

1 **GOSH: derivative-free global optimization using**
2 **multi-dimensional space-filling curves**

3 Daniela Lera* and Yaroslav D. Sergeyev†

 This version of the article has been accepted
 for publication, after peer review
 and is subject to Springer Nature’s AM terms of use,
 but is not the Version of Record and does not
 reflect post-acceptance improvements, or any corrections.
 The Version of Record is available online at:
 <https://doi.org/10.1007/s10898-017-0589-7>

4 **Abstract**

5 Global optimization is a field of mathematical programming dealing with
6 finding global (absolute) minima of multi-dimensional multiextremal func-
7 tions. Problems of this kind where the objective function is non-differentiable,
8 satisfies the Lipschitz condition with an unknown Lipschitz constant, and is
9 given as a “black-box” are very often encountered in engineering optimization
10 applications. Due to the presence of multiple local minima and the absence
11 of differentiability, traditional optimization techniques using gradients and
12 working with problems having only one minimum cannot be applied in this
13 case. These real-life applied problems are attacked here by employing one of
14 the mostly abstract mathematical objects – space-filling curves. A practical
15 derivative-free deterministic method reducing the dimensionality of the prob-
16 lem by using space-filling curves and working simultaneously with all possible
17 estimates of Lipschitz and Hölder constants is proposed. A smart adaptive
18 balancing of local and global information collected during the search is per-
19 formed at each iteration. Conditions ensuring convergence of the new method
20 to the global minima are established. Results of numerical experiments on
21 1000 randomly generated test functions show a clear superiority of the new
22 method w.r.t. the popular method DIRECT and other competitors.

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

†Corresponding author; Dipartimento di Ingegneria Informatica, Modellistica, Elettronica 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 Department of Software and Supercomputing, Lobachevskiy University of Nizhni Novgorod, Gagarin Av. 23, Nizhni Novgorod, Russia

23 **Key Words.** Global optimization, space-filling curves, derivative-free meth-
24 ods, acceleration, Lipschitz functions.

25 1 Introduction

26 Many real-world optimization problems are stated as a global optimization problem
27 since functions describing these applications are often multiextremal, non-differen-
28 tiable, and hard to evaluate even at one point (see, for example [17, 21, 22, 31, 34,
29 35, 47, 52]). In this paper, we focus our attention on continuous global optimization
30 problems

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

31 where S is a hyperinterval in \mathbf{R}^N and the objective function $F(y)$ can be multiex-
32 tremal, non-differentiable, and given as a “black-box”, i.e., any information regard-
33 ing its analytical representation or any other data describing its structure is not
34 available. However, it is supposed that $F(y)$ satisfies the Lipschitz condition

$$|F(y') - F(y'')| \leq L\|y' - y''\|, \quad y', y'' \in S, \quad (1.2)$$

35 with an unknown Lipschitz constant L , $0 < L < \infty$, in the Euclidean norm. This
36 statement can be very often encountered in practice and in the literature there exist
37 numerous methods for dealing with the problem (1.1), (1.2) (see, e.g., [1, 3, 4, 5, 13,
38 17, 21, 23, 32, 33, 34, 41, 47, 48, 51, 52]).

39 In this paper, we consider the applied problem (1.1), (1.2) by using one of the
40 mostly abstract mathematical objects – space-filling curves introduced by Peano in
41 1890 and independently by Hilbert in 1891 (even though we use Hilbert’s version of
42 the curves, the traditional terminology for this kind of objects is “Peano curves” due
43 to the priority of Peano). The curves under consideration emerge as the limit objects
44 generated by an iterative process and are fractals constructed using the principle of
45 self-similarity. It is possible to prove that the curves fill in the hypercube $S \subset \mathbf{R}^N$,
46 i.e., they pass through every point of S (this fact gave rise to the term “space-
47 filling curves”). It is known that it is possible to reduce the dimension of the global
48 optimization problem (1.1), (1.2) and to move from a multivariate problem to a
49 univariate one (see studies in this direction in [2, 38, 44, 45, 46, 47]).

50 More precisely, it can be shown (see [2, 45, 47]) that, by using space-filling curves,
51 the multi-dimensional global minimization problem (1.1), (1.2) can be turned into
52 a one-dimensional problem and that finding the global minimum of the Lipschitz
53 function $F(y)$, $y \in S \subset R^N$, is equivalent to determining the global minimum of the
54 one-dimensional function $f(x)$ over the interval $[0, 1]$, i.e., it follows

$$f(x) = F(p(x)), \quad x \in [0, 1], \quad (1.3)$$

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

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

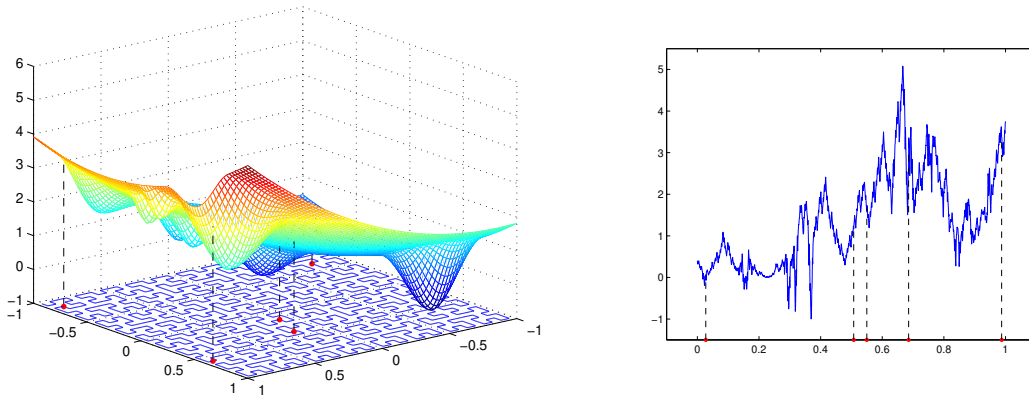


Figure 1: A two-dimensional function from [10] satisfying the Lipschitz condition together with an approximation of level 5 to Peano curve (left) and the corresponding univariate Hölderian function (right). Red dots show points on the curve where the objective function has been evaluated.

56 holds (see [47]) for the function $f(x)$ with the constant

$$H = 2L\sqrt{N + 3}, \quad (1.5)$$

57 where L is the Lipschitz constant of the original multi-dimensional function $F(y)$
 58 from (1.1), (1.2). In Fig. 1-right, the reduced function in one dimension correspond-
 59 ing to the test function in two dimensions from Fig. 1-left is shown. Clearly, a
 60 numerical approximation of the Peano curve is used in computations for the reduc-
 61 tion. Thus, one can try to attack the problem (1.1), (1.2) by proposing algorithms
 62 for minimizing Hölderian function (1.3), (1.4) in one dimension.

63 It can be seen from the statement of the original problem (1.1), (1.2) that the only
 64 available information regarding the multi-dimensional function $F(y)$ is that $F(y)$
 65 satisfies the Lipschitz condition (1.2) with an unknown constant L . As a result,
 66 the way the Lipschitz information is used by an optimization algorithm becomes
 67 crucial for its performance, convergence, and speed. In the literature there exist
 68 several methods to estimate L (see [4, 11, 12, 15, 16, 17, 18, 42, 43, 44, 47, 50]),
 69 and it is known that an overestimation of L may slow down the search while an
 70 underestimate of the constant can lead to loss of the global solution.

71 First, there exist algorithms that for the whole domain S use the same a priori
 72 given estimate of L or its adaptive estimate recalculated during the search at each
 73 iteration (see, e.g., [4, 17, 18, 33, 34, 36, 43, 44, 47]). This approach does not take into
 74 account any local information about the behavior of the objective function over small
 75 subregions of the domain S . This drawback can slow down the search significantly.
 76 A more advanced approach proposed originally in [39, 40] suggests to adaptively
 77 approximate local Lipschitz constants $\tilde{L}(D_j)$ in different subregions $D_j \subset S$ of
 78 the search region S during the process of optimization. This procedure performs
 79 a local tuning on the behavior of the objective function balancing global and local

80 information obtained during the search (see also interesting hybridization ideas in
81 [49, 50]). It has been shown in [20, 24, 39, 44, 47] that the local tuning techniques can
82 lead to a significant acceleration of the global search. Another interesting approach
83 that has been introduced in [19] in the popular method called DIRECT uses at each
84 iteration several estimates of the Lipschitz constant L simultaneously. This way
85 to deal with Lipschitz information attracts a wide interest of researchers (see, e.g.,
86 [6, 7, 8, 9, 19, 23, 29, 30, 31, 32, 33]) and is under scrutiny in this work, as well.

87 In this paper, we propose to use Peano curves and instead of using the Lipschitz
88 information in many dimensions to work with the Hölder information in one dimen-
89 sion trying to obtain several estimates of the Hölder constant using the DIRECT
90 methodology. It should be stressed that such a transposition of the approach is not
91 trivial at all. In fact, in the literature (see [14, 24, 25, 27, 28, 44]) there exist several
92 methods estimating global and local Hölder constants whereas the usage of the DI-
93 RECT approach encounters a number of serious difficulties (see [26]) in the context
94 of Hölder optimization. In Section 2, we describe a strategy that solves them and
95 allows us to work with several estimates of the Hölder constant at each iteration.
96 Then, a two-phases procedure intended to accelerate the search is presented in Sec-
97 tion 3. A new algorithm using both discoveries for solving the problem (1.1), (1.2)
98 and its convergence properties are described in Section 4. Section 5 presents results
99 of numerical experiments that compare the new method with its competitors on
100 1000 test functions randomly generated by the GKLS-generator from [10]. Finally,
101 Section 6 contains a brief conclusion.

102 2 Two ways to represent Hölderian minorants

103 Due to the use of the Peano space-filling curves, the N -dimensional problem (1.1),
104 (1.2) is turned into the one-dimensional problem (1.3), (1.4) with the one-dimensional
105 objective function $f(x)$ from (1.3) satisfying the Hölder condition (1.4) with a con-
106 stant $0 < H < \infty$ over the interval $[0, 1]$. It follows from (1.4) that, for all $x, z \in [0, 1]$
107 we have

$$f(x) \geq f(z) - H|x - z|^{1/N}. \quad (2.1)$$

This fact means that the function

$$G(x) = f(z) - H|x - z|^{1/N},$$

108 with $z \in [0, 1]$ fixed, is a minorant (or support function) for $f(x)$ over $[0, 1]$, i.e.

$$f(x) \geq G(x), \quad x \in [0, 1].$$

109 Analogously, if we consider subintervals $d_i = [a_i, b_i]$, $1 \leq i \leq k$, belonging to $[0, 1]$
110 we obtain that the following function

$$G^k(x) = g_i(x), \quad x \in [a_i, b_i], \quad 1 \leq i \leq k, \quad (2.2)$$

$$111 \quad g_i(x) = \begin{cases} g_i^-(x) = f(m_i) - H(m_i - x)^{1/N}, & x \in [a_i, m_i], \\ g_i^+(x) = f(m_i) - H(x - m_i)^{1/N}, & x \in [m_i, b_i], \end{cases} \quad (2.3)$$

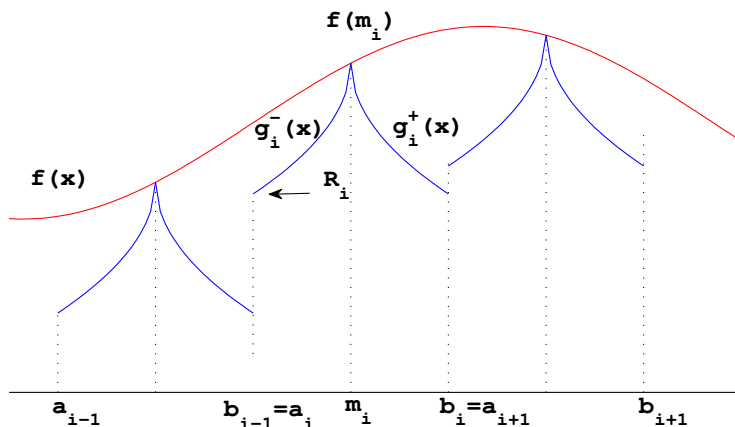


Figure 2: Hölder support functions.

112

$$m_i = (a_i + b_i)/2 \quad (2.4)$$

113 is a discontinuous nonlinear minorant for $f(x)$ (see Fig. 2) and the values R_i , $1 \leq$
 114 $i \leq k$, are lower bounds for the function $f(x)$ over each interval d_i , $1 \leq i \leq k$. These
 115 values are called *characteristics of intervals* and can be calculated as follows if an
 116 overestimate $H_1 \geq H$ of the Hölder constant H is given

$$R_i = R_i(H_1) = \min_{x \in [a_i, b_i]} g_i(x) = f(m_i) - H_1 |(b_i - a_i)/2|^{1/N}. \quad (2.5)$$

117 As was mentioned in the introduction, the DIRECT algorithm (see [19]) uses at
 118 each iteration several estimates of the Lipschitz constant for selecting a suitable set
 119 of subintervals in the central points of which to evaluate the objective function. This
 120 selection can be easily done thanks to a smart representation of the intervals in a
 121 diagram in two dimensions. This representation is the core point of DIRECT and can
 122 be done since the Lipschitz information is used by this method producing so piece-
 123 wise linear minorants. In order to use the same methodology in the framework of
 124 the Hölderian optimization it is necessary to be able to find a suitable representation
 125 of intervals, as well.

126 Let us try to do this following the idea of DIRECT and show that a simple
 127 transposition from Lipschitz to Hölder world does not work. We represent in a
 128 two-dimensional diagram each interval $d_i = [a_i, b_i]$ by a point with coordinates
 129 $(h_i, f(m_i))$, where $h_i = 0.5(b_i - a_i)$ and m_i is from (2.4) exactly as DIRECT does. In

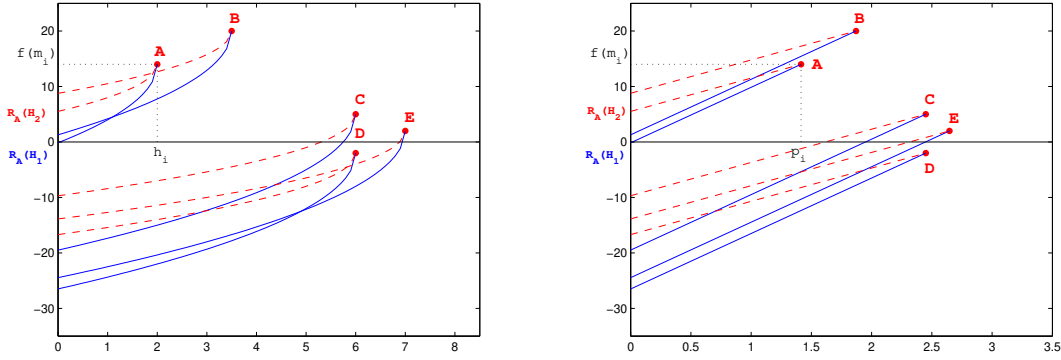


Figure 3: Representation of intervals in the Euclidean metric (left) and in the Hölderian metric (right).

130 Fig. 3-left, we have represented five different intervals d_A , d_B , d_C , d_D , and d_E by the
 131 points A , B , C , D , and E , respectively. If we consider a fixed overestimate H_1 of
 132 the Hölder constant, we can observe the corresponding nonlinear support functions
 133 (2.3) (shown in blue solid lines) related to these intervals. The characteristic $R_A(H_1)$
 134 of the interval represented by the dot A is obtained as the intersection of the curve
 135 (2.3) constructed at the point A with the vertical coordinate axis. It can be seen
 136 that the best (the lowest) characteristic is $R_D(H_1)$ and the interval d_D would be
 137 subdivided at the next iteration if H_1 is chosen as the estimate for H . However,
 138 the choice of $R_D(H_1)$ is not easy since, as it can be seen from Fig. 3-left, the curves
 139 constructed using the estimate H_1 intersect one another in various ways.

140 In addition, remind that we do not know the real value of H and wish to try all
 141 possible estimates of H from zero to infinity. The auxiliary functions corresponding
 142 to the second estimate H_2 are shown in Fig. 3-left by red dashed lines. They produce
 143 again a lot of intersections among themselves and with the curves corresponding
 144 to H_1 . It becomes clear that the selection of the lowest characteristic for all possible
 145 estimates of H even with such a small number of intervals becomes complicated
 146 and it is unclear how to select intervals by varying estimates of the Hölder constant
 147 from 0 to infinity.

148 In order to overcome this difficulty and to give a more transparent procedure
 149 for selection of the best characteristics, a different representation of the intervals
 150 is proposed. The idea consists of the usage of the metric of Hölder instead of the
 151 Euclidean one in the construction of the diagram. More precisely, a generic interval
 152 $d_i = [a_i, b_i]$ belonging to a current partition $\{D^k\}$ at the k th iteration is represented
 153 by a dot P_i with the coordinates (p_i, w_i) where

$$p_i = |(b_i - a_i)/2|^{1/N}, \quad w_i = f(m_i), \quad (2.6)$$

154 and m_i is from (2.4).

155 In Fig. 3-right, the representation of the same five intervals considered in Fig. 3-
 156 left can be observed in the new metric. A great simplification can be clearly seen

157 since there are no more nonlinear curves and intersections between them for each
 158 fixed estimate of H . The obtained diagram is very similar to that used by the
 159 DIRECT method, in the Lipschitzian case [19]. In Fig. 3-right, the characteristic
 160 $R_A(H_1)$ of the interval represented by the point A is exactly the intersection of
 161 the line passing through A with slope H_1 and the vertical coordinate axis. Notice
 162 that, as expected, the values in the vertical coordinate axis coincide with those of
 163 Fig. 3-left. The selection of intervals with the best characteristics corresponding to
 164 different estimates of H becomes so much easier and is discussed in the following
 165 two sections.

166 3 Selection of intervals: Two-phase approach

167 In this section, we describe in detail the intervals selection procedure that will be
 168 used in the method to be introduced in Section 4. As was already said above, at each
 169 iteration k the method should select in a suitable way a promising set of subintervals
 170 in which it intends to intensify the search and execute new trials (*trial* is evaluation
 171 of $f(x)$ at a point x that is called *trial point*). In order to accelerate the search, a
 172 two-phase technique that balances the global and local information collected during
 173 the work of the method is introduced.

174 In order to describe the selection procedure let us discuss Fig. 4 that shows
 175 a possible scenario at a generic iteration k of the algorithm. The interval $[0, 1]$
 176 (remind that since Peano curves are applied, the search is performed over the one-
 177 dimensional interval $[0, 1]$ (see (1.3))) is subdivided into subintervals $d_i = [a_i, b_i]$,
 178 $i = 1, \dots, I(k)$, belonging to the current partition D^k . Each interval is represented
 179 by a point in the two-dimensional diagram in Fig. 4, with coordinates given by (2.6),
 180 and is characterized, for each fixed value of H , by a lower bound given by R_i from
 181 (2.5). Points with the same abscissa represent intervals that have the same width.
 182 In Fig. 4, there are nine different groups of intervals corresponding to the points A ,
 183 B , ..., I . At each iteration $k \geq 1$ of the method each group of intervals receives a
 184 positive integer index $l = l(k)$. The first group of large intervals (the column of the
 185 dot A in Fig. 4) gets the index $l = 1$, and the subsequent intervals are identified
 186 progressively by indices 2, 3, 4, ... etc. So, in Fig. 4 there are nine groups with
 187 indices 1, 2, ..., 9. The index 9 is referred to the group of intervals with minimal
 188 width (column of the point I).

189 For any fixed value H of the Hölder constant, it is easy (see Fig. 3-right where
 190 lower bounds for $H = H_1$ and $H = H_2$ are shown) to identify the interval cor-
 191 responding to the minimal lower bound with respect to the other intervals in the
 192 current partition. By varying the value of H from 0 to infinity, the method should
 193 select a set of intervals corresponding to the smallest lower bound from (2.5) for
 194 some particular estimate of the Hölder constant H . These intervals should be par-
 195 titioned during the next iteration and are called *nondominated* intervals and it can
 196 be easily seen that they are located on the lower-convex hull of the set of dots repre-
 197 senting the intervals. In Fig. 4 the nondominated intervals are identified by points
 198 located at the bottom of each group with the same horizontal coordinate, that is

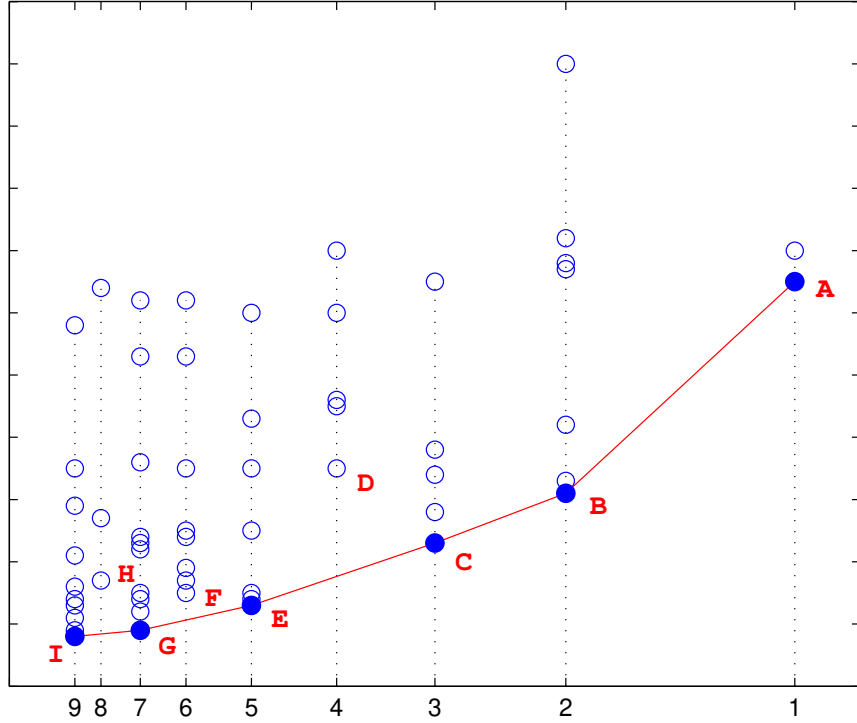


Figure 4: The nondominated intervals d_A , d_B , d_C , d_E , d_G and d_I are represented by dots A , B , C , E , G and I .

199 points A , B , C , E , G and I . In practice, to determine these intervals algorithms
 200 for identifying the convex hull of the dots can be used, for example, the algorithm
 201 called Jarvis march, or gift wrapping, see [37]. Notice that the points H , F , and D
 202 do not represent nondominated intervals even though they are the lowest in their
 203 groups. This happens because (see, e.g., the point F) the point G dominates F at
 204 smaller values of Hölder constant H and the point E dominates F at higher values
 205 of H .

206 The two phases (that can interchange each other several times during the work
 207 of the method) are the following: investigation of large unexplored intervals in order
 208 to find attraction regions of local minimizers that are better than the current best
 209 found solution (global phase) and a local improvement of the current best found
 210 solution (local phase). In order to explain their functioning let us remind that all
 211 the intervals on the diagram (see Fig. 4) are ordered in the increasing order from
 212 smaller to larger intervals along the horizontal axis. Thus, well explored zones of the
 213 search region corresponding to attraction regions of already visited local minima are
 214 located on the left-hand part of the diagram (small intervals) whereas unexplored
 215 zones of the domain are represented on the right-hand part of the diagram (large

216 intervals). If during the work of the global phase a better solution than the current
 217 one has been obtained, then the method switches to the local phase in order to
 218 improve the new best record. After several improving steps the method switches
 219 back to the global phase and the search of new promising minima continues until
 220 the satisfaction of a stopping rule.

221 During the global phase the new algorithm explores mainly large intervals, thus
 222 it identifies the set of nondominated intervals not among all groups of intervals but
 223 only among some groups with indices lower than a calculated “middle index” r . This
 224 index represents a separator between the groups of large intervals and small ones.
 225 The global phase is performed until a function value improving the current minimal
 226 value on at least 1% is obtained. When this happens, the method switches to the
 227 local phase in the course of which the obtained new solution is improved locally. In
 228 the case when the algorithm is not switched to the local phase during more than a
 229 fixed number $IglobMax$ of iterations (the improvement of the current minimum is
 230 still not found by exploring large intervals), it performs one “security” iteration in
 231 which determines nondominated intervals considering all groups of intervals present
 232 in the diagram.

233 Thus, during each iteration in the global phase the algorithm identifies a set
 234 of nondominated intervals. The subdivision of each of these intervals is performed
 235 only if a significant improvement on the function values with respect to the cur-
 236 rent minimal value $f_{min}(k)$ is expected, i.e., once an interval $d_t \in \{D^k\}$ becomes
 237 nondominated, it can be subdivided only if the following condition is satisfied

$$R_t(\tilde{H}) \leq f_{min}(k) - \xi, \quad (3.1)$$

238 where the lower bound $R_t = R_t(\tilde{H})$ is from (2.5) and the parameter ξ prevents the
 239 algorithm from subdividing already well-explored small subintervals.

240 During the local phase improving the just found new best solution the algorithm
 241 always explores three intervals: the interval containing the best current point (best
 242 interval) and the intervals located on the right and on the left of it. This phase
 243 finishes when the width of at least one of these intervals is less than a given accuracy.
 244 After the end of the local phase the algorithm switches back to the global phase and
 245 tries to find better solutions that can be located far away from the current best point.
 246 Notice that during the local phase a security iteration is carried out after performing
 247 a fixed number $IlocMax$ of iterations without switching to the global phase. This
 248 is done in order to avoid a too long concentrating of efforts at local minima that
 249 are not global solutions. As before, at the security iteration nondominated intervals
 250 among all groups of intervals present in the diagram are taken into consideration.

251 Once the selection phase (local or global) has been concluded, the chosen inter-
 252 vals are subdivided in order to produce new trial points by the following partition
 253 strategy. At a generic iteration k , let S_k be the set of the intervals to be partitioned
 254 and $d_t = [a_t, b_t]$ be an element of S_k represented by the corresponding point in the
 255 diagram at Fig. 4. Each interval d_t of the set S_k is subdivided into three equal parts

$$[a_t, b_t] = [a_t, u_t] \cup [u_t, v_t] \cup [v_t, b_t], \quad (3.2)$$

256 of the length $(b_t - a_t)/3$, with

$$u_t = a_t + (b_t - a_t)/3, \quad v_t = b_t - (b_t - a_t)/3. \quad (3.3)$$

257 The three new generated intervals are added to the current partition $\{D^k\}$ and
 258 to the diagram in Fig. 4 and the interval $[a_t, b_t]$ is deleted from both. Finally, two
 259 new trials, $f(c_1)$ and $f(c_2)$, are executed at the central points of the new intervals
 260 $[a_t, u_t]$ and $[v_t, b_t]$, where

$$c_1 = (a_t + u_t)/2, \quad c_2 = (v_t + b_t)/2. \quad (3.4)$$

261 Notice that the midpoint of the third interval $[u_t, v_t]$ is also the midpoint of the
 262 initial interval $[a_t, b_t]$ and, therefore, the function $f(x)$ has already been calculated
 263 in it at previous iterations.

264 We conclude this section by reminding that the objective function $f(x)$ is ob-
 265 tained by applying Peano curve that theoretically is introduced as a limit object
 266 being a fractal constructed using principles of the self-similarity. In practice, com-
 267 putable approximations of the Peano curve are used. Let us denote them by
 268 $p_M(x)$, where M is the level of approximation of the curve (see approximations
 269 with $M = 3, 4$, and 5 in Figs. ??–1, respectively). The choice of the level M of
 270 the curve is essential to obtain a good performance of the method: in fact, a level
 271 that is too low can be insufficient to fill in the domain in an appropriate way cre-
 272 ating so a risk to lose the optimal solution. On the other hand, when the value
 273 of M increases, the function in one dimension becomes more oscillating, especially
 274 if the dimension N of the original problem (1.1) grows up (see [28] for a detailed
 275 discussion). With increasing the dimension N , the width of intervals selected for
 276 partitioning can become very small (remind that we are in $[0, 1]$ and the metric of
 277 Hölder is used) and even get close to the computer precision. For these reasons it
 278 is required an additional check of the width of the interval before subdivision. Namely,
 279 the interval $d_t = [a_t, b_t]$ is partitioned only if the following condition is satisfied

$$b_t - a_t > \delta, \quad (3.5)$$

280 where δ is a parameter of the method.

281 4 The GOSH algorithm

282 In this section, a new algorithm called *GOSH* (*Global Optimization algorithm work-*
 283 *ing with a Set of estimates of the Hölder constant*) is presented.

284 To describe the algorithm formally, we need to specify some notations. Suppose
 285 that at an iteration $k \geq 1$ a partition $\{D^k\}$ of $D = [0, 1]$ has been obtained. Suppose
 286 also that each interval $d_i \in \{D^k\}$ is represented by a dot in the two-dimensional
 287 diagram from Fig. 4 and each group of intervals with the same width is numbered by
 288 the same integer index: this index is an integer positive number that varies between
 289 $imax(k)$ (index that identifies the column of the larger intervals) and $imin(k)$ (index
 290 of the column of the smaller intervals). The following notations are also adopted:

291 $f_{min}(k)$ is the best function value (the “record” value) at the iteration k , and
292 $x_{min}(k)$ is the corresponding coordinate.

293 $d_{min}(k)$ is the interval containing the point $x_{min}(k)$.

294 $f_{prec}(k)$ is the old best record. It serves to memorize the record $f_{min}(k)$ at the start
295 of the current phase (local or global).

296 $Lcount$ and $Gcount$ are counters of iterations performed during the local and global
297 phases, respectively.

298 $IlocMax$ and $IglobMax$ are maximal allowed numbers of iterations that can be
299 executed during the local and global phases, respectively, before making the
300 general security iteration (in which the nondominated intervals are selected
301 from the entire search domain).

302 $phase$ is a flag specifying the current phase. It is equal to “loc” and “glob” in the
303 local and global phases, respectively.

304 $p_M(x)$ is the M -approximation of the Peano curve.

305 S^k is the set of intervals, $S^k \subset D^k$, that will be subdivided and the corresponding
306 set J^k is the set of their indices.

307 $jloc$ is a flag that takes into account the fact that the set S^k can be empty. In this
308 case $jloc = 0$, otherwise $jloc = 1$.

309 We are ready now to describe the algorithm.

310 **Algorithm GOSH**

311 **Step 0.** (Initialization). Set the current iteration number $k := 1$.

312 Split the initial interval $D = [0, 1]$ in three equal parts and set $x^1 = 1/6$, $x^2 =$
313 $1/2$, $x^3 = 5/6$ and compute the values of the function $z^j = f(x^j) = F(p_M(x^j))$,
314 $j = 1, 2, 3$.

315 Set the current partition of the search interval $D^1 = \{[0, 1/3], [1/3, 2/3], [2/3, 1]\}$.

316 Set the current number of intervals $I = 3$ and the current number of trials
317 $T = 3$.

318 Set $f_{min}(1) = \min\{z^1, z^2, z^3\}$, and $x_{min}(1) = \arg \min\{f(x^i) : i = 1, 2, 3\}$.

319 Set $phase = loc$, $Lcount = Gcount = 0$.

320 After executing k iterations, the iteration $k + 1$ consists of the following steps.

321 **Step 1.** (Intervals selection) Identify the set S^k , $S^k \subset D^k$, and the corresponding
322 set J^k as follows.

324 **Step 1.1** (Global phase) **if** ($phase == glob$) **then**
325 **if** ($Gcount < IglobMax$)
326 Determine nondominated intervals that satisfy conditions (3.1)
327 and (3.5) by considering only groups of intervals with indices
328 going from $imax(k)$ up to $r(k) = \lfloor (p(k) + imax(k))/2 \rfloor$,
329 where $\lfloor x \rfloor$ denotes the integer part of x and $p(k)$ is the index
330 of the group the interval $d_{min}(k)$ belongs to.
331 $Gcount = Gcount + 1$
332 **elseif** ($Gcount == IglobMax$)
333 Determine nondominated intervals that satisfy conditions (3.1)
334 and (3.5) by considering all the groups of intervals with indices
335 between $imax(k)$ and $p(k)$
336 $Gcount = 0$
337 **endif**

338 **Step 1.2** (Local phase) **if** ($phase == loc$) **then**
339 $jloc = 1$
340 **if** ($Lcount < IlocMax$)
341 Determine the interval $d_{min}(k)$ and the two intervals, denoted by
342 $dr_{min}(k)$ and $dl_{min}(k)$ located on the right and on the left of it,
343 respectively. They are selected only if the condition (3.5) is
344 satisfied.
345 $Lcount = Lcount + 1$
346 **elseif** ($Lcount == IlocMax$)
347 Determine nondominated intervals that satisfy conditions (3.1)
348 and (3.5) by considering all the groups of intervals with indices
349 between $imax(k)$ and $p(k)$.
350 $Lcount = 0$
351 **endif**
352 **endif**

353 Include found intervals in the set S^k and their indices in the set J^k .
354 If $S^k = \emptyset$ then $jloc = 0$ and go to Step 3.

355 **Step 2.** (Subdivision of intervals) Set $D^{k+1} = D^k$ and perform Steps 2.1–2.3.

Step 2.1 (Interval selection). Select a new interval $d_t = [a_t, b_t]$ from S^k such that

$$t = \arg \max_{j \in J^k} \{b_j - a_j\}.$$

357 **Step 2.2** (Subdivision and sampling). Subdivide interval d_t in three new
358 equal subintervals, named d_{t1} , d_{t2} , d_{t3} of the length $(b_t - a_t)/3$ following
359 (3.2), (3.3) and produce two new trial points accordingly to (3.4).
Eliminate the interval d_t from D^{k+1} , i.e., set $D^{k+1} = D^{k+1} \setminus \{d_t\}$, and update
 D^{k+1} with the insertion of the three new intervals, i.e.,

$$D^{k+1} = D^{k+1} \cup \{d_{t1}\} \cup \{d_{t2}\} \cup \{d_{t3}\}.$$

360 Increase both the current number of intervals $I = I + 2$, and the current
361 number of trials $T = T + 2$.
362 Update the current record f_{min} and the current record point x_{min} , if
363 necessary.
364 Set $amp(j) = (b_t - a_t)/3$, $j \in J^k$.

365 **Step 2.3** (Next interval). Eliminate the interval d_t from S^k , i.e., set
366 $S^k = S^k \setminus \{d_t\}$ and $J^k = J^k \setminus \{t\}$.
367 If $S^k \neq \emptyset$, then go to Step 2.1. Otherwise calculate $amploc = \min_{j \in J^k} amp(j)$
368 and go to Step 3.

369 **Step 3.** (Switch)
370 **if** ($f_{min}(k) \leq f_{prec}(k) - 0.01 \cdot |f_{prec}(k)|$)
371 $f_{prec}(k) = f_{min}(k)$
372 **if** ($phase == glob$) then $Lcount = 0$ **endif**
373 $phase = loc$
374 **elseif** ($phase == loc$.&. $amploc \geq \delta'$.&. $jloc == 1$)
375 $phase = loc$
376 **else**
377 **if** ($phase == loc$) then $Gcount = 0$ **endif**
378 $phase = glob$
379 **endif**

380 **Step 4.** (End of the current iteration). Increase the iteration counter $k = k + 1$.
381 Go to Step 1 and start the next iteration.

382 Different stopping criteria can be used in the *GOSH* algorithm introduced above.
383 One of them will be introduced in the next section presenting numerical experiments.

384 Let us make some comments upon the introduced method. Step 1 is the phase
385 of selection of the intervals that, as was said before, can be either global or local.
386 Suppose that at a generic iteration k of the algorithm the situation is that shown
387 in Fig. 4, with 9 different groups of intervals, and assume that the interval $d_{min}(k)$
388 containing the current minimum point $x_{min}(k)$, belongs to the group of intervals
389 identified by the index 7 (so exactly the point G). If $phase = loc$ then 3 intervals
390 will be selected: $d_{min}(k)$, that corresponds to the point G in the diagram Fig. 4
391 and the intervals located to the right and to the left of it in $[0, 1]$, respectively.
392 Notice, that the latter two intervals, namely $dr_{min}(k)$ and $dl_{min}(k)$, can belong to
393 two different groups of intervals in the diagram and not necessarily to the group
394 with the index 7. In contrast, if the situation where $phase = glob$ takes place then
395 the separator index r is calculated where $r = \lfloor \frac{7+1}{2} \rfloor = 4$ and the nondominated
396 intervals are searched only among the groups of intervals from index 1 to index 4.
397 In this example, intervals represented by the points A , B , and C at the diagram in
398 Fig. 4 will be selected and split in three parts. Dots A , B , and C will disappear
399 from the diagram and there will be three new points in the column of B , three in
400 the column of C , and three in that of D .

401 If in the local phase it happens that $Lcount = IlocMax$ (or, analogously, in the
402 global phase $Gcount = IglobMax$) then nondominated intervals among all groups
403 of intervals are retrieved. Thus, in the diagram at Fig. 4 intervals represented by
404 points $A, B, C, E, G,$ and I will be split. The three intervals obtained by the
405 interval d_I will be represented by three points in the newly created column with the
406 index 10. Notice that only intervals that satisfy condition (3.5) are selected for the
407 further subdivision. It should be also emphasized that in Step 3, at the situation
408 $phase = loc$, the local exploration continues until the width of at least one of the 3
409 selected intervals is smaller than a fixed $\delta' \geq \delta$, with δ from (3.5).

410 Let us consider now convergence properties of the *GOSH* algorithm. The first
411 result discusses a connection between the original multi-dimensional problem and
412 the reduced univariate one. To obtain the latter problem and to go to the interval
413 $[0, 1]$ an approximation $p_M(x)$ of the Peano curve of a fixed level M is applied and
414 in the course of the algorithm a lower bound U_M^* of the multi-dimensional function
415 $F(y)$ is calculated along the curve. In order to return to the original problem (1.1),
416 (1.2) in N dimensions, it is important to understand how a lower bound for $F(y)$
417 over the entire domain $[a, b]$ in \mathbf{R}^N can be obtained from U_M^* . The following theorem
418 gives the answer to this problem.

Theorem 4.1 *Let U_M^* be a lower bound along the space-filling curve $p_M(x)$ for a multi-dimensional function $F(y)$, $y \in [a, b] \subset \mathbf{R}^N$, satisfying Lipschitz condition with constant L , i.e.,*

$$U_M^* \leq F(p_M(x)), \quad x \in [0, 1].$$

Then the value

$$U^* = U_M^* - 2^{-(M+1)}L\sqrt{N}$$

419 is a lower bound for $F(y)$ over the entire region $[a, b]$.

420 **Proof.** See [28] or the recent monograph [44] for the proof of this result. \square

421 Theorem 4.1 is important because it links the multi-dimensional problem (1.1),
422 (1.2) to the one-dimensional problem (1.3), (1.4), so we can concentrate our attention
423 on the convergence properties in the one-dimensional interval $[0, 1]$. Let us suppose
424 that the maximal number of generated trial points tends to infinity, and prove that
425 the infinite sequence of trial points generated by the *GOSH* convergence to any
426 point of the one-dimensional search domain. This kind of convergence is called
427 *everywhere dense* convergence.

428 **Theorem 4.2** *If $\delta = 0$ in (3.5), then for any point $x \in [0, 1]$ and any $\eta > 0$ there*
429 *exists an iteration number $k(\eta) \geq 1$ and a trial point $x^{i(k)}$, $k > k(\eta)$, such that*
430 *$|x - x^{i(k)}| < \eta$.*

431 **Proof.** In the selection Step 2 of the algorithm the two phases, local and global,
432 are alternated. In the local phase of *GOSH* an interval is subdivided only if its

433 width is greater than a fixed $\delta' > 0$, δ' from Step 3 of the scheme *GOSH*. When
 434 the width of the selected interval becomes less than δ' , the algorithm switches to the
 435 global phase. Since it is assumed that $\delta = 0$ in (3.5), and since the one-dimensional
 436 search region has a finite length and δ' is a positive finite number, then there exists
 437 a finite iteration number $j = j(\delta')$ such that, for all iterations greater than j , only
 438 the global phase will be used during the work of the *GOSH*.

439 In the global phase the algorithm *GOSH* always selects for partitioning at least
 440 one interval d_t from the group of largest intervals (in Fig. 4 the group with index 1).
 441 Infact there always exists a sufficiently large estimate H_∞ of the Hölder constant
 442 H , such that the interval d_t is the nondominated interval with respect to H_∞ , and
 443 condition (3.5) is satisfied. Therefore, at each iteration, the intervals with the largest
 444 width will be partitioned into three subintervals of the length equal to a third of the
 445 length of the subdivided interval. Notice that each group of intervals contains only
 446 a finite number of intervals since the interval is finite and all its subintervals have a
 447 finite length. Thus, after a sufficiently large number of iterations $k > k(\eta)$, all the
 448 intervals of the group with the maximal width will be partitioned. Such a procedure
 449 will be repeated with a new group of the largest intervals (the group with index 2
 450 in Fig. 4) and so on until the largest intervals of the current partition will have the
 451 length smaller than η . As a result, in the neighborhood of radius η of any point in
 452 $[0, 1]$ there will exist at least one trial point generated by the *GOSH*. \square

453 5 Numerical experiments

454 In this section, results of some numerical experiments are presented. The new al-
 455 gorithm *GOSH* has been compared with the original *DIRECT* method [7] and its
 456 locally-biased modification *LBDirect* proposed in [8, 9]. In order to show the use-
 457 fulness of the two-phase approach, the *GOSH* has been compared with its simplified
 458 version (called *CORE* hereinafter) that does not apply the local phase at all and
 459 only the global phase is used.

460 Ten different classes of functions generated by the GKLS-generator, a free soft-
 461 ware downloadable from <http://wwwinfo.deis.unical.it/~yaro/GKLS.html> and de-
 462 scribed in [10] have been used in the experiments. This generator constructs classes
 463 of multi-dimensional and multiextremal test functions with known global and lo-
 464 cal minima: each function is obtained by a paraboloid, systematically distorted by
 465 polynomials. Each class contains 100 test functions with the same number of local
 466 minima. In order to generate a specific class, only five parameters should be de-
 467 fined by the user (see Table 1), and it possible to generate harder or simpler test
 468 classes very easily. For example, a more difficult test class can be obtained either
 469 by decreasing the radius r^* of the attraction region of the global minimizer or by
 470 increasing the distance d from the paraboloid vertex to the global minimizer. In
 471 Table 1 we can see a complete description of the 10 classes that we have used in the
 472 experiments, for a total of 1000 test functions, in dimensions $N = 2, 3, 4, 5$, and 6.
 473 For each dimension two different classes, a simple class and a hard one, have been
 474 generated. The number of local minima m was taken equal to 10 and the global

Class	Difficulty	N	f^*	m	d	r^*
1	Simple	2	-1.0	10	0.90	0.20
2	Hard	2	-1.0	10	0.90	0.10
3	Simple	3	-1.0	10	0.66	0.20
4	Hard	3	-1.0	10	0.90	0.20
5	Simple	4	-1.0	10	0.66	0.20
6	Hard	4	-1.0	10	0.90	0.20
7	Simple	5	-1.0	10	0.90	0.40
8	Hard	5	-1.0	10	0.90	0.30
9	Simple	6	-1.0	10	0.90	0.40
10	Hard	6	-1.0	10	0.90	0.30

Table 1: *Description of 10 classes of test functions used in experiments*

475 minimum f^* was fixed to -1 for all the classes. In Fig. 1-left, an example of the
476 test function no. 4 belonging to the class 1 is shown.

477 Let us describe the stopping rules used in the experiments. The tested algorithms
478 stopped their work either when the maximal number of trials T_{max} , equal to 10^6 was
479 reached. Remind, that the GKLS generates problems with known minima. This
480 gives the possibility to use the vicinity of trials to the global minimizer as a measure
481 of success of the work of algorithms and to construct an appropriate stopping rule. Let
482 us denote as y_i^* the global minimizer of the i -th function of a test class, $1 \leq i \leq 100$.
483 Then, the following condition can be applied.

484 **Stopping criterion.** A method stops its work on the i -th function of a class
485 when it generates a trial point falling in a ball B_i having a radius ρ and the center
486 at the global minimizer of the i -th function, i.e.,

$$B_i = \{y \in R^N : \|y - y_i^*\| \leq \rho\}, \quad 1 \leq i \leq 100. \quad (5.1)$$

In the experiments, the radius ρ in (5.1) was fixed equal to $0.01\sqrt{N}$ for classes
1, 2, 3, 4, and 5, and $0.02\sqrt{N}$ for classes 6, 7, 8, 9, and 10. It should be added also
that the parameter ξ in (3.1) was fixed as follows

$$\xi = 10^{-4} \cdot |f_{min}(k)|,$$

487 where $f_{min}(k)$ is the current best function value. This choice has been considered by
488 many authors (see [8, 9]), in particular, it has been used in the *DIRECT* method
489 [7] with the most robust results. For this reason, in our experiments the same
490 value was used, as well. Notice that for the *DIRECT* and *LBDirect* methods it
491 is recommended (see, e.g., [7]) to verify stopping conditions after the end of each
492 iteration and this rule has been used in our experiments since the usage of the rule
493 (5.1) gives an insignificant improvement only.

494 The value of the parameter δ in (3.5) was fixed equal to 10^{-4} for classes 1 and
495 2, 10^{-7} for classes 3 and 4, 10^{-9} for the class 5, 10^{-10} for classes 6 and 7, 10^{-11} for
496 classes 8, 10 and equal to 10^{-12} for the class 9. The parameter δ' in Step 3 of the
497 algorithm *GOSH* was chosen equal to δ .

498 In the algorithms *GOSH* and *CORE*, an M -approximation of the Peano curve
 499 has been considered. In particular the level M of the curve must be chosen taking
 500 in mind the constraint $NM < K$, where N is the dimension of the problem and K
 501 is the number of digits in the mantissa depending on the computer that is used for
 502 the implementation (see [44] for more details). In our experiments we had $K = 52$,
 503 thus the value $M = 10$ has been used for classes 1–8 and $M = 8$ for classes 9 and
 504 10.

505 In the *GOSH* algorithm we must fix the parameters *IglobMax* and *IlocMax*, in
 506 Step 1.1 and Step 1.2, that specify the maximal allowed number of iterations exe-
 507 cuted on the global and local phase, respectively, before making the general security
 508 iteration, in which the nondominated intervals in the entire domain are selected.
 509 Different choices of these parameters can affect the speed of the search towards the
 510 global solution. For this reason, a sensitivity analysis with 6 different values of the
 511 parameters *IglobMax* and *IlocMax* for each class has been executed. The obtained
 512 results are shown in Table 2. For each class the average and the maximal number
 513 of function evaluations calculated for all the 100 functions is reported. The best
 514 results are shown in bold.

515 Table 3 shows results of experiments comparing the behavior of the *GOSH*
 516 method with the algorithms *CORE*, *DIRECT*, and *LBDirect* on the 10 classes of
 517 test functions. Taking into account the sensitivity analysis, the following values of
 518 the two parameters of *GOSH* have been chosen: *IlocMax* = 5 for classes 1, 5, 8,
 519 *IlocMax* = 10 for classes 4, 6, 7 and *IlocMax* = 15 for classes 2, 3, 9, 10. *IglobMax*
 520 was fixed equal to 5 for classes 1, 2, 3, 5, 8, 9, *IglobMax* = 15 for the class 10 and
 521 equal to 20 for classes 4, 6 and 7. The values of these two parameters corresponding
 522 to the best result in relation to the column “Max” of Tables 2 have been chosen.

523 Table 3 illustrates results of experiments with all the 10 classes and the four
 524 methods. Notice that in the column “Average” the symbol “ > ” means that,
 525 after performing T_{max} iterations, the global minimum has not been found for all
 526 functions of the class. The column “Max” reports the maximum number of function
 527 evaluations required to satisfy the stopping criterion for all the 100 functions of
 528 the class: the notation 1000000(*i*) means that after evaluating 1000000 trials, the
 529 method was not able to find the global solution for “*i*” functions of the considered
 530 class. The best results are shown in bold.

531 Finally, in Fig. 5 the behavior of the four methods for the function no. 55 of
 532 the class 2 is shown. In the first row Figure 5 (a) shows 1541 trials generated by
 533 *DIRECT* to find the global minimum of the problem and (b) 2281 trials produced
 534 by the *LBDirect*. In the second row Figure 5 (c) shows 597 trial points calculated
 535 by the *CORE* and (d) 269 produced by the *GOSH* algorithm to solve the same
 536 problem. Trial points chosen by the “local-phase” strategy are shown in red.

537 6 A brief conclusion

538 The problem of global minimization of a multi-dimensional, non-differentiable, and
 539 multiextremal function satisfying the Lipschitz condition over a hyperinterval, with

N	IlocMax	IglobMax	Average		Maximum	
			Simple class	Hard class	Simple class	Hard class
2	5	5	180.70	560.00	521	1691
	5	15	191.16	565.32	1009	3345
	10	5	184.50	563.10	531	1683
	15	5	184.50	563.10	531	1683
	15	15	194.36	569.29	1017	3337
	10	20	197.96	568.44	1199	3367
3	5	5	895.12	1733.94	3895	7335
	5	15	930.55	1683.20	6389	6651
	10	5	917.52	1745.14	3879	7337
	15	5	920.44	1784.74	3839	7347
	15	15	961.82	1698.24	6379	6655
	10	20	977.94	1693.02	6769	6589
4	5	5	8904.92	18523.44	139409	207665
	5	15	10074.92	17625.32	243635	197053
	10	5	8892.94	18553.14	139469	207675
	15	5	8904.48	18541.28	139465	207589
	15	15	10084.68	17633.54	243417	197119
	10	20	10956.02	17466.18	309549	194499
5	5	5	6437.50	18154.77	37829	107637
	5	15	6441.84	18108.82	38319	121363
	10	5	6063.46	18166.36	29319	107749
	15	5	6434.54	18361.00	37837	107757
	15	15	6437.98	18158.98	38277	122413
	10	20	6130.40	18401.63	27113	157107
6	5	5	25271.45	99318.66	151651	565015
	5	15	26968.77	104292.00	299723	538787
	10	5	25348.27	99285.62	150357	565231
	15	5	25265.09	99262.50	149281	565031
	15	15	27007.13	104281.72	299541	538751
	10	20	28276.60	109029.94	373875	616875

Table 2: *Results of the sensitivity analysis. The best values are shown in bold.*

540 an unknown Lipschitz constant has been considered in this paper. An approach
541 based on the reduction of the dimension by using numerical approximations to
542 space-filling curves in order to pass from the original Lipschitz multi-dimensional
543 problem to a univariate one satisfying the Hölder condition has been used. It has
544 been shown that it is possible to organize a simultaneous work with multiple es-
545 timates of the Hölder constant. Such a kind of techniques has been proposed for
546 Lipschitz optimization in 1994 in [19] and for a long time created difficulties in
547 the framework of Hölder global optimization. A geometric technique working with a
548 number of possible Hölder constants chosen from a set of values varying from zero to
549 infinity has been proposed and an accelerating “two-phase” technique that performs
550 a smart balancing of the local and global information has been introduced. Con-
551 ditions ensuring convergence of the method *GOSH* to the global minimizers have
552 been established. Extensive numerical experiments executed on 1000 test functions
553 have shown a very promising performance of the proposed algorithm with respect

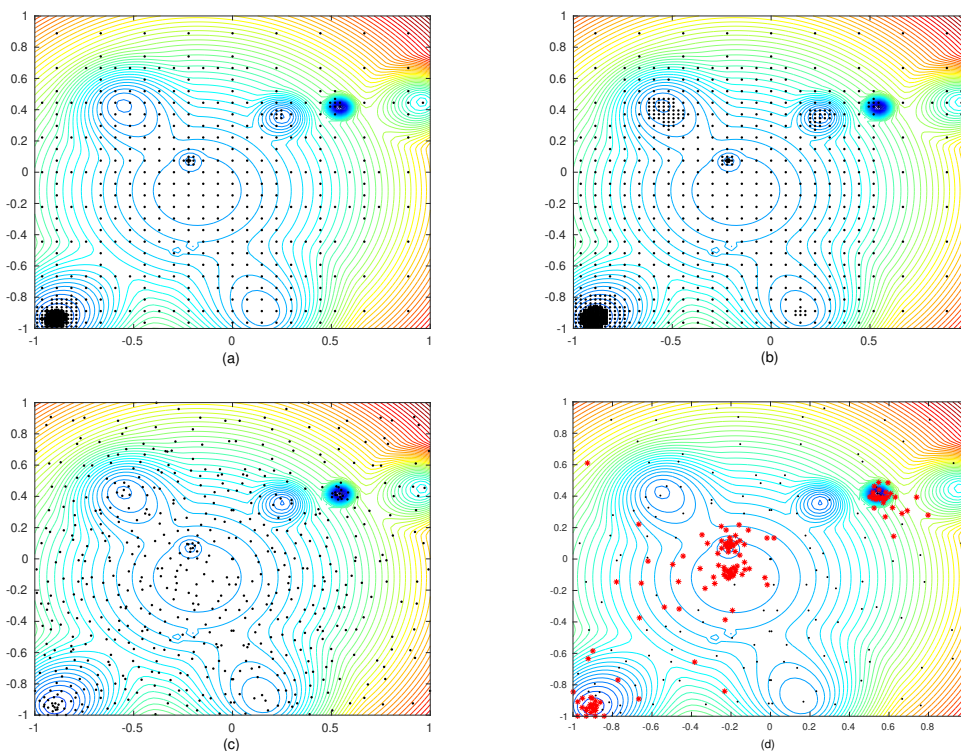


Figure 5: *Function no.55, class 2.* (a): 1541 trials generated by *DIRECT* and (b) 2281 by *LBDirect*. (c): 597 trials calculated by *CORE* and (d) 257 produced by the *GOSH*. Trial points chosen by the “local-phase” strategy are shown in red by the symbol “*”.

554 to its direct competitors, in particular for hard problems. Thus, one of the mostly
 555 abstract mathematical objects – space-filling curves – have been used to develop a
 556 practical derivative-free global optimization algorithm that can be successfully used
 557 in numerical computations.

558 **Acknowledgement.** The research of Ya.D. Sergeyev was supported by the Russian
 559 Science Foundation, project No 15-11-30022 “Global optimization, supercomputing
 560 computations, and applications”.

561 References

- 562 [1] K. A. Barkalov and V. P. Gergel. Parallel global optimization on GPU. *J.*
 563 *Global Optim.*, 66(1):3–20, 2016.
- 564 [2] A. R. Butz. Space filling curves and mathematical programming. *Information*
 565 *and Control*, 12(4):313–330, 1968.

Class	Average number of trials				Maximal number of trials			
	DIRECT	LBDirect	CORE	GOSH	DIRECT	LBDirect	CORE	GOSH
1	208.54	304.28	174.24	180.70	1159	2665	565	521
2	1081.42	1291.70	622.60	563.10	3201	4245	1749	1683
3	1140.68	1893.02	1153.64	920.44	13369	20779	5267	3839
4	>42334.36	5245.72	2077.60	1693.02	1000000(4)	32603	9809	6589
5	>47768.28	21932.94	10628.86	8904.92	1000000(4)	179383	162183	139409
6	>95908.99	74193.53	25875.16	17466.18	1000000(7)	372633	319493	194499
7	>33878.09	31955.06	7306.04	6130.40	1000000(3)	146623	36819	27113
8	>149578.61	>93876.77	28391.70	18154.77	1000000(13)	1000000(1)	153323	107637
9	>244382.63	184266.74	33366.14	25265.09	1000000(23)	873617	161577	149281
10	>549165.37	>441282.91	132415.20	104281.72	1000000(49)	1000000(19)	707543	538751

Table 3: *Results of experiments*

- 566 [3] J. M. Calvin and A. Žilinskas. One-dimensional p-algorithm with convergence
567 rate $o(n^{-3+\delta})$ for smooth functions. *J. of Optimization Theory and Applications*,
568 106(2):297–307, 2000.
- 569 [4] Yu. G. Evtushenko and M. Posypkin. A deterministic approach to global box-
570 constrained optimization. *Optimization Letters*, 7(4):819–829, 2013.
- 571 [5] D. Famularo, P. Pugliese, and Ya. D. Sergeyev. A global optimization technique
572 for checking parametric robustness. *Automatica*, 35:1605–1611, 1999.
- 573 [6] D. E. Finkel and C. T. Kelley. Additive scaling and the DIRECT algorithm.
574 *J. Global Optim.*, 36(4):597–608, 2006.
- 575 [7] M. J. Gablonsky. DIRECT v2.04 fortran code with documentation. Technical
576 report, <http://www4.ncsu.edu/~ctk/SOFTWARE/DIRECTv204.tar.gz>, 2001.
- 577 [8] M. J. Gablonsky. Modifications of the DIRECT algorithm. Technical report,
578 Ph.D thesis, North Carolina State University, Raleigh, NC, 2001.
- 579 [9] M. J. Gablonsky and C. T. Kelley. A locally-biased form of the DIRECT
580 algorithm. *J. Global Optim.*, 21:27–37, 2001.
- 581 [10] M. Gaviano, D. E. Kvasov, D. Lera, and Ya. D. Sergeyev. Algorithm 829:
582 Software for generation of classes of test functions with known local and global
583 minima for global optimization. *ACM Trans. Math. Software*, 29(4):469–480,
584 2003.
- 585 [11] V. P. Gergel, V. A., and A. V. Gergel. Adaptive nested optimization scheme
586 for multidimensional global search. *J. Global Optim.*, 66(1):35–51, 2016.
- 587 [12] V. P. Gergel, V. A. Grishagin, and R. A. Israfilov. Local tuning in nested
588 scheme of global optimization. *Procedia Computer Science*, 51:865–874, 2015.
589 (International Conference on Computational Science ICCS 2015 – Computa-
590 tional Science at the Gates of Nature).

- 591 [13] J. W. Gillard and D. E. Kvasov. Lipschitz optimization methods for fitting a
592 sum of damped sinusoids to a series of observations. *Stat. and Its Interface*,
593 10(1):59–70, 2016.
- 594 [14] E. Gourdin, B. Jaumard, and R. Ellaia. Global optimization of Hölder func-
595 tions. *J. Global Optim.*, 8:323–348, 1996.
- 596 [15] V. A. Grishagin and R. A. Israfilov. Global search acceleration in the nested
597 optimization scheme. In *AIP Conf. Proc.*, volume 1738, pages 400010:1–4. 2016.
- 598 [16] V. A. Grishagin, R. A. Israfilov, and Ya. D. Sergeyev. Convergence conditions
599 and numerical comparison of global optimization methods based on dimension-
600 ality reduction schemes. *Appl. Math. Comput.*, 318:270–280, 2018.
- 601 [17] R. Horst and P. M. Pardalos, editors. *Handbook of Global Optimization*, vol-
602 ume 1. Kluwer Academic Publishers, Dordrecht, 1995.
- 603 [18] R. Horst and H. Tuy. *Global Optimization – Deterministic Approaches*.
604 Springer–Verlag, Berlin, 1996.
- 605 [19] D. R. Jones, C. D. Perttunen, and B. E. Stuckman. Lipschitzian optimization
606 without the lipschitz constant. *J. of Optimization Theory and Applications*,
607 79:157–181, 1993.
- 608 [20] D. E. Kvasov, C. Pizzuti, and Ya. D. Sergeyev. Local tuning and partition
609 strategies for diagonal GO methods. *Numer. Math.*, 94(1):93–106, 2003.
- 610 [21] D. E. Kvasov and Ya. D. Sergeyev. Lipschitz global optimization methods in
611 control problems. *Automation and Remote Control*, 74(9):1435–1448, 2013.
- 612 [22] D. E. Kvasov and Ya. D. Sergeyev. Deterministic approaches for solving practi-
613 cal black-box global optimization problems. *Advances in Engineering Software*,
614 80:58–66, 2015.
- 615 [23] D. E. Kvasov and Ya. D. Sergeyev. A univariate global search working with a set
616 of Lipschitz constants for the first derivative. *Optimization Letters*, 3(2):303–
617 318, 2009.
- 618 [24] D. Lera and Ya. D. Sergeyev. Global minimization algorithms for Hölder func-
619 tions. *BIT*, 42(1):119–133, 2002.
- 620 [25] D. Lera and Ya. D. Sergeyev. Acceleration of univariate global optimization al-
621 gorithms working with Lipschitz functions and Lipschitz first derivatives. *SIAM*
622 *J. Optimization*, 23(1):508–529, 2013.
- 623 [26] D. Lera and Ya. D. Sergeyev. Deterministic global optimization using space-
624 filling curves and multiple estimates of Lipschitz and Hölder constants. *Com-
625 munications in Nonlinear Science and Numerical Simulation*, 23:328–342, 2015.

- 626 [27] D. Lera and Ya. D. Sergeyev. An information global minimization algorithm
627 using the local improvement technique. *J. Global Optim.*, 48(1):99–112, 2010.
- 628 [28] D. Lera and Ya. D. Sergeyev. Lipschitz and Hölder global optimization using
629 space-filling curves. *Appl. Numer. Maths.*, 60:115–129, 2010.
- 630 [29] G. Liuzzi, S. Lucidi, and V. Piccialli. A direct-based approach exploiting lo-
631 cal minimizations for the solution for large-scale global optimization problem.
632 *Computational Optimization and Applications*, 45(2):353–375, 2010.
- 633 [30] G. Liuzzi, S. Lucidi, and V. Piccialli. A partition-based global optimization
634 algorithm. *J. Global Optim.*, 48(1):113–128, 2010.
- 635 [31] R. Paulavičius, L. Chiter, and J. Žilinskas. Global optimization based on bisection
636 of rectangles, function values at diagonals, and a set of Lipschitz constants.
637 *J. Global Optim.*, 2017. In press.
- 638 [32] R. Paulavičius, Ya. D. Sergeyev, D. E. Kvasov, and J. Žilinskas. Globally-biased
639 DISIMPL algorithm for expensive global optimization. *J. Global Optim.*, 59(2-
640 3):545–567, 2014.
- 641 [33] R. Paulavičius and J. Žilinskas. *Simplicial Global Optimization*. SpringerBriefs
642 in Optimization. Springer, New York, 2014.
- 643 [34] J. D. Pintér. *Global Optimization in Action (Continuous and Lipschitz Opti-
644 mization: Algorithms, Implementations and Applications)*. Kluwer Academic
645 Publishers, Dordrecht, 1996.
- 646 [35] J. D. Pintér. Global optimization: software, test problems, and applications. In
647 P. M. Pardalos and H. E. Romeijn, editors, *Handbook of Global Optimization*,
648 volume 2, pages 515–569. Kluwer Academic Publishers, Dordrecht, 2002.
- 649 [36] S. A. Piyavskij. An algorithm for finding the absolute extremum of a function.
650 *USSR Comput. Math. Math. Phys.*, 12(4):57–67, 1972. (In Russian: Zh. Vychisl.
651 Mat. Mat. Fiz., 12(4) (1972), pp. 888–896).
- 652 [37] F. P. Preparata and M. I. Shamos. *Computational Geometry: An Introduction*.
653 Springer–Verlag, New York, 1993.
- 654 [38] H. Sagan. *Space-Filling Curves*. Springer, New York, 1994.
- 655 [39] Ya. D. Sergeyev. An information global optimization algorithm with local tun-
656 ing. *SIAM J. Optimization*, 5(4):858–870, 1995.
- 657 [40] Ya. D. Sergeyev. A one-dimensional deterministic global minimization algo-
658 rithm. *Comput. Math. Math. Phys.*, 35(5):705–717, 1995.

- 659 [41] Ya. D. Sergeyev, P. Daponte, D. Grimaldi, and A. Molinaro. Two methods for
660 solving optimization problems arising in electronic measurements and electrical
661 engineering. *SIAM J. Optim.*, 10(1):1–21, 1999.
- 662 [42] Ya. D. Sergeyev and V. A. Grishagin. Sequential and parallel algorithms for
663 global optimization. *Optim. Methods Softw.*, 3:111–124, 1994.
- 664 [43] Ya. D. Sergeyev and D. E. Kvasov. *Diagonal Global Optimization Methods*.
665 Fizmatlit, Moscow, 2008. (In Russian).
- 666 [44] Ya. D. Sergeyev, R. G. Strongin, and D. Lera. *Introduction to Global Optimiza-*
667 *tion Exploiting Space-Filling Curves*. SpringerBriefs in Optimization. Springer,
668 New York, 2013.
- 669 [45] R. G. Strongin. *Numerical Methods in Multiextremal Problems: Information-*
670 *Statistical Algorithms*. Nauka, Moscow, 1978. (In Russian).
- 671 [46] R. G. Strongin and Ya. D. Sergeyev. Global optimization: fractal approach and
672 non-redundant parallelism. *J. Global Optim.*, 27:25–50, 2003.
- 673 [47] R. G. Strongin and Ya. D. Sergeyev. *Global Optimization with Non-Convex*
674 *Constraints: Sequential and Parallel Algorithms*. Kluwer Academic Publishers,
675 Dordrecht, 2000. (2nd ed., 2012; 3rd ed., 2014, Springer, New York).
- 676 [48] A. Žilinskas. On similarities between two models of global optimization: Sta-
677 tistical models and radial basis functions. *J. Global Optim.*, 48(1):173–182,
678 2010.
- 679 [49] A. Žilinskas and J. Žilinskas. Parallel hybrid algorithm for global optimization
680 of problems occurring in mds-based visualization. *Comput. Math. Appl.*, 52(1-
681 2):211–224, 2006.
- 682 [50] A. Žilinskas and J. Žilinskas. A hybrid global optimization algorithm for non-
683 linear least squares regression. *J. Global Optim.*, 56(2):265–277, 2013.
- 684 [51] A. A. Zhigljavsky. *Theory of Global Random Search*. Kluwer Academic Pub-
685 lishers, Dordrecht, 1991.
- 686 [52] A. A. Zhigljavsky and A. Žilinskas. *Stochastic Global Optimization*. Springer,
687 New York, 2008.