Software for Generation of Classes of Test Functions with Known Local and Global Minima for Global Optimization

MARCO GAVIANO
Università di Cagliari, Italy
DMITRI E. KVASOV
Università di Roma "La Sapienza", Italy, and University of Nizhni Novgorod, Russia
DANIELA LERA
Università di Cagliari, Italy
and
YAROSLAV D. SERGEYEV *
Università della Calabria, Italy, and University of Nizhni Novgorod, Russia

A procedure for generating non-differentiable, continuously differentiable, and twice continuously differentiable classes of test functions for multiextremal multidimensional box-constrained global optimization and a corresponding package of C subroutines are presented. Each test class consists of 100 functions. Test functions are generated by defining a convex quadratic function systematically distorted by polynomials in order to introduce local minima. To determine a class, the user defines the following parameters: (i) problem dimension, (ii) number of local minima, (iii) value of the global minimum, (iv) radius of the attraction region of the global minimizer, (v) distance from the global minimizer to the vertex of the quadratic function. Then, all other necessary parameters are generated randomly for all 100 functions of the class. Full information about each test function including locations and values of all local minima is supplied to the user. Partial derivatives are also generated where possible.

Categories and Subject Descriptors: G.1.6 [**Numerical Analysis**]: Optimization—*Global optimization*; *Nonlinear programming*; G.4 [**Mathematics of Computing**]: Mathematical Software—*Certification and testing*

General Terms: Algorithms, Experimentation, Performance

Additional Key Words and Phrases: Global optimization, test problems generation, known local minima

*Author to whom correspondence should be addressed, e-mail: yaro@si.deis.unical.it

This research was partially supported by the following projects: FIRB RBAU01JYPN, FIRB RBNE01WBBB, and RFBR 01-01-00587.
Authors' addresses: M. Gaviano and D. Lera, Dipartimento di Matematica, Università di Cagliari, Via Ospedale, 72, 09100 Cagliari – Italy; D.E. Kvasov and Ya.D. Sergeyev, DEIS, Università della Calabria, Via P.Bucci, Cubo 41C, 87036 Rende (CS) – Italy.

1. INTRODUCTION

A wide literature is dedicated to development of numerical algorithms for solving the global optimization problem (see, for example, references given in Horst and Pardalos [1995]). The problem may be formulated as

$$f^* = f(x^*) = \min_{x \in \mathcal{F}} f(x), \quad \mathcal{F} \subset \mathbb{R}^N, \tag{1}$$

where $f(x)$ is a multiextremal and possibly non-differentiable function and $\mathcal{F}$ is a compact set.

One of the approaches to studying and verifying validity of numerical algorithms is their comparison on test problems (see, e.g., Ali et al. [2003], Dixon and Szegö [1978], Facchinei et al. [1997], Floudas and Pardalos [1990], Floudas et al. [1999], Gaviano and Lera [1998], Horst and Pardalos [1995], Kalantari and Rosen [1986], Khoury et al. [1993], Li and Pardalos [1992], Locatelli [2003], Moré et al. [1981], Moshirvaziri [1994], Moshirvaziri et al. [1996], Pardalos [1987; 1991], Pintér [2002], Schittkowski [1980; 1987], Schoen [1993], Sung and Rosen [1982]). Many global optimization tests were taken from real-life problems and for this reason comprehensive information about them is not available. The number of local minima may be unknown, as well as their locations, regions of attraction, and even values (including that of the global minimum).

Recently Gaviano and Lera [1998] introduced two types of functions with a priori known local minima and their regions of attraction. The tests proposed take a convex quadratic function (called hereafter 'paraboloid') systematically distorted by cubic polynomials and by quintic polynomials to introduce local minima and to construct test functions that are continuously differentiable in some region $\Omega \supseteq \mathcal{F}$ (called hereafter 'D-type' test functions) and twice continuously differentiable in $\Omega \supseteq \mathcal{F}$ (called hereafter 'D2-type' test functions), where $\mathcal{F}$ is from (1) and Ω is a hyperrectangle.

To define a function of one of these types it is necessary to determine a number of correlated parameters. Unfortunately, the correlations do not allow simple and fast generation of the test functions. Additionally, generation of different functions having similar properties becomes difficult and non-intuitive when dimension and/or number of local minima increase.

In this paper, in addition to the two types of test functions from Gaviano and Lera [1998], the third type of non-differentiable test functions (called hereafter 'ND-type') is presented and a generator for these three types of test functions is proposed. The software to be introduced generates classes of test functions and provides procedures for calculating the first order derivatives of the D-type test functions and the first and second order derivatives of the D2-type test functions.

Each class contains 100 functions and is defined by the following parameters (the only ones to be determined by the user):

(1) problem dimension;
(2) number of local minima;
(3) value of the global minimum;
(4) radius of the attraction region of the global minimizer;
(5) distance from the global minimizer to the vertex of the paraboloid.

The other necessary parameters (i.e., locations of all minimizers, their regions of attraction, and values of minima) are chosen randomly by the generator. After generation a special notebook containing a complete description of all the functions from the generated class is supplied to the user.

The rest of the paper is structured as follows. In Section 2, a mathematical description of the three types of test functions is given. Section 3 introduces the generator and details of its implementation. Section 4 is devoted to usage of the generator.

2. MATHEMATICAL DESCRIPTION

In this section, the three types of test functions are briefly described. Let us start with the D-type and D2-type functions (see Gaviano and Lera [1998]). A function $f(x)$ of the D-type is determined over an admissible region $\Omega \supseteq \mathcal{F}$, where $\mathcal{F}$ is from (1) and

$$\Omega = [a, b\,] = \{x \in \mathbb{R}^N \,:\, a \le x \le b\}, \quad a < b, \;\; a, b \in \mathbb{R}^N. \tag{2}$$

The function is constructed by modifying a paraboloid Z:

$$Z :\;\; g(x) = \|x - T\|^2 + t, \quad x \in \Omega, \tag{3}$$

(hereafter $\|\cdot\|$ denotes the Euclidean norm) with the minimum t at a point $T \in \text{int}(\Omega)$ in such a way that the resulting function $f(x)$ has m, $m \ge 2$, local minimizers: point T from (3) (we denote it by $M_1 := T$) and points

$$M_i \in \text{int}(\Omega), \quad M_i \ne T, \;\; M_i \ne M_j, \quad i, j = 2, \ldots, m, \;\; i \ne j. \tag{4}$$

The paraboloid Z from (3) is modified by a function $C_i(x)$, which is constructed by using cubic polynomials within balls $S_i \subset \Omega$ around each point M_i, $i = 2, \ldots, m$, where

$$S_i = \{x \in \mathbb{R}^N :\, \|x - M_i\| \;\le \rho_i,\, \rho_i > 0\}, \quad i = 1, \ldots, m. \tag{5}$$

Functions $Q_i(x)$, $i = 2, \ldots, m$, use quintic polynomials to determine the D2-type test functions.

Selection of radii ρ_i, $i = 1, \ldots, m$, is carried out in such a manner that sets S_i from (5) do not overlap:

$$S_i \cap S_j = \varnothing, \quad i, j = 1, \ldots, m, \;\; i \ne j. \tag{6}$$

It is not required that each attraction region S_i, $i = 1, \ldots, m$, be entirely contained in Ω. Note that we use the notation "attraction region" with respect to the balls S_i, $i = 1, \ldots, m$, just for simplicity. Naturally, definition of the real attraction region for each local minimizer will depend on the method used for optimization and will change from one algorithm to another.

Formally, D-type functions [Gaviano and Lera 1998] are described as follows:

$$f(x) = \begin{cases} C_i(x), & x \in S_i,\, i \in \{2, \ldots, m\}, \\ g(x), & x \notin S_2 \cup \ldots \cup S_m\,, \end{cases} \tag{7}$$

where $g(x)$ is from (3), sets S_i, $i = 2, \ldots, m$, from (5) satisfy (6), and

$$C_i(x) = \left(\frac{2}{\rho_i^2} \frac{< x - M_i,\, T - M_i >}{\|x - M_i\|} - \frac{2}{\rho_i^3} A_i \right) \|x - M_i\|^3 +$$

$$+\left(1-\frac{4}{\rho_i}\frac{<x-M_i,\,T-M_i>}{\|x-M_i\|}+\frac{3}{\rho_i^2}A_i\right)\|x-M_i\|^2+f_i. \tag{8}$$

In (8) radii ρ_i, $i=2,\dots,m$, determine the sets S_i from (5), $<\cdot,\cdot>$ denotes the usual scalar product, and the values A_i, $i=2,\dots,m$, are found as

$$A_i=\|T-M_i\|^2+t-f_i, \tag{9}$$

where $f_1=t$ and f_i, $i=2,\dots,m$, are the function values at local minimizers M_i:

$$f_i=\min\{g(x)\,:x\in B_i\}-\gamma_i,\quad \gamma_i>0, \tag{10}$$

where B_i is the boundary of the ball S_i:

$$B_i=\{x\in\mathbb{R}^N:\ \|x-M_i\|\ =\rho_i,\,\rho_i>0\},\quad i=2,\dots,m, \tag{11}$$

and γ_i is a parameter ensuring that the value f_i is less than the minimum of the paraboloid Z from (3) over B_i.

Analogously, D2-type functions [Gaviano and Lera 1998] are defined by

$$f(x)=\begin{cases} Q_i(x), & x\in S_i,\, i\in\{2,\dots,m\},\\ g(x), & x\notin S_2\cup\ldots\cup S_m\,,\end{cases} \tag{12}$$

where

$$\begin{aligned}Q_i(x)=&\left[-\frac{6}{\rho_i^4}\frac{<x-M_i,\,T-M_i>}{\|x-M_i\|}+\frac{6}{\rho_i^5}A_i+\frac{1}{\rho_i^3}(1-\frac{\delta}{2})\right]\|x-M_i\|^5+\\ &\left[\frac{16}{\rho_i^3}\frac{<x-M_i,\,T-M_i>}{\|x-M_i\|}-\frac{15}{\rho_i^4}A_i-\frac{3}{\rho_i^2}(1-\frac{\delta}{2})\right]\|x-M_i\|^4+\\ &\left[-\frac{12}{\rho_i^2}\frac{<x-M_i,\,T-M_i>}{\|x-M_i\|}+\frac{10}{\rho_i^3}A_i+\frac{3}{\rho_i}(1-\frac{\delta}{2})\right]\|x-M_i\|^3+\\ &\frac{1}{2}\delta\|x-M_i\|^2+f_i\end{aligned} \tag{13}$$

with A_i and f_i, $i=2,\dots,m$, from (9) and (10), and δ is an arbitrary positive real number (see [Gaviano and Lera 1998, Lemma 3.1]).

The properties of these functions have been studied by Gaviano and Lera [1998]. In particular, the following results can be proved:

i. D-type functions (7)–(8) are continuously differentiable in Ω [Gaviano and Lera 1998, Lemma 2.1].

ii. D2-type functions (12)–(13) are twice continuously differentiable in Ω [Gaviano and Lera 1998, Lemma 3.1].

Let us now describe the ND-type test functions, which are continuous in Ω but non-differentiable in the whole region Ω. An analogous procedure is considered: the paraboloid Z from (3) is modified by a function $P_i(x)$ constructed from second degree polynomials within each region $S_i\subset\Omega$ from (5) in such a way that the resulting function $f(x)$ is continuous in the feasible region Ω from (2), differentiable at each local minimizer M_i, $i=2,\dots,m$, from (4), but generally non-differentiable at the points of the boundaries B_i of the balls S_i, $i = 2, \ldots, m$, determined by (11). That is,

$$f(x) = \begin{cases} P_i(x), & x \in S_i,\ i \in \{2, \ldots, m\}, \\ g(x), & x \notin S_2 \cup \ldots \cup S_m\,, \end{cases} \tag{14}$$

where $g(x)$ is from (3), sets S_i, $i = 2, \ldots, m$, from (5) satisfy (6), and

$$P_i(x) = \left(1 - \frac{2}{\rho_i} \frac{< x - M_i,\, T - M_i >}{\|x - M_i\|} + \frac{1}{\rho_i^2} A_i\right) \|x - M_i\|^2 + f_i. \tag{15}$$

In (15) the values ρ_i, A_i, and f_i ($i = 2, \ldots, m$) are determined in the same way as for the D- and D2-type functions by formulae (5)–(6), (9), and (10), respectively.

3. GENERATION OF TESTS CLASSES

As one can see from the previous section, all three function types have many parameters to be coordinated. Moreover, their characteristics (for example, the mutual positions of the local minimizers, the global minimizer, and the paraboloid vertex; the size of the attraction regions of local minimizers; the function values at local minima) influence the properties of the test functions significantly from the point of view of global optimization algorithms. For example, coincidence of the global minimizer with the paraboloid vertex leads to generation of too simple functions. Existence of many deep minima having narrow regions of attraction can lead to the impossibility of global minimizer location even by the most "intelligent" global optimization algorithms. All these features should be added to the general scheme from Section 2 in order to obtain well-structured test classes.

In the generator, the user sets just a few parameters defining a desirable class while all the other parameters are chosen randomly. The generator is also employed in maintaining conditions distinguishing each class – for example, the distance of the global minimizer from the minimizer of the paraboloid, dependence of the local minima values on the attraction regions sizes, etc. Thus, the generator gives the researcher the ability to construct classes of 100 test functions of arbitrary dimension with arbitrary number of local minima.

This section describes how a class consisting of D-type test functions is generated. Classes consisting of D2-type and ND-type functions are constructed analogously.

Each test class generated by the introduced software contains 100 test functions $f(x)$ and is defined by the following parameters to be fixed by the user:

(1) the problem dimension N, $N \geq 2$;
(2) the number of local minimizers m, $m \geq 2$, including the minimizer T for the paraboloid (3) (all the minimizers are chosen randomly);
(3) the global minimum value f^*, the same for all the functions of the class;
(4) the radius ρ^* of the attraction region of the global minimizer x^*;
(5) the distance r^* from the paraboloid vertex T to the global minimizer $x^* \in \Omega$ (whose coordinates are also chosen randomly).

By changing these parameters the user can create classes with different properties.

Each function of a test class is specified by its number n, $1 \leq n \leq 100$. The other parameters of the functions from (3)–(15) are chosen randomly by means of the random number generator proposed in Knuth [1997].

The input parameters f^*, r^*, and ρ^* must be chosen in such a way that the following simple conditions are satisfied:

$$f^* < t \tag{16}$$

(which means that the global minimizer is not a vertex of the paraboloid; this requirement allows us to avoid too simple functions with a global minimum at the vertex of the paraboloid Z from (3)),

$$0 < r^* < 0.5 \min_{1 \le j \le N} |b(j) - a(j)| \tag{17}$$

(i.e., the global minimizer x^* belongs to the admissible region Ω even in the case when the paraboloid vertex T is at the center of Ω), and

$$0 < \rho^* \le 0.5 r^*. \tag{18}$$

Note that it is not required that each attraction region S_i, $i = 1, \dots, m$, from (5) entirely belongs to Ω.

The admissible region Ω is taken as $\Omega = [-1, 1]^N$ and the minimal value of the paraboloid (3) is fixed at $t = 0$ by default (naturally, these parameters can be changed by the user).

Let us discuss in more detail the random procedure generating parameters for test functions. (The unique difference for the D2-type is that the parameter δ from (13) is required; this parameter is chosen randomly from the open interval $(0, \Delta)$, where Δ is a positive number taken by default $\Delta = 10$.) Hereafter the vertex T from (3) in the set (4) of local minimizers has the index 1, $M_1 := T$, and the global minimizer x^* has the index 2, $M_2 := x^*$. Naturally, among the minimizers M_i, $i = 3, \dots, m$, another global minimizer $y^* \ne x^*$ can be generated.

First, coordinates of the paraboloid vertex T, coordinates of the global minimizer x^*, and coordinates of the remaining local minimizers M_i (controlling the satisfaction of (4)) are chosen randomly. Then, the attraction regions radii ρ_i, $i \ne 2$, from (5) are determined: to do this the attraction regions of each local minimizer from (4) ($i \ne 2$ because the attraction region of the global minimizer is fixed: $\rho_2 = \rho^*$) are expanded until condition (6) is not violated. Finally, values of the function $f(x)$ at local minima M_i, $i = 3, \dots, m$, are fixed by choosing random values γ_i, $i = 3, \dots, m$, from (10) (recall that $f_1 = t$ and $f_2 = f^*$).

Let us consider these three principal operations in detail.

Coordinates of the local minimizers M_i, $i = 3, \dots, m$, from (4), coordinates of the vertex T of the paraboloid (3), and location of the global minimizer x^* are chosen randomly at the intersection of Ω and the sphere of radius r^* with a center at T so that (4) is satisfied. For the positioning of x^* we use generalized spherical coordinates

$$x_j^* := T_j + r^* \cos\phi_j \prod_{k=1}^{j-1} \sin\phi_k, \quad j = 1, \dots, N-1,$$

$$x_N^* := T_N + r^* \prod_{k=1}^{N-1} \sin\phi_k, \tag{19}$$

where the components of the vector

$$\phi = (\phi_1, \dots, \phi_N) \in \Phi = \{0 \le \phi_1 \le \pi;\ 0 \le \phi_j \le 2\pi,\ j = 2, \dots, N\}$$

are chosen randomly. In this case, if some $x_k^* \notin \Omega$, $1 \le k \le N$, this coordinate is redefined as

$$x_k^* := 2T_k - x_k^*.$$

After selection of coordinates of the paraboloid vertex T and of the global minimizer x^*, coordinates of the points M_i, $i = 3, \dots, m$, are generated in such a way that beside condition (4) the condition

$$\|M_i - x^*\| - \rho^* = \zeta, \quad \zeta > 0 \tag{20}$$

is satisfied with some positive parameter ζ. This condition follows from (6) and does not allow the local minimizers to be very close to the attraction region of the global minimizer x^*. Thus, in (20) the parameter ζ should not be too small. The value $\zeta = \rho^*$ is chosen by default.

The next step of the test function construction sets attraction regions. Each value ρ_i, $i \ne 2$, from (5) is initially calculated as half of the minimum distance between the minimizer M_i and the remaining local minimizers

$$\rho_i := 0.5 \min_{1 \le j \le m,\ j \ne i} \|M_i - M_j\|, \quad i = 1, \dots, m,\ i \ne 2,$$

$$\rho_2 := \rho^* \tag{21}$$

(in such a way that the attraction regions from (5) do not overlap). Then, an attempt to increase the values ρ_i, $i = 1, \dots, m$, $i \ne 2$ (i.e., an attempt to enlarge the attraction regions) is made:

$$\rho_i := \max\left(\rho_i, \min_{1 \le j \le m,\ j \ne i} \{\|M_i - M_j\| - \rho_j\}\right), \quad i = 1, \dots, m,\ i \ne 2. \tag{22}$$

Because of the recursive character of formulae (22), an expansion of the attraction regions depends on the order in which these regions are selected (an ascending order of the indices is chosen).

Finally, the values of the radii ρ_i are corrected by the weight coefficients w_i:

$$\rho_i := w_i\, \rho_i, \quad i = 1, \dots, m,$$

where $0 < w_i \le 1$, $i = 1, \dots, m$, and the values w_i are chosen by default as

$$w_i = 0.99, \quad i = 1, \dots, m,\ i \ne 2, \qquad \text{and} \qquad w_2 = 1. \tag{23}$$

At the last step the function values f_i, $i = 3, \dots, m$, at the local minima are generated by using formula (10), where γ_i must be specified. Each value γ_i, $i = 3, \dots, m$, is chosen (note that the values γ_1 and γ_2 are not considered because the function values $f_1 = t$ at the paraboloid vertex and $f_2 = f^*$ at the global minimizer have been fixed by the user without using (10)) as the minimum of two values generated randomly from the open intervals $(\rho_i, 2\rho_i)$ and $(0, Z_{B_i} - f^*)$, where Z_{B_i} is the minimum of the paraboloid Z from (3) over B_i from (11). In such a way, the

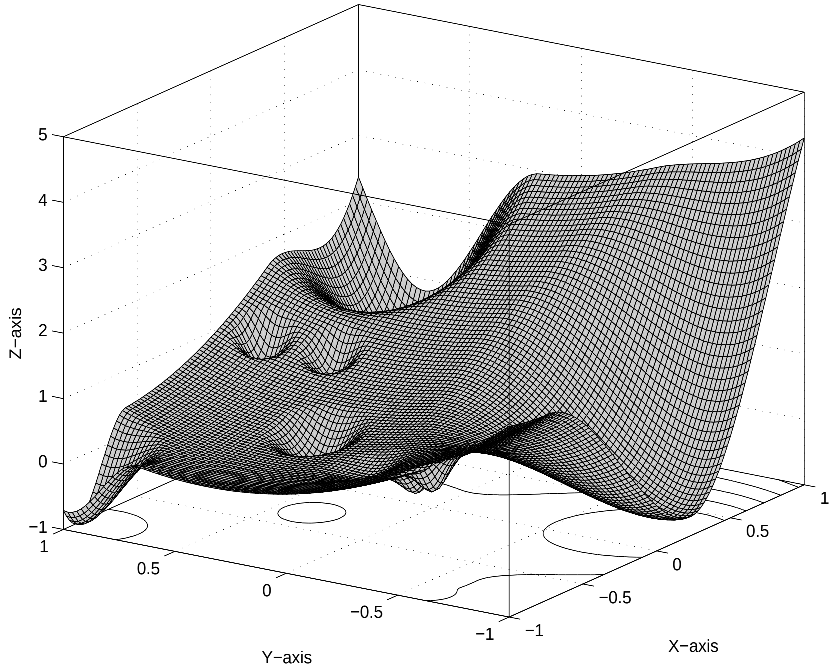

Fig. 1. The function number 9 from a class of two-dimensional D-type test functions with 10 local minima

values f_i in (10) depend on radii ρ_i of the attraction regions S_i, $i = 3, \ldots, m$, and at the same time the following condition is satisfied:

$$f^* \leq f_i, \quad i = 3, \ldots, m.$$

Note that dependence of the function values at local minima on the radii of the attraction regions is not respected by the global optimum value $f_2 = f^*$ because the user defines the function value at the global minimizer and the radius ρ^* of its region of attraction directly when choosing the corresponding test class.

Figure 1 shows an example of the D-type test function. This function is defined in the region $\Omega = [-1, 1]^2$ and is number 9 in the class of D-type functions with the following parameters:

(1) dimension $N = 2$;

(2) number of local minima $m = 10$;

(3) value of the global minimum $f^* = -1$;

(4) radius of the attraction region of the global minimizer $\rho^* = \frac{1}{3}$;

(5) distance from the global minimizer x^* to the vertex T of the paraboloid from (3) is $r^* = \frac{2}{3}$.

The generated global minimizer of this function is $x^* = (-0.911, 0.989)$ and the paraboloid minimizer is $T = (-0.711, 0.353)$.

4. USAGE OF THE TEST CLASSES GENERATOR

The generator package has been written in ANSI Standard C and successfully tested on Windows and UNIX platforms. Our implementation follows the procedure described in Section 3. First, the general structure of the package is described, then instructions for using the test classes generator (called hereafter GKLS-generator) are given.

4.1 Structure of the package

The package includes the following files:

gkls.c – the main file;

gkls.h – the header file that users should include in their application projects in order to call subroutines from the file **gkls.c**;

rnd_gen.c – the file containing the uniform random number generator proposed in Knuth [1997; 2002];

rnd_gen.h – the header file for linkage to the file **rnd_gen.c**;

example.c – an example of the GKLS-generator usage;

Makefile – an example of a UNIX makefile provided to UNIX users for a simple compilation and linkage of separate files of the application project.

For implementation details the user can consult the C codes. Note that the random number generator in **rnd_gen.c** uses the logical-and operation '&' for efficiency, so it is not strictly portable unless the computer uses two's complement representation for integer. It does not limit portability of the package because almost all modern computers are based on two's complement arithmetic.

4.2 Calling sequence for generation and usage of the tests classes

Here we describe how to generate and use classes of the ND-, D-, and D2-type test functions. Again, we concentrate on the D-type functions. The operations for the remaining two types are analogous.

To utilize the GKLS-generator the user must perform the following steps:

Step 1. Input of the parameters defining a specific test class.

Step 2. Generating a specific test function of the defined test class.

Step 3. Evaluation of the generated test function and, if necessary, its partial derivatives.

Step 4. Memory deallocating.

Let us consider these steps in turn.

4.2.1 *Input of the parameters defining a specific test class.* This step is subdivided into: (a) defining the parameters of the test class, (b) defining the admissible region Ω, and (c) checking (if necessary).

– *(a) Defining the parameters of the test class.* The parameters to be defined by the user determine a specific class (of the ND-, D- or D2-type) of 100 test functions (a specific function is retrieved by its number). There are the following parameters:

GKLS_dim – (**unsigned int**) dimension N (from (1)) of test functions; $N \geq 2$ (since multidimensional problems are considered in (1)) and $N <$ NUM_RND in **rnd_gen.h**; this value is limited by the power of **unsigned int**-representation; default $N = 2$;

GKLS_num_minima – (**unsigned int**) number m (from (4)) of local minima including the paraboloid Z minimum (from (3)) and the global minimum; $m \geq 2$; the upper bound of this parameter is limited by the power of **unsigned int**-representation; default $m = 10$;

GKLS_global_value – (**double**) global minimum value f^* of $f(x)$; condition (16) must be satisfied; the default value is -1.0 (defined in the file **gkls.h** as a constant GKLS_GLOBAL_MIN_VALUE);

GKLS_global_dist – (**double**) distance r^* from the paraboloid vertex T in (3) to the global minimizer $x^* \in \Omega$ of $f(x)$; condition (17) must be satisfied; the default value is

$$\text{GKLS\_global\_dist} \stackrel{\text{def}}{=} \min_{1 \leq j \leq N} |b(j) - a(j)| / 3,$$

where the vectors a and b determine the admissible region Ω in (2);

GKLS_global_radius – (**double**) radius ρ^* of the attraction region of the global minimizer $x^* \in \Omega$ of $f(x)$; condition (18) must be satisfied; the default value is

$$\text{GKLS\_global\_radius} \stackrel{\text{def}}{=} \min_{1 \leq j \leq N} |b(j) - a(j)| / 6.$$

The user may call subroutine *GKLS_set_default*() to set the default values of these five variables.

– *(b) Defining the admissible region* Ω. With N determined, the user must allocate dynamic arrays *GKLS_domain_left* and *GKLS_domain_right* to define the boundary of the hyperrectangle Ω. This is done by calling subroutine

int *GKLS_domain_alloc* ();

which has no parameters and returns the following error codes defined in **gkls.h**:

GKLS_OK – no errors;

GKLS_DIM_ERROR – the problem dimension is out of range; it must be greater than or equal to 2 and less than NUM_RND defined in **rnd_gen.h**;

GKLS_MEMORY_ERROR – there is not enough memory to allocate.

The same subroutine defines the admissible region Ω. The default value $\Omega = [-1, 1]^N$ is set by *GKLS_set_default*().

– *(c) Checking.* The following subroutine allows the user to check validity of the input parameters:

int *GKLS_parameters_check* ().

It has no parameters and returns the following error codes (see **gkls.h**):

GKLS_OK – no errors;

GKLS_DIM_ERROR – problem dimension error;

GKLS_NUM_MINIMA_ERROR – number of local minima error;

GKLS_BOUNDARY_ERROR – the admissible region boundary vectors are ill-defined;

GKLS_GLOBAL_MIN_VALUE_ERROR – the global minimum value is not less than the paraboloid (3) minimum value t defined in **gkls.h** as a constant GKLS_PARABOLOID_MIN;

GKLS_GLOBAL_DIST_ERROR – the parameter r^* does not satisfy (17);

GKLS_GLOBAL_RADIUS_ERROR – the parameter ρ^* does not satisfy (18).

4.2.2 *Generating a specific test function of the defined test class.* After a specific test class has been chosen (i.e., the input parameters have been determined) the user can generate a specific function that belongs to the chosen class of 100 test functions. This is done by calling subroutine

int *GKLS_arg_generate* (**unsigned int** *nf*);

where

nf – the number of a function from the test class (from 1 to 100).

This subroutine initializes the random number generator, checks the input parameters, allocates dynamic arrays, and generates a test function following the procedure of Section 3. It returns an error code that can be the same as for subroutines *GKLS_parameters_check*() and *GKLS_domain_alloc*(), or additionally:

GKLS_FUNC_NUMBER_ERROR – the number of a test function to generate exceeds 100 or it is less than 1.

GKLS_arg_generate() generates the list of all local minima and the list of the global minima as parts of the structures *GKLS_minima* and *GKLS_glob*, respectively. The first structure gathers the following information about all local minima (including the paraboloid minimum and the global one): coordinates of local minimizers, local minima values, and attraction regions radii. The second structure contains information about the number of global minimizers and their indices in the set of local minimizers. It has the following fields:

num_global_minima – (**unsigned int**) total number of global minima;

gm_index – (**unsigned int ***) list of indices of generated minimizers, which are the global ones (elements 0 to ($num\_global\_minima - 1$) of the list) and the local ones (the remaining elements of the list).

The elements of the list *GKLS_glob.gm_index* are indices to a specific minimizer in the first structure *GKLS_minima* characterized by the following fields:

local_min – (**double ****) list of local minimizers coordinates;

f – (**double ***) list of local minima values;

rho – (**double ***) list of attraction regions radii;

peak – (**double ***) list of parameters γ_i values from (10);

w_rho – (**double ***) list of parameters w_i values from (23).

The fields of these structures can be useful if one needs to study properties of a specific generated test function more deeply.

4.2.3 *Evaluation of a generated test function or its partial derivatives.* While there exists a structure *GKLS_minima* of local minima, the user can evaluate a test function (or partial derivatives of D- and D2-type functions) that is determined by its number (a parameter to the subroutine *GKLS_arg_generate*()) within the chosen test class. If the user wishes to evaluate another function within the same class he should deallocate dynamic arrays (see the next subsection) and recall the generator *GKLS_arg_generate*() (passing it the corresponding function number) without resetting the input class parameters (see subsection 4.2.1). If the user wishes to change the test class properties he should reset also the input class parameters.

Evaluation of an ND-type function is done by calling subroutine

double *GKLS_ND_func* (x).

Evaluation of a D-type function is done by calling subroutine

double *GKLS_D_func* (x).

Evaluation of a D2-type function is done by calling subroutine

double *GKLS_D2_func* (x).

All these subroutines have only one input parameter

x – (**double ***) a point $x \in \mathbb{R}^N$ where the function must be evaluated.

All the subroutines return a test function value corresponding to the point x. They return the value GKLS_MAX_VALUE (defined in **gkls.h**) in two cases: (a) vector x does not belong to the admissible region Ω and (b) the user tries to call the subroutines without generating a test function.

The following subroutines are provided for calculating the partial derivatives of the test functions (see Appendix).

Evaluation of the first order partial derivative of the D-type test functions with respect to the variable x_j (see (A.1)–(A.2) in Appendix) is done by calling subroutine

double *GKLS_D_deriv* (j, x).

Evaluation of the first order partial derivative of the D2-type test functions with respect to the variable x_j (see (A.3)–(A.4) in Appendix) is done by calling subroutine

double *GKLS_D2_deriv1* (j, x).

Evaluation of the second order partial derivative of the D2-type test functions with respect to the variables x_j and x_k (see in Appendix the formulae (A.5)–(A.6) for the case $j \neq k$ and (A.7)–(A.8) for the case $j = k$) is done by calling subroutine

double *GKLS_D2_deriv2* (j, k, x).

Input parameters for these three subroutines are:

j, k – (**unsigned int**) indices of the variables (that must be in the range from 1 to *GKLS_dim*) with respect to which the partial derivative is evaluated;

x – (**double ***) a point $x \in \mathbb{R}^N$ where the derivative must be evaluated.

All subroutines return the value of a specific partial derivative corresponding to the point x and to the given direction. They return the value GKLS_MAX_VALUE (defined in **gkls.h**) in three cases: (a) index (j or k) of a variable is out of the range [1,*GKLS_dim*]; (b) vector x does not belong to the admissible region Ω; (c) the user tries to call the subroutines without generating a test function.

Subroutines for calculating the gradients of the D- and D2-type test functions and for calculating the Hessian matrix of the D2-type test functions at a given feasible point are also provided. These are

int *GKLS_D_gradient* (x, g),

int *GKLS_D2_gradient* (x, g),

int *GKLS_D2_hessian* (x, h).

Here

x – (**double ***) a point $x \in \mathbb{R}^N$ where the gradient or Hessian matrix must be evaluated;

g – (**double ***) a pointer to the gradient vector calculated at x;

h – (**double ****) a pointer to the Hessian matrix calculated at x.

Note that before calling these subroutines the user must allocate dynamic memory for the gradient vector g or the Hessian matrix h and pass the pointers g or h as parameters of the subroutines.

These subroutines call the subroutines described above for calculating the partial derivatives and return an error code (**GKLS_DERIV_EVAL_ERROR** in the case of an error during evaluation of a particular component of the gradient or the Hessian matrix, or **GKLS_OK** if there are no errors).

4.2.4 *Memory deallocating.* When the user concludes his work with a test function he should deallocate dynamic arrays allocated by the generator. This is done by calling subroutine

void *GKLS_free* (**void**);

with no parameters.

When the user abandons the test class he should deallocate dynamic boundaries vectors *GKLS_domain_left* and *GKLS_domain_right* by calling subroutine

void *GKLS_domain_free* (**void**);

again with no parameters.

It should be finally highlighted that if the user, after deallocating memory, wishes to return to the same class, generation of the class with the same parameters produces the same 100 test functions.

An example of the generation and use of some of the test classes can be found in the file **example.c**.

A. APPENDIX

Formulae of derivatives of the D- and D2-type test functions

In this section, analytical expressions of the partial derivatives of the D- and D2-type test functions are given. We denote by $T = (T_1, \dots, T_N)$ the minimizer of the paraboloid Z from (3) and by $M_i = (m_1^i, ..., m_N^i)$, $i = 2, ...m$, the local minima (from (4)) of a test function. Thus, for a D-type test function $f(x)$ given by (7)–(8) we have (see Gaviano and Lera [1998]):

$$\frac{\partial f(x)}{\partial x_j} = \begin{cases} \frac{\partial C_i(x)}{\partial x_j}, & x \in S_i,\ i \in \{2, \dots, m\}, \\ 2(x_j - T_j), & x \notin S_2 \cup \dots \cup S_m\,, \end{cases} \tag{A.1}$$

for $j = 1, \ldots, N$, and

$$\begin{aligned} \frac{\partial C_i(x)}{\partial x_j} &= \frac{2}{\rho_i^2} h_j(x)\|x - M_i\| + 3\left(\frac{2}{\rho_i^2}\frac{<x - M_i, T - M_i>}{\|x - M_i\|} - \frac{2}{\rho_i^3}A_i\right) \times \\ &\times (x_j - m_j^i)\|x - M_i\| - \frac{4}{\rho_i} h_j(x) + \\ &+ 2\left(1 - \frac{4}{\rho_i}\frac{<x - M_i, T - M_i>}{\|x - M_i\|} + \frac{3}{\rho_i^2}A_i\right)(x_j - m_j^i), \end{aligned} \tag{A.2}$$

with $h_j(x) = (T_j - m_j^i)\|x - M_i\| - <x - M_i, T - M_i> (x_j - m_j^i)/\|x - M_i\|$.

The first order partial derivatives of the D2-type test functions $f(x)$ given by (12)–(13) are calculated as follows (see Gaviano and Lera [1998]):

$$\frac{\partial f(x)}{\partial x_j} = \begin{cases} \frac{\partial Q_i(x)}{\partial x_j}, & x \in S_i,\ i \in \{2, \ldots, m\}, \\ 2(x_j - T_j), & x \notin S_2 \cup \ldots \cup S_m\,, \end{cases} \tag{A.3}$$

for $j = 1, \ldots, N$, and

$$\begin{aligned} \frac{\partial Q_i(x)}{\partial x_j} &= -\frac{6}{\rho_i^4} h_j(x)\|x - M_i\|^3 + 5(x_j - m_j^i)\|x - M_i\|^3 \times \\ &\times \left[-\frac{6}{\rho_i^4}\frac{<x - M_i, T - M_i>}{\|x - M_i\|} + \frac{6}{\rho_i^5}A_i + \frac{1}{\rho_i^3}(1 - \frac{\delta}{2})\right] + \\ &+ \frac{16}{\rho_i^3} h_j(x)\|x - M_i\|^2 + 4(x_j - m_j^i)\|x - M_i\|^2 \times \\ &\times \left[\frac{16}{\rho_i^3}\frac{<x - M_i, T - M_i>}{\|x - M_i\|} - \frac{15}{\rho_i^4}A_i - \frac{3}{\rho_i^2}(1 - \frac{\delta}{2})\right] - \\ &- \frac{12}{\rho_i^2} h_j(x)\|x - M_i\| + 3(x_j - m_j^i)\|x - M_i\| \times \\ &\times \left[-\frac{12}{\rho_i^2}\frac{<x - M_i, T - M_i>}{\|x - M_i\|} + \frac{10}{\rho_i^3}A_i + \frac{3}{\rho_i}(1 - \frac{\delta}{2})\right] + \\ &+ \delta(x_j - m_j^i), \end{aligned} \tag{A.4}$$

with $h_j(x) = (T_j - m_j^i)\|x - M_i\| - <x - M_i, T - M_i> (x_j - m_j^i)/\|x - M_i\|$.

Let us now consider the second order derivatives $\partial^2 f(x)/\partial x_j \partial x_k$ and $\partial^2 f(x)/\partial x_j^2$ of the D2-type test functions $f(x)$. For mixed partial derivatives $\partial^2 f(x)/\partial x_j \partial x_k$ we have

$$\frac{\partial^2 f(x)}{\partial x_j \partial x_k} = \begin{cases} \frac{\partial^2 Q_i(x)}{\partial x_j \partial x_k}, & x \in S_i,\ i \in \{2, \ldots, m\}, \\ 0, & x \notin S_2 \cup \ldots \cup S_m\,, \end{cases} \tag{A.5}$$

for $j, k = 1, \ldots, N,\ j \neq k$, and

$$\begin{aligned} \frac{\partial^2 Q_i(x)}{\partial x_j \partial x_k} &= -\frac{6}{\rho_i^4}\left[\frac{\partial h_j(x)}{\partial x_k}\|x - M_i\|^3 + 3h_j(x)(x_k - m_k^i)\|x - M_i\|\right] - \\ &- \frac{30}{\rho_i^4} h_k(x)(x_j - m_j^i)\|x - M_i\| + 15(x_j - m_j^i)(x_k - m_k^i)\|x - M_i\| \times \end{aligned}$$

$$
\begin{aligned}
&\times \left[-\frac{6}{\rho_i^4}\frac{<x-M_i, T-M_i>}{\|x-M_i\|} + \frac{6}{\rho_i^5}A_i + \frac{1}{\rho_i^3}(1-\frac{\delta}{2})\right] + \\
&+ \frac{16}{\rho_i^3}\left[\frac{\partial h_j(x)}{\partial x_k}\|x-M_i\|^2 + 2h_j(x)(x_k - m_k^i)\right] + \\
&+ \frac{64}{\rho_i^3}h_k(x)(x_j - m_j^i) + 8(x_j - m_j^i)(x_k - m_k^i) \times \\
&\times \left[\frac{16}{\rho_i^3}\frac{<x-M_i, T-M_i>}{\|x-M_i\|} - \frac{15}{\rho_i^4}A_i - \frac{3}{\rho_i^2}(1-\frac{\delta}{2})\right] - \\
&- \frac{12}{\rho_i^2}\left[\frac{\partial h_j(x)}{\partial x_k}\|x-M_i\| + h_j(x)\frac{(x_k - m_k^i)}{\|x-M_i\|}\right] - \\
&- \frac{36}{\rho_i^2}h_k(x)\frac{(x_j - m_j^i)}{\|x-M_i\|} + 3(x_j - m_j^i)\frac{(x_k - m_k^i)}{\|x-M_i\|} \times \\
&\times \left[-\frac{12}{\rho_i^2}\frac{<x-M_i, T-M_i>}{\|x-M_i\|} + \frac{10}{\rho_i^3}A_i + \frac{3}{\rho_i}(1-\frac{\delta}{2})\right],
\end{aligned}
\tag{A.6}
$$

with

$$\frac{\partial h_j(x)}{\partial x_k} = (T_j - m_j^i)\frac{(x_k - m_k^i)}{\|x-M_i\|} - \frac{h_k(x)}{\|x-M_i\|^2}(x_j - m_j^i),$$

and

$$h_k(x) = (T_k - m_k^i)\|x-M_i\| - <x-M_i, T-M_i> \frac{(x_k - m_k^i)}{\|x-M_i\|},$$

while for pure partial derivatives $\partial^2 f(x)/\partial x_j^2$ we have

$$\frac{\partial^2 f(x)}{\partial x_j^2} = \begin{cases} \frac{\partial^2 Q_i(x)}{\partial x_j^2}, & x \in S_i,\ i \in \{2, \dots, m\}, \\ 2, & x \notin S_2 \cup \dots \cup S_m\,, \end{cases} \tag{A.7}$$

for $j = 1, \dots, N$, and

$$
\begin{aligned}
\frac{\partial^2 Q_i(x)}{\partial x_j^2} \;=\; &-\frac{6}{\rho_i^4}\left[\frac{\partial h_j(x)}{\partial x_j}\|x-M_i\|^3 + 3h_j(x)(x_j - m_j^i)\|x-M_i\|\right] + \\
&+ \left[5\|x-M_i\|^3 + 15(x_j - m_j^i)^2\|x-M_i\|\right] \times \\
&\times \left[-\frac{6}{\rho_i^4}\frac{<x-M_i, T-M_i>}{\|x-M_i\|} + \frac{6}{\rho_i^5}A_i + \frac{1}{\rho_i^3}(1-\frac{\delta}{2})\right] - \\
&- \frac{30}{\rho_i^4}h_j(x)(x_j - m_j^i)\|x-M_i\| + \\
&+ \frac{16}{\rho_i^3}\left[\frac{\partial h_j(x)}{\partial x_j}\|x-M_i\|^2 + 2h_j(x)(x_j - m_j^i)\right] + \\
&+ \frac{64}{\rho_i^3}h_j(x)(x_j - m_j^i) + \left[4\|x-M_i\|^2 + 8(x_j - m_j^i)^2\right] \times \\
&\times \left[\frac{16}{\rho_i^3}\frac{<x-M_i, T-M_i>}{\|x-M_i\|} - \frac{15}{\rho_i^4}A_i - \frac{3}{\rho_i^2}(1-\frac{\delta}{2})\right] -
\end{aligned}
\tag{A.8}
$$

$$
\begin{aligned}
&- \frac{12}{\rho_i^2}\left[\frac{\partial h_j(x)}{\partial x_j}\|x-M_i\| + h_j(x)\frac{(x_j-m_j^i)}{\|x-M_i\|}\right] - \\
&- \frac{36}{\rho_i^2} h_j(x)\frac{(x_j-m_j^i)}{\|x-M_i\|} + \left[3\|x-M_i\| + 3\frac{(x_j-m_j^i)^2}{\|x-M_i\|}\right] \times \\
&\times \left[-\frac{12}{\rho_i^2}\frac{<x-M_i, T-M_i>}{\|x-M_i\|} + \frac{10}{\rho_i^3}A_i + \frac{3}{\rho_i}(1-\frac{\delta}{2})\right] + \delta,
\end{aligned}
$$

with

$$
\frac{\partial h_j(x)}{\partial x_j} = (T_j - m_j^i)\frac{(x_j-m_j^i)}{\|x-M_i\|} - \frac{h_j(x)}{\|x-M_i\|^2}(x_j-m_j^i) - \frac{<x-M_i, T-M_i>}{\|x-M_i\|}.
$$

ACKNOWLEDGMENTS

The authors thank Associate Editor Michael Saunders and an anonymous referee for their subtle suggestions.

REFERENCES

Ali, M. M., Khompatraporn, C., and Zabinsky, Z. B. 2003. A numerical evaluation of several global optimization algorithms on selected benchmark test problems. *Submitted*.

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

Facchinei, F., Júdice, J., and Soares, J. 1997. Generating box-constrained optimization problems. *ACM Trans. Math. Soft. 23,* 3 (Sept.), 443–447.

Floudas, C. A. and Pardalos, P. M. 1990. A collection of test problems for constrained global optimization algorithms. In *Lecture Notes in Computer Science*, G. Goos and J. Hartmanis, Eds. Vol. 455. Springer Verlag, Berlin–New York.

Floudas, C. A., Pardalos, P. M., Adjiman, C., Esposito, W., Gümüs, Z., Harding, S., Klepeis, J., Meyer, C., and Schweiger, C. 1999. *Handbook of Test Problems in Local and Global Optimization*. Kluwer Academic Publishers, Dordrecht.

Gaviano, M. and Lera, D. 1998. Test functions with variable attraction regions for global optimization problems. *J. Global Optimizat. 13,* 2 (Sept.), 207–223.

Horst, R. and Pardalos, P. M., Eds. 1995. *Handbook of Global Optimization*. Kluwer Academic Publishers, Dordrecht.

Kalantari, B. and Rosen, J. B. 1986. Construction of large-scale global minimum concave quadratic test problems. *J. Optim. Theory Appl. 48,* 2, 303–313.

Khoury, B. N., Pardalos, P. M., and Du, D.-Z. 1993. A test problem gnerator for the Steiner problem in graphs. *ACM Trans. Math. Soft. 19,* 4 (Dec.), 509–522.

Knuth, D. 1997. *The Art of Computer Programming, Vol. 2: Seminumerical Algorithms*, third ed. Addison-Wesley, Reading, Massachusetts.

Knuth, D. 2002. Home page at: http://sunburn.stanford.edu/~knuth/.

Li, Y. and Pardalos, P. M. 1992. Generating quadratic assignement test problems with known optimal permutations. *Comp. Optim. Appl. 1,* 2, 163–184.

Locatelli, M. 2003. A note on the Griewank test function. *J. Global Optimizat. 25,* 2 (Feb.), 169–174.

Moré, J., Garbow, B., and Hillstrom, K. 1981. Testing unconstrained optimization software. *ACM Trans. Math. Soft. 7,* 1 (Mar.), 17–41.

Moshirvaziri, K. 1994. Construction of test problems for a class of reverse convex programs. *J. Optim. Theory Appl. 81,* 2, 343–354.

MOSHIRVAZIRI, K., AMOUZEGAR, M. A., AND JACOBSEN, S. E. 1996. Test problem construction for linear bilevel programming problem. *Special Issue: Hierarchical and Bilevel Programming, J. Global Optimizat. 8,* 3 (Apr.), 235–244.

PARDALOS, P. M. 1987. Generation of large-scale quadratic programs for use as global optimization test problems. *ACM Trans. Math. Soft. 13,* 2 (June), 133–137.

PARDALOS, P. M. 1991. Construction of test problems in quadratic bivalent programming. *ACM Trans. Math. Soft. 17,* 1 (Mar.), 74–87.

PINTÉR, J. 2002. Global optimization: Software, test problems, and applications. In *Handbook of Global Optimization*, P. M. Pardalos and H. E. Romeijn, Eds. Vol. 2. Kluwer Academic Publishers, Dordrecht, 515–569.

SCHITTKOWSKI, K. 1980. *Nonlinear Programming Codes.* Springer Verlag, Berlin–New York.

SCHITTKOWSKI, K. 1987. *More Test Examples for Nonlinear Programming Codes.* Springer Verlag, Berlin–New York.

SCHOEN, F. 1993. A wide class of test functions for global optimization. *J. Global Optimizat. 3*, 133–137.

SUNG, Y. Y. AND ROSEN, J. B. 1982. Global minimum test problem construction. *Math. Progr. 24*, 353–355.