as a basis for developing new methods for solving the multi-dimensional problem (1.1), (1.2). In order to present the new multi-dimensional algorithms, we first describe a General Algorithmic Scheme for solving problem (1.1), (1.2) in a compact form. Then, by specifying Step 2 and Step 4, we give four different algorithms.

Algorithm 2.1 *(General Algorithmic Scheme)*

STEP 0. Set $x^1 = 0$, $x^2 = 1$ and compute the values of the function $z^j = f(x^j) = F(p_m(x^j))$, $j = 1, 2$, where $p_m(x)$ is the m-approximation of the Hilbert curve. After executing k trials the choice of new trial points is done as follows.

STEP 1. Renumber the trial points $x^1,\ x^2,\ \ldots,\ x^k$ of the previous iterations by subscripts so that

$$0 = x_1 < x_2 < \ldots < x_{k-1} < x_k = 1. \tag{2.4}$$

Thus, two numerations are used in the algorithm. Superscripts indicate the number, i, of the iteration at which the point x^i has been produced and subscripts are used to order trial points during each iteration.

STEP 2. Call the function HOLDER-CONST($setH$) in order to compute the value h_i being an estimate of the Hölder constant of $f(x)$ over the interval $[x_{i-1}, x_i]$, $i = 2, ...k$. The parameter $setH$ can assume the values 1 or 2.

STEP 3. For each interval (x_{i-1}, x_i), $i = 2, ..., k$, compute the point y_i and the characteristic M_i:

$$y_i = \frac{1}{2}(x_i + x_{i-1}) - \frac{z_i - z_{i-1}}{2rh_i(x_i - x_{i-1})^{\frac{1-N}{N}}}, \tag{2.5}$$

$$M_i = \min\left\{z_{i-1} - rh_i(y_i - x_{i-1})^{1/N},\ z_i - rh_i(x_i - y_i)^{1/N}\right\}, \tag{2.6}$$

where $z_j = F(p_m(x_j))$, $j = 1, ..., k$, and $r > 1$ is a reliability parameter of the method.

STEP 4. Call the function SELECT($setINT$) that returns an interval (x_{t-1}, x_t) for the next possible trial. The parameter $setINT$ can assume the values 1 or 2.

STEP 5. If

$$|x_t - x_{t-1}|^{1/N} \leq \varepsilon, \tag{2.7}$$

where $\varepsilon > 0$ is a given search accuracy, then calculate an estimate of the global minimum as

$$F_k^* = \min\{z_i\ :\ 1 \leq i \leq k\}$$

and **STOP**. Otherwise, execute the next trial at the point

$$x^{k+1} = y_t \tag{2.8}$$

set $k = k + 1$ and go to **STEP 1**.

Let us comment upon the general scheme. In the course of its work the Scheme 2.1 constructs an auxiliary piecewise function in one dimension that after executing k trials is

$$L^k(x) = l_i(x), \quad \text{for } x \in [x_{i-1}, x_i], \quad i = 2, ..., k \tag{2.9}$$

where, for $\bar{h}_i = rh_i$:

$$l_i(x) = \max\left\{f(x_{i-1}) - \bar{h}_i(x - x_{i-1})^{1/N},\ f(x_i) - \bar{h}_i(x_i - x)^{1/N}\right\}. \tag{2.10}$$

If the constant $\bar{h}_i$ is equal or larger than the Hölder constant H, then it follows from (1.4) that the function $L^k(x)$ is a low-bounding function for $f(x)$ for every interval $[x_{i-1}, x_i]$, $i = 2, ..., k$, i.e.,

$$L^k(x) \leq f(x), \qquad \forall x \in [0, 1]. \tag{2.11}$$

In the Scheme 2.1, for each interval $[x_{i-1}, x_i]$, $i = 2, ..., k$, we approximate the point

$$\tilde{y}_i = \text{argmin}\{l_i(x) :\ x \in [x_{i-1}, x_i]\}$$

by the point y_i from (2.5) found as intersection of the lines $r_{left}(x)$ and $r_{right}(x)$ (see [12] and an illustration in Fig. 4)

$$r_{left}(x) = -\bar{h}_i(x_i - x_{i-1})^{\frac{1-N}{N}}x + \bar{h}_i(x_i - x_{i-1})^{\frac{1-N}{N}}x_{i-1} + f(x_{i-1}),$$

$$r_{right}(x) = \bar{h}_i(x_i - x_{i-1})^{\frac{1-N}{N}}x - \bar{h}_i(x_i - x_{i-1})^{\frac{1-N}{N}}x_i + f(x_i).$$

The characteristic M_i in (2.6) represents the minimum among the values of the auxiliary functions

$$l_i^-(x) = f(x_{i-1}) - \bar{h}_i(x - x_{i-1})^{1/N}, \quad l_i^+(x) = f(x_i) - \bar{h}_i(x_i - x)^{1/N}$$

evaluated at the point y_i. By making use of the Hilbert curves we have a correspondence between a cube in dimension N and an interval in one dimension (see Fig. 4).

3 Four algorithms

In order to obtain from the general scheme 2.1 a global optimization algorithm, it is necessary to define the routines used in Steps 2 and 4 of the scheme. This section proposes four specific algorithms effecting this operation in different ways. In Step 2, we can make two different choices of the constant h_i, according to the value of the parameter $setH$. For $setH = 1$, we consider a procedure that estimates the global constant during the search for each iteration k, whereas for $setH = 2$ we consider another procedure that determines estimates of the local Hölder constants

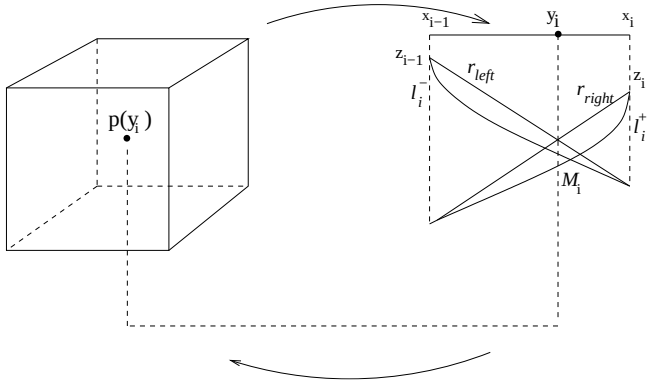

Figure 4: *The points y_i and $p(y_i)$.*

at subintervals of the search domain in the course of the work of the algorithm. Let us describe both procedures.

HOLDER-CONST(1)

Set

$$h_i = \max\{\xi, h^k\}, \qquad i = 2, ..., k \tag{3.1}$$

where $\xi > 0$ is a small number that takes into account our hypothesis that $f(x)$ is not constant over the interval $[0, 1]$ and the value h^k is calculated as follows

$$h^k = \max\{m_i : i = 2, ..., k\} \tag{3.2}$$

with

$$m_i = \frac{|z_i - z_{i-1}|}{|x_i - x_{i-1}|^{1/N}}, \quad i = 2, ..., k. \tag{3.3}$$

HOLDER-CONST(2)

Set

$$h_i = \max\{\lambda_i, \gamma_i, \xi\}, \qquad i = 2, ..., k \tag{3.4}$$

with

$$\lambda_i = \max\{m_{i-1}, m_i, m_{i+1}\}, \quad i = 3, ..., k-1, \tag{3.5}$$

where m_i is from (3.3), and when $i = 2$ and $i = k$ we consider only m_2, m_3, and m_{k-1}, m_k respectively. The value

$$\gamma_i = h^k \frac{|x_i - x_{i-1}|}{X^{max}}, \tag{3.6}$$

where h^k is from (3.2) and

$$X^{max} = \max\{|x_i - x_{i-1}|^{1/N}, \ i = 2, ..., k\}.$$

The parameter $\xi > 0$ has the same sense as in HOLDER-CONST(1).

In the HOLDER-CONST(1) the same estimates h_i are used over the whole search region for $f(\cdot)$. However, global estimates of the constant may provide a poor information about the behavior of the objective function over every small subinterval $[x_{i-1}, x_i] \subset [0,1]$: if the local Hölder constant related to the interval $[x_{i-1}, x_i]$ is significantly less than the global constant then the methods using only (3.1) work slowly over such intervals (see [12],[20],[21],[27]). In the HOLDER-CONST(2) we consider two different components λ_i and γ_i, in (3.4), that take into account respectively local and global information obtained during the previous iterations. When the interval $[x_{i-1}, x_i]$ is large, the local information is not reliable and the global part γ_i increases. When $[x_{i-1}, x_i]$ is small, then the local information becomes relevant, γ_i is small, and the local component λ_i is used. It has been proved that, for a number of global optimization algorithms, usage of the local information can accelerate the search significantly (see [12],[20],[21],[27]).

In STEP 4, we choose the interval for performing a new trial. As regards to the choice of this interval we have considered both the traditional criterion used in [12] (corresponding to the value 1 of the parameter $SetINT$) and a new criterion that we shall call the *local improvement*, corresponding to the value 2 of the parameter $SetINT$.

SELECT(1)

Select the interval (x_{t-1}, x_t) such that

$$M_t = \min\{M_i \ : \ 2 \le i \le k\} \tag{3.7}$$

and t is the minimal number satisfying (3.7).

SELECT(2)

$flag$ is a parameter initially equal to zero.

$imin$ is the index corresponding to the current estimate of the minimal value of the function,

that is: $z_{imin} = f(x_{imin}) \le f(x_i)$, $i = 1, ..., k$.

z^k is the result of the last trial corresponding to a point x_j in the line (2.4), i.e., $x^k = x_j$.

IF (flag=1) THEN

IF $z^k < z_{imin}$ THEN $imin = j$.

Local improvement: Alternate the choice of the interval (x_{t-1}, x_t) among $t = imin+1$ and $t = imin$, if $imin = 2, ..., k-1$, (if $imin = 1$ or $imin = k$ take $t = 2$ or $t = k$ respectively) in such a way that for $\delta > 0$ it follows

$$|x_t - x_{t-1}| > \delta. \tag{3.8}$$

ELSE (flag=0)

$t = \text{argmin}\{M_i \ : \ 2 \le i \le k\}$

ENDIF

flag=NOTFLAG(flag)

In SELECT(1), at each iteration, we continue the search at an interval corresponding to the minimal value of the characteristic M_i, $i = 2, ..., k$ (see (3.7)). In this way it is possible that the search goes on, for a certain number of iterations, at subregions of the domain "distant" from the best found approximation to the global solution, and only successively concentrates at the interval containing a global minimizer. However, very often it is of crucial importance to be able to find a good approximation of the global minimum in the lowest number of iterations. Due to this reason, in SELECT(2) we take into account the criterion (3.7) used in SELECT(1) and related to the minimal characteristic, but we alternate it with a new selection method that forecast to continue the search in the part of the domain corresponding to the best value of the objective function found up to now. The parameter "flag" assuming values 0 or 1 allows us to alternate the two methods.

More precisely, in SELECT(2) first we identify the index *imin* corresponding to the current minimum among the found values of the objective function, and then we select the interval on the right of the current point of minimum x_{imin}, that is (x_{imin}, x_{imin+1}), or the interval on the left of x_{imin}, (x_{imin-1}, x_{imin}). SELECT(2) keeps working alternatively to the right and the left of the current x_{imin} until a new trial point with value less than z_{imin} is found. The search moves from the right to the left of the best found approximation trying to improve it. Since we are not sure that the found best approximation is really located in the neighborhood of the global minimizer, the local improvement is alternated in SELECT(2) with the usual rule (3.7) providing the global search of new subregions possibly containing the global solution. The parameter δ defines the width of the intervals subdivided during the phase of the local improvement.

Thus, by considering the Scheme 2.1 with the function HOLDER-CONST(1) and SELECT(1) or SELECT(2) we have two algorithms that we shall call AG (the *A*lgorithm with *G*lobal approximation of the Hölder constant) and AGI (the *A*lgorithm with *G*lobal approximation of the Hölder constant and local *I*mprovement). The Scheme 2.1 with the function HOLDER-CONST(2) and SELECT(1) or SELECT(2) gives two other algorithms that we shall call AL (the *A*lgorithm with the *L*ocal tuning) and ALI (the *A*lgorithm with the *L*ocal tuning and local *I*mprovement).

Let us study now the convergence properties of the four proposed algorithms. Theorem 2.1 linking the multi-dimensional global optimization problem (1.1), (1.2) to the one-dimensional problem (1.3), (1.4) allows us to concentrate our attention on the one-dimensional methods on the one-dimensional curve. We shall study properties of an infinite (i.e., $\varepsilon = 0$ in (2.7)) sequence $\{x^k\}$, $x^k \in [0, 1]$, $k \geq 1$, of trial points generated by the Scheme 2.1.

Definition 3.1 *Convergence to a point $x' \in (0, 1)$ is said to be bilateral if there exist two subsequences of $\{x^k\}$ converging to x' one from the left, the other from the right.*

Theorem 3.1 *Assume that the objective function $f(x)$ satisfies the condition (1.4), and let x' be any limit point of $\{x^k\}$ generated by the AG or by the AL. Then the following assertions hold:*

1. *convergence to x' is bilateral, if $x' \in (0,1)$;*

2. *$f(x^k) \geq f(x')$, for any $k \geq 1$;*

3. *if there exists another limit point $x'' \neq x'$, then $f(x'') = f(x')$;*

4. *if the function $f(x)$ has a finite number of local minima in $[0,1]$, then the point x' is locally optimal;*

5. *(Sufficient conditions for convergence to a global minimizer). Let x^* be a global minimizer of $f(x)$. If there exists an iteration number k^* such that for all $k > k^*$ the inequality*

$$\bar{h}_{j(k)} > H_{j(k)} \tag{3.9}$$

holds, where $\bar{h}_{j(k)} = rh_{j(k)}$, in which $h_{j(k)}$ is an estimate (calculated at STEP 2 of the algorithm) of the Hölder constant $H_{j(k)}$ for the interval $[x_{j(k)-1}, x_{j(k)}]$ containing x^*, and r is the reliability parameter of the method. Then the set of limit points of the sequence $\{x^k\}$ coincides with the set of global minimizers of the function $f(x)$.

Proof. Theorem is proved analogously to proofs of theorems 3.1–3.5 from [12]. □

Note that assertion 4 in Theorem 3.1 describes conditions of local optimality on the curve. In the multi-dimensional region the point $p_m(x')$ can be a point which is not a local optimum. Such situations for the class of "Dived the Best" algorithms the Scheme 2.1 belongs to have been studied in detail in [19]. It is also important to emphasize that assertion 5 regards the global optimum x^* of the one-dimensional problem. Since the global minimizer, y^*, in the N-dimensional space can have up to 2^N images on the curve (see Fig. 2) and in the process of optimization a curve $p_m(x)$ is used, in order to have convergence to the point y^* it is sufficient to have convergence to one of the images of y^* on the curve. Of course, in the limit case ($m \to \infty$ and $\varepsilon = 0$ in (2.7)) if condition (3.9) is satisfied for one of the images, all global minimizers will be found. But in practice we work with a finite $m < \infty$ and $\varepsilon > 0$, i.e., with a finite trial sequence, and the search can stop after finding the only image of y^* providing nevertheless the required approximation of the global minimizer y^*. This effect leads to a serious acceleration of the search and the local improvement technique used in the methods AGI and ALI have been introduced in order to enforce this effect.

Theorem 3.2 *Assertions 1–5 of Theorem 3.1 hold for the algorithms AGI and ALI for a fixed finite $\delta > 0$ and $\varepsilon = 0$, where δ satisfies (3.8) and ε is from (2.7).*

Proof. Since $\delta > 0$ and $\varepsilon = 0$, the algorithms AGI and ALI use the local improvement only at the initial stage of the search until the selected interval (x_{t-1}, x_t) is greater than δ. When $|x_t - x_{t-1}| \leq \delta$ the interval cannot be divided by the local improvement and the selection criterion (3.7) is used. Thus, since the one-dimensional search region has a finite length and δ is a fixed finite number, there exists a finite iteration number j such that at all iterations $k > j$ only selection criterion (3.7) will be used.

As a result, at the remaining part of the search, the methods AGI and ALI behave themselves as the algorithms AG and AL. This consideration concludes the proof. □

The following theorem ensures existence of the values of parameter r satisfying condition (3.9) providing so determining all global minimizers of $f(x)$ by the four proposed methods.

Theorem 3.3 *For any function $f(x)$ satisfying (1.4) with $H < \infty$ there exists a value r^* such that for all $r > r^*$ the four algorithms determine all global minimizers of the function $f(x)$ over the search interval $[0, 1]$.*

Proof. It follows from (3.1), (3.4), and the finiteness of $\xi > 0$ that approximations of the Hölder constant h_i in the four methods are always greater than zero. Since $H < \infty$ in (1.4) and any positive value of the parameter r can be chosen in the Scheme 2.1, it follows that there exists an r^* such that condition (3.9) will be satisfied for all global minimizers for $r > r^*$. This fact, due to Theorems 3.1 and 3.2, proves the theorem. □

4 Numerical experiments

In this section, we present numerical results of experiments executed for testing performance of the new algorithms AG, AL, AGI, and ALI. Four series of experiments have been executed with new algorithms. In all the experiments we have considered the FORTRAN implementation of the methods tested. Since in the considered kind of real life problems each evaluation of the objective function is usually a time consuming operation (see [10, 14, 17, 23, 24, 27, 29]), the number of function evaluations executed by the methods until the satisfaction of a stopping rule has been chosen as the main criterion of the comparison.

4.1 Classes of test functions

In the field of global optimization there exists an old set of "standard" test functions (see [4]). However, recently it has been discovered by several authors (see [1, 13, 28]) that these problems are not suitable for testing global optimization methods since the functions belonging to the set are too simple and methods can hardly miss the region of attraction of the global minimizer. As a consequence, the number of trials executed by methods is usually very small and, therefore, non-representative. These functions are especially inappropriate for testing algorithms proposed to work with the global optimization of real multiextremal black-box functions where it is necessary to execute many trials in order to better explore the search region and to reduce the risk of missing the global solution.

The algorithms proposed in this paper are oriented exactly on such a type of hard global optimization problems. Hence, more sophisticated and systematic tests are required to verify their performance. In our numerical experiments several classes of

N-dimensional test functions generated by the GKLS-generator from [9] (an example of a function generated by the GKLS can be seen in Fig. 1) have been used. This generator has several advantages that allow one to use it as a good tool for the numerical comparison of algorithms.

It generates classes of 100 test functions (see [9] for a detailed explanation, examples of its usage, etc.) with the same number of local minima and supplies a complete information about each of the functions: its dimension, the values of all local minimizers, their coordinates, regions of attraction, etc. It is possible to generate harder or simpler test classes easily. Only five parameters (see Table 1) should be defined by the user and the other parameters are generated randomly. An important feature of the generator consists of the complete repeatability of the experiments : if you use the same five parameters then each run of the generator will produce the same class of functions.

The GKLS-generator works by constructing test functions $F(y)$ in $\mathbf{R}^N$ taking a convex quadratic function $g(y)$, i.e., a paraboloid $g(y) = \|y - T\|^2 + t$, that is then distorted over the sets

$$R_k = \{y \in \mathbf{R}^N \mid \|y - P_k\| \leq r_k\}, \qquad k = 1, \cdots, m,$$

by assigning function values f_k at P_k. The general form of $F(y)$ is

$$F(y) = \begin{cases} C_k(y) & \text{if } y \in R_k, k \in \{1, \ldots, m\} \\ \|y - T\|^2 + t & \text{if } y \notin R_1 \bigcup, \cdots, \bigcup R_m. \end{cases} \tag{4.1}$$

where

$$\begin{aligned} C_k(y) \;=\; & \left(\frac{2}{r_k^2} \frac{\langle y - P_k, T - P_k \rangle}{\|y - P_k\|} - \frac{2}{r_k^3} A \right) \|y - P_k\|^3 \\ & + \left(1 - \frac{4}{r_k} \frac{\langle y - P_k, T - P_k \rangle}{\|y - P_k\|} + \frac{3}{r_k^2} A \right) \|y - P_k\|^2 + f_k \end{aligned} \tag{4.2}$$

with $A = \|T - P_k\|^2 + t - f_k$.

The generator gives the possibility to use several types of functions. In the described experiments, "cubic" continuous multiextremal functions have been used. In all series of experiments we have considered classes of 100 N-dimensional functions with 10 local minima over the domain $[-1, 1] \subset \mathbf{R}^N$. For each dimension $N = 2, 3, 4$ two test classes were considered: a simple class and a difficult one. Note (see Table 1) that a more difficult test class can be created either by decreasing the radius, r_g, of the approximate attraction region of the global minimizer, or by increasing the distance, d, from the global minimizer to the paraboloid vertex.

The experiments have been carried out by using one of the following two stopping criteria:

a) The value $\varepsilon = 0$ is fixed in the stopping rule (2.7) and the search terminates when a trial point falls in a ball B_i having a radius ρ and the center at the global minimizer of the considered function, i.e.,

$$B_i = \{y \in \mathbf{R}^N : \|y - y_i^*\| \leq \rho\}, \tag{4.3}$$

Class n.		N	m	f^*	d	r_g
1	simple	2	10	-1.0	0.66	0.33
2	hard	2	10	-1.0	0.90	0.20
3	simple	3	10	-1.0	0.66	0.33
4	hard	3	10	-1.0	0.90	0.20
5	simple	4	10	-1.0	0.66	0.33
6	hard	4	10	-1.0	0.90	0.20

Table 1: *Description of GKLS classes of test functions used in the experiments: the global minimum value, f^*; the distance from the global minimizer to the vertex of the paraboloid, d; the radius of the attraction region of the global minimizer, r_g.*

Class	N	Max trials			Average		
		Direct	LBDirect	AG	Direct	LBDirect	AG
1	2	127	165	239	68.14	70.74	90.06
2	2	1159	2665	938	208.54	304.28	333.14
3	3	1179	1717	3945	238.06	355.30	817.74
4	3	77951	85931	26964	5857.16	9990.54	3541.82
5	4	90000(1)	90000(15)	27682	>12206.49	>23452.25	3950.36
6	4	90000(43)	90000(62)	90000(1)	>57333.89	>65236.00	>22315.59

Table 2: *Results of experiments*

where y_i^* denotes the global minimizer of the i-th function of the test class, $i = 1, ..., 100$.

b) A value $\varepsilon > 0$ is fixed and the search terminates when the rule (2.7) is satisfied; then it is counted the number of functions of the class for which the method under consideration was able to put a point in the ball B_i, $i = 1, ..., 100$.

4.2 Comparison AG – Direct – LBDirect

In the first series of experiments we compare the basic algorithm AG with the original Direct algorithm proposed by Jones, Perttunen, and Stuckman (see [11]) and its recent locally biased modification LBDirect introduced by Gablonsky and Kelley (see [7, 8]). These methods have been chosen for comparison because they, just as the methods belonging to the Scheme 2.1, do not require the knowledge of the Lipschitz constant of the objective function and the knowledge of the objective function gradient. The FORTRAN implementations of these two methods described in [5, 7] and downloadable from [6] have been used in all the experiments. Parameters recommended by the authors have been used in both methods.

In all experiments the stopping rule a) was used with $\varepsilon = 0$ and $\rho = 0.01\sqrt{N}$ for classes 1 – 4 and $\rho = 0.02\sqrt{N}$ for classes 5 and 6, where ρ is from (4.3). In all the cases the maximal number of function evaluations has been taken equal to 90000; the parameter $\xi = 10^{-8}$, ξ from (3.1). The choices of the reliability parameter r, for all the experiments, are given in Appendix.

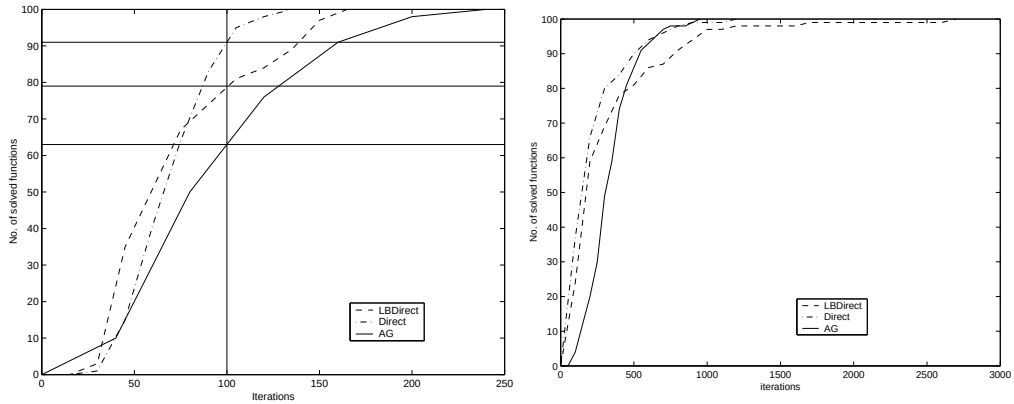

Figure 5: *Methods AG, Direct, and LBDirect, N=2. Class no.1, left; Class no.2, right*

Results of numerical experiments with the six GKLS tests classes from Table 1 are shown in Table 2. The columns "Max trials" report the maximal number of trials required for satisfying the stopping rule a) for all 100 functions of the class. The notation "90000 (j)" means that after 90000 function evaluations the method under consideration was not able to solve j problems. The "Average" columns in Table 2 report the average number of trials performed during minimization of the 100 functions from each GKLS class. The simbol ">" reflects the situations when not all functions of a class were successfully minimized by the method under consideration: that is the method stopped when 90000 trials had been executed during minimizations of several functions of this particular test class. In these cases, the value 90000 was used in calculations of the average value, providing in such a way a lower estimate of the average.

Fig.5 shows the behavior of the three methods for $N = 2$ on classes 1 and 2 from Table 1, respectively (for example, it can be seen in Fig.5-left that after 100 function evaluations the LBDirect has found the solution at 79 problems, Direct at 91 problems and the AG at 63 problems). Fig. 6 illustrates the results of the experiment for $N = 3$ on classes 3 and 4 from Table 1, respectively.

Fig.7 shows the behavior of the three methods for $N = 4$ on classes 5 and 6 from Table 1, respectively (it can be seen in Fig.7-left that after 10000 function evaluations the LBDirect has found the solution at 58 problems, Direct at 73 problems and the AG at 93 problems). It can be seen from Fig.7-left that after 90000 evaluations of the objective function the Direct method has not found the solution for 1 function, and the LBDirect has not found the solution for 15 functions of the class 5. Fig.7-right shows that the Direct and LBDirect methods were not able to locate after executing the maximal possible value of function evaluations, 90000, the global minimum of 43 and 62 functions of the class 6, respectively. The AG was able to solve all the problems in the classes 1–5; AG has not found the solution only in 1 function in the class 6.

As it can be seen from Table 1 and Figs. 5–7, for simple problems Direct and LBDirect are better than the AG and for harder problems AG is better than its

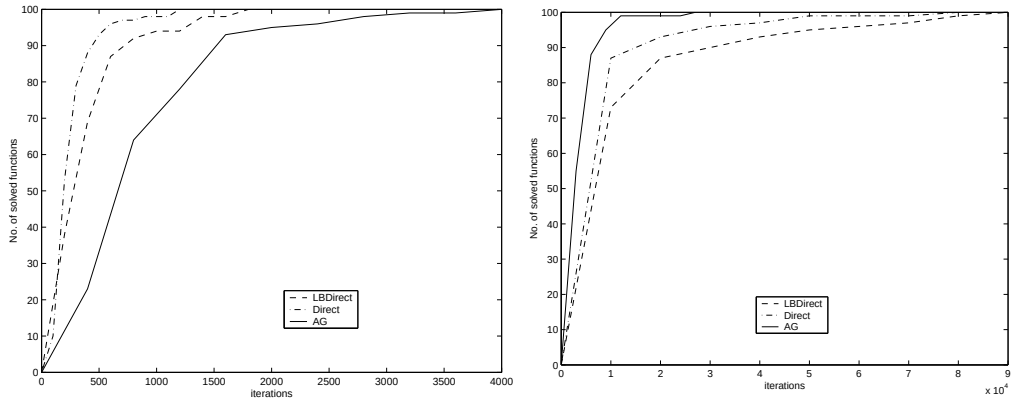

Figure 6: *Methods AG, Direct, and LBDirect, N=3. Class no.3, left; Class no.4, right*

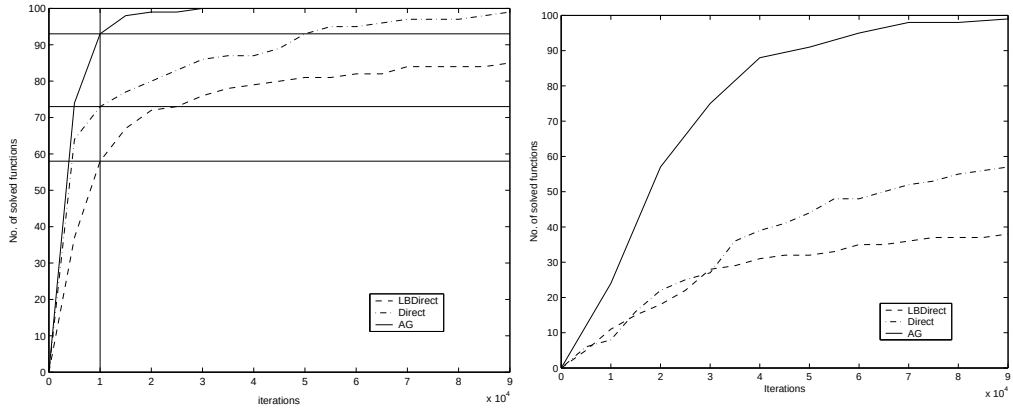

Figure 7: *Methods AG, Direct, and LBDirect, N=4. Class no.5, left; Class no.6, right*

competitors. The advantage of the AG becomes more pronounced both when classes of test functions become harder and when the dimension of problems increases. It can be noticed also that on the taken test classes the performance of the LBDirect is worse with respect to the Direct (note that these results are in a good agreement with experiments executed in [22, 23]). A possible reason of this behavior can be the following. Since the considered test functions have many local minima and due to its locally-biased character, LBDirect spends too much time exploring various local minimizers which are not global.

4.3 Comparison AG – AGI and AL – ALI

In the second series of experiments, the efficiency of the local improvement technique was studied. For this purpose, the algorithms AG and AL were compared with the algorithms AGI and ALI, respectively, on the class 1 from Table 1 (see Fig. 8). All experiments were performed with $\xi = 10^{-8}$, $\delta = 10^{-6}$, δ from (3.8), and $\varepsilon = 0$ using

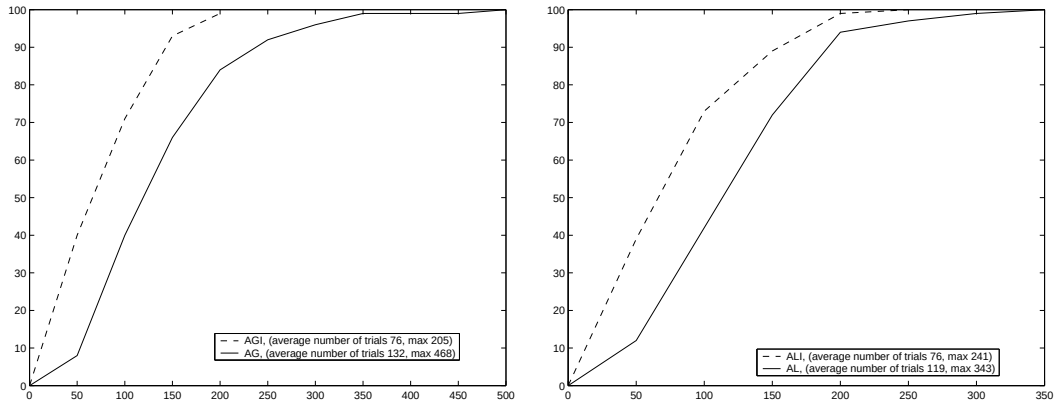

Figure 8: *Methods AGI and AG using the global estimate, left. Methods ALI and AL using local estimates, right.*

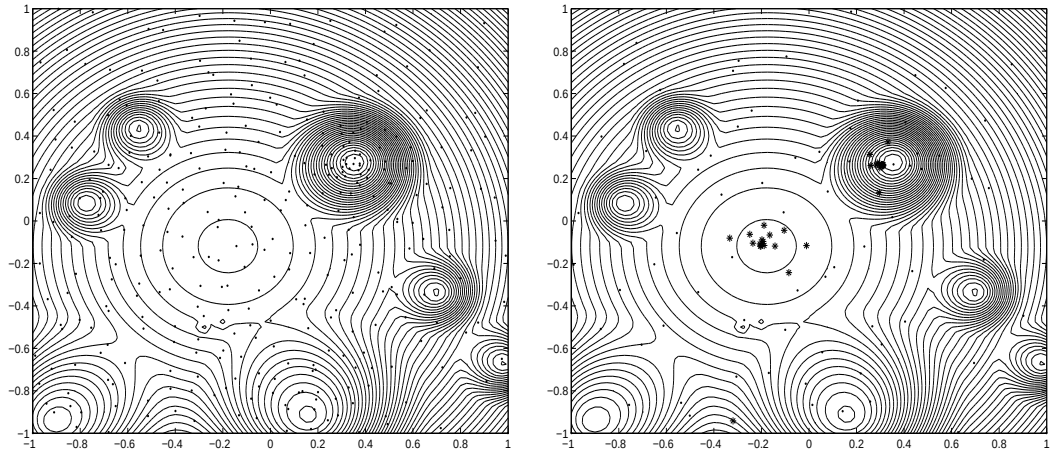

Figure 9: *Function no.55, class 1. Trial points produced by the AG, left. Trial points produced by the AGI, right. Trial points chosen by the local improvement strategy are shown by the symbol "*".*

the strategy a) with the radius $\rho = 0.01\sqrt{N}$, where ρ is from (4.3). The choices of the reliability parameter r are given in Appendix.

In order to illustrate a different behavior of the methods using the local improvement technique, Fig. 9 shows behavior of the AG and the AGI on problem no.55 from class 1. Fig. 9-left shows 337 points of trials executed by the AG to find the global minimum of the problem and Fig. 9-right presents 107 points of trials executed by the AGI to solve the same problem. Recall, that the search has been stopped using the rule a), i.e., as soon as a point within the ball B_{55} has been placed.

4.4 Comparison AGI – ALI

In the third series of experiments (see Fig. 10), the algorithms AGI and ALI are compared in order to study the influence of the local tuning technique in the situation

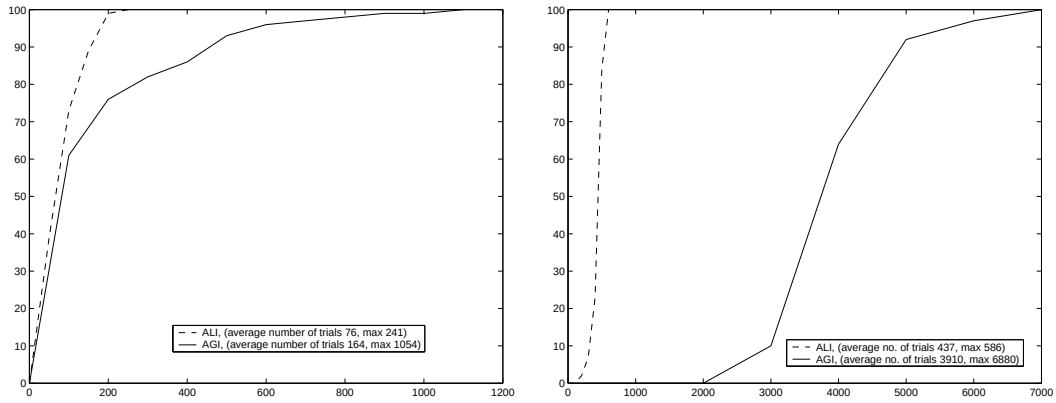

Figure 10: *ALI and AGI:* $\varepsilon = 0$*, left. ALI and AGI:* $\varepsilon = .001$*, right.*

when the local improvement is applied too. The choices of the reliability parameter r are given in Appendix and the other parameters have been chosen as in the second series of experiments. In Fig. 10-left, the rule a) is used. It can be noticed that the method ALI is faster in finding the global solution: the maximum number of iterations executed by ALI is 241 against 1054 carried out by the algorithm AGI. In Fig. 10-right, the strategy b) is used, the algorithms stop when the rule (2.7) is satisfied, with $\varepsilon = 0.001$. This criterion is very important because in solving real-life problems we do not know a priori the global solution of the problem. Thus, it is very important to study, how many trials should execute the methods to find the solution and to stop by using the practical criterion b). It can be seen that the ALI is very fast to stop, whereas the method AGI executes a global analysis of the whole domain of each objective function so that the stopping rule (2.7) is verified after a higher number of trials.

4.5 Comparison AG – ALI

In the fourth series of experiments we compare the basic algorithm AG with the algorithm ALI using both the local tuning and the local improvement, on classes 1, 3 and 5 from Table 1. The practical rule b) was used in these experiments. The choices of the reliability parameter r are given in Appendix.

In dimension 2, the values of δ, ξ, and ρ were the same as in the experiments above; ε was fixed equal to 0.001. In Fig. 11-left the behavior of the two methods can be seen. Note that after 500 iterations the stopping rule in the ALI was verified for 84 functions and all the minima have been found, whereas the algorithm AG stopped only at 2 functions.

For $N = 3$, the radius $\rho = 0.01\sqrt{N}$ has been used. The parameters of the methods have been chosen as follows: the search accuracy $\varepsilon = 0.0022$, $\delta = 10^{-6}$, and $\xi = 10^{-8}$. In Fig. 11-right the behavior of the methods can be seen. All global minima have been found.

In the last experiment of this series, the class of functions with $N = 4$ has been used. The methods AG and ALI worked with the following parameters: $\varepsilon = 0.005$,

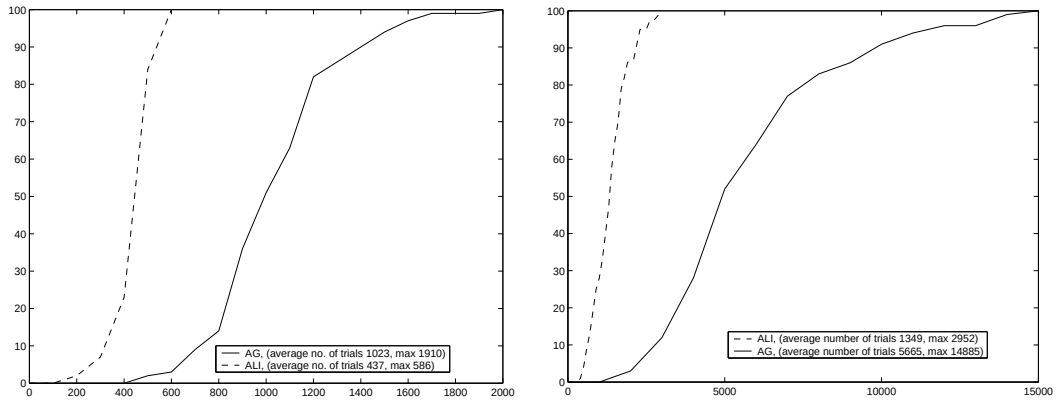

Figure 11: *N=2, class 1, methods AG and ALI, left. N=3, class 3, methods AG and ALI, right.*

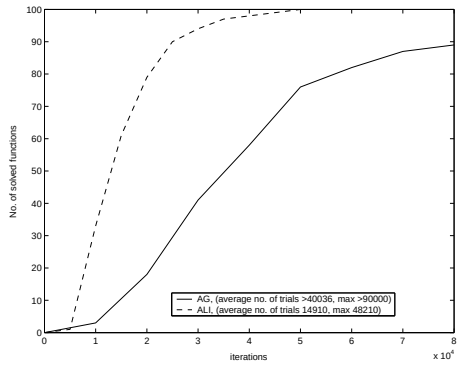

Figure 12: *N=4, class 5, methods AG and ALI*

$\rho = 0.04\sqrt{N}$, $\delta = 10^{-8}$, $\xi = 10^{-8}$. The algorithm AG was not able to stop within the maximal number of trials, 90000, for 11 functions, however, the a posteriori analysis has shown that the global minima have been found for these functions, too. Fig. 12 illustrates the results of the experiment.

5 Conclusion

In this paper, the multi-dimensional global optimization problem where the objective function satisfies the Lipschitz condition over a closed hyperinterval has been considered. Four algorithms have been proposed and their convergence conditions to the global optimum have been studied. All of them use Hilbert type space-filling curves to reduce the original problem to a Hölder one-dimensional problem. A theoretical relation between the order of a Hilbert space-filling curve approximation used for to reduce the problem dimension and the accuracy of the obtained solution of the reduced one-dimensional problem has been established. This result has allowed the authors to studied theoretical relations between solutions to the original multi-

dimensional problem and to the reduced one-dimensional problems approximating the original solution.

The algorithms proposed do not require the knowledge of the Lipschitz constant and use only the values of the objective function in their work. The first two methods proposed, AG and AGI, adaptively estimates the Hölder constant during the search. In the algorithms AL and ALI, the local tuning on the behavior of the functions is executed during the work of the global optimization procedure: the local Hölder constants over different subregions of the search domain are adaptively estimated. In the methods AGI and ALI, the newly introduced local improvement strategy has been used in order to accelerate the search.

The algorithms have been tested on six classes of 100 functions taken from literature. Numerical experiments show quite a satisfactory performance of the new algorithms. The slowest of them, the AG, has outperformed the popular method Direct and its recent locally biased modification LBDirect on the 600 test functions considered. The ALI, the algorithm using both the local tuning and the local improvement strategies, was very fast in comparison with the other three methods proposed in the paper. The methods AL and ALI demonstrated a high level of adaptation to the behavior of the objective function. A proper choice of the reliability parameter r can accelerate the search significantly. In contrast, the methods AG and AGI using the global estimates of the Hölder constant are less sensible to this parameter but are slower than the algorithms using the local tuning. Finally, the newly introduced local improvement technique forcing the global optimization methods to improve the best found approximation immediately after its discovering has proved its high efficiency.

Appendix

In this section we specify the values of the reliability parameter r used in all the experiments. Due to Theorem 3.3, every function optimized by the AG, AGI, AL and ALI algorithms has a crucial value r^* of this parameter. Therefore, when one executes tests with a class of 100 different functions it becomes difficult to use specific values of r for each function, hence in our experiments at most two or three values of this parameter have been fixed for the entire class. Clearly, such a choice does not allow the algorithms to show their complete potential. However, even under these unfavorable conditions, the four algorithms proposed in the paper have shown a nice performance. Note that the meaning of parameters of this kind in Lipschitz global optimization is discussed in detail, e.g., in [14, 23, 24, 27, 29].

The following values of the reliability parameter r was used for the AG method in the first series of experiments: for the class 1 the value $r = 1.1$ for 98 functions, and $r = 1.2$ for the remaining two functions. The value $r = 1.4$ was used for 97 functions of the class 2 and $r = 1.5$ for the remaining 3 functions of this class. In dimension $N = 3$ the value $r = 1.1$ was applied for all 100 functions of classes 3 and for 99 functions of class 4; the value $r = 1.2$ for 1 function of class 4. In dimension $N = 4$ the value $r = 1.1$ was used for all the functions of class 5 and for 98 functions

of class 6, and $r = 1.3$ for the remaining two function of class 6.

In the second series of experiment the following values of the parameter r have been used: in the methods AG and AGI the reliability parameter $r = 1.3$; in the ALI the value $r = 2.8$ was used for all 100 functions of the class and in the method AL the same value $r = 2.8$ was used for 98 functions and $r = 2.9$ for the remaining two functions.

In the third series of experiments the same value of the parameter $r = 2.8$ has been used in both methods (AGI and ALI).

In the fourth series of experiments the following values of the parameter r have been used: in dimension N=2, in the AG the value $r = 1.3$ and in the ALI the value $r = 2.8$. In dimension N=3 the value $r = 1.1$ has been applied in the method AG for all 100 functions of the class; in the method ALI, $r = 3.1$ has been used for 73 functions of the class, $r = 3.4$ for 20 functions, and $r = 3.9$ for the remaining 7 functions. In dimension N=4, $r = 1.1$ in the method AG; $r = 6.5$ in the ALI for 77 functions of the class, $r = 6.9$ for 17 functions, $r = 7.7$ for the remaining 6 functions.

References

[1] Addis B., Locatelli M., *A new class of test functions for global optimization*, Journal of Global Optimization, 38, 479–501 (2007).

[2] Bomze I.M., Csendes T., Horst R. and Pardalos P.M., *Developments in Global Optimization*, Kluwer Academic Publishers, Dordrecht (1997).

[3] Butz A.R., *Space filling curves and mathematical programming*, Information and Control, 12(4), 313–330 (1968).

[4] Dixon L.C.W. and Szegö G.P., Eds, *Towards Global Optimization*, Vol.2, North-Holland, Amsterdam, 1978.

[5] Gablonsky M.J., *An implemention of the Direct Algorithm*, Technical report CRSC-TR04-30, Center for Research in Scientific Computation, North Carolina State University, Raleigh, NC, 2004.

[6] Gablonsky M.J., *DIRECT v2.04 FORTRAN code with documentation*, 2001, http://www4.ncsu.edu/ ctk/SOFTWARE/DIRECTv204.tar.gz.

[7] Gablonsky M.J., *Modifications of the DIRECT Algorithm*, Ph.D thesis, North Carolina State University, Raleigh, NC, 2001.

[8] Gablonsky M.J. and Kelley C.T., *A locally-biased form of the DIRECT Algorithm*, Journal of Global Optimization, 21, 27–37 (2001).

[9] Gaviano M., Kvasov D.E., Lera D. and Sergeyev Ya.D., *Software for generation of classes of test functions with known local and global minima for global optimization*, ACM TOMS, 29(4), 469–480 (2003).

[10] Horst R. and Pardalos P.M., *Handbook of Global Optimization*, Kluwer Academic Publishers, Doordrecht (1995).

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

[12] Lera D. and Sergeyev Y.D., *Global minimization algorithms for Hölder functions*, BIT, 42(1), 119–133 (2002).

[13] Locatelli M., *On the multilevel structure of global optimization problems*, Computational Optimization and Applications, 30, 5–22 (2005).

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

[15] Piyavskii S.A., *An algorithm for finding the absolute extremum of a function*, USSR Computational Mathematics and Mathematical Physics, 12, 57–67 (1972).

[16] Platzman L.K. and Bartholdi J.J. III, *Spacefilling curves and the planar travelling salesman problem*, J. ACM, 36(4), 719–737 (1989).

[17] Ratschek H. and Rokne J., *New Computer Methods for Global Optimization*, Ellis Horwood, Chichester (1988).

[18] Sagan H., *Space-Filling Curves*, Springer–Verlag, New York (1994).

[19] Sergeyev Ya.D., *On convergence of "Divide the Best" global optimization algorithms*, Optimization, 44(3), 303–325 (1998).

[20] Sergeyev Ya.D., *Global one-dimensional optimization using smoth auxiliary functions*, Mathematical Programming, 81(1), 127–146 (1998).

[21] Sergeyev Ya.D., Daponte P., Grimaldi D. and Molinaro A., *Two methods for solving optimization problems arising in electronic measurements and electrical engineering*, SIAM J. Optim., 10(1), 1–21 (1999).

[22] Sergeyev Ya.D., Kvasov D.E. *Global search based on efficient diagonal partitions and a set of Lipschitz constants*, SIAM J. Optim. 16(3), 910–937 (2006).

[23] Sergeyev Ya.D., Kvasov D.E. *Diagonal global optimization methods*, Fizmatlit, Moscow (2008), (in Russian).

[24] Strongin R.G., *Numerical Methods in Multiextremal Problems: Information-Statistical Algorithms*, Nauka, Moscow (1978), (In Russian).

[25] Strongin R.G., *The information approach to multiextremal optimization problems*, Stochastics and Stochastics Reports 27, 65–82 (1989).

[26] Strongin R.G., *Algorithms for multi-extremal mathematical programming problems employing the set of joint space-filling curves*, Journal of Global Optimization 2, 357–378 (1992).

[27] Strongin R.G. and Sergeyev Ya.D., *Global Optimization with Non-convex Constraints: Sequential and Parallel Algorithms*, Kluwer Academic Publishers, Dordrecht (2000).

[28] Törn A., Ali M.M. and Viitanen S., *Stochastic global optimization: Problem classes and solution techniques*, Journal of Global Optimization 14, 437–447 (1999).

[29] Zhigljavsky A.A. and Žilinskas A., *Stochastic Global Optimization*, Springer–Verlag, New York, 2008.