Benchmark Functions for CEC'2013 Special Session and Competition on Niching Methods for Multimodal Function Optimization

Xiaodong Li, Andries Engelbrecht, Michael G. Epitropakis

Evolutionary Algorithms (EAs) in their original forms are usually designed for locating a single global solution. These algorithms typically converge to a single solution because of the global selection scheme used. Nevertheless, many real-world problems are "multimodal" by nature, i.e., multiple satisfactory solutions exist. It may be desirable to locate many such satisfactory solutions so that a decision maker can choose one that is most proper in his/her problem domain. Numerous techniques have been developed in the past for locating multiple optima (global or local). These techniques are commonly referred to as "niching" methods. A niching method can be incorporated into a standard EA to promote and maintain formation of multiple stable subpopulations within a single population, with an aim to locate multiple globally optimal or suboptimal solutions. Many niching methods have been developed in the past, including crowding [1], fitness sharing [2], deterministic crowding [3], derating [4], restricted tournament selection [5], parallelization [6], stretching and deflation [7], clustering [8], clearing [9], and speciation [10], etc.

Although these niching methods have been around for many years, further advances in this area have been hindered by several obstacles: most studies focus on very low dimensional multimodal problems (2 or 3 dimensions), therefore it is difficult to assess these methods' scalability to high dimensions; some niching methods introduces new parameters which are difficult to set, making these methods difficult to use; different benchmark test functions or different variants of the same functions are used, hence comparing the performance of different niching methods is difficult.

We believe it is now time to adopt a unifying framework for evaluating niching methods, so that further advance in this area can be made with ease. In this technical report, we put together 20 benchmark test functions (including several identical functions with different dimension sizes), with different characteristics, for evaluating niching algorithms. The first 10 benchmark functions are simple, well known and widely used functions, largely based on some recent studies on niching [11], [12], [13]. The remaining benchmark functions are more complex and follow the paradigm of composition functions defined in the IEEE CEC 2005 special session on real-parameter optimization [14].

Several of the test functions included here are scalable to dimension, and the number of global optima can be adjusted freely by the user. Performance measures are also defined and suggested here for comparing different niching methods. The source codes of the benchmark test functions are made available in Matlab, Java and C++ source codes. The competition files can be downloaded from the CEC'2013 special session on niching methods website[1].

In the following sections, we will describe the mathematical formula and properties of the included multimodal benchmark test functions, evaluation criteria, and the ranking method for entries submitted to this competition.

I. Summary of test functions

This benchmark set includes the following 20 multimodal test functions:

- F_1: Five-Uneven-Peak Trap (1D)
- F_2: Equal Maxima (1D)
- F_3: Uneven Decreasing Maxima (1D)
- F_4: Himmelblau (2D)
- F_5: Six-Hump Camel Back (2D)
- F_6: Shubert (2D, 3D)
- F_7: Vincent (2D, 3D)
- F_8: Modified Rastrigin - All Global Optima (2D)
- F_9: Composition Function 1 (2D)
- F_{10}: Composition Function 2 (2D)
- F_{11}: Composition Function 3 (2D, 3D, 5D, 10D)
- F_{12}: Composition Function 4 (3D, 5D, 10D, 20D)

These multimodal test functions have following properties:

- All test functions are formulated as maximization problems;
- F_1, F_2 and F_3 are simple 1D multimodal functions;
- F_4 and F_5 are simple 2D multimodal functions. These functions are not scalable;
- F_6 to F_8 are scalable multimodal functions. The number of global optima for F_6 and F_7 are determined by the dimension D. However, for F_8, the number of global optima is independent from D, therefore can be controlled by the user.
- F_9 to F_{12} are scalable multimodal functions constructed by several basic functions with different properties. F_9

Xiaodong Li is with the School of Computer Science and IT, RMIT University, VIC 3001, Melbourne, Australia, email: xiaodong.li@rmit.edu.au

Andries Engelbrecht is with the Department of Computer Science, School of Information Technology, University of Pretoria, Pretoria 0002, South Africa, email: engel@cs.up.ac.za

Michael G. Epitropakis is with the Department of Computing Science and Mathematics, School of Natural Sciences, University of Stirling, Stirling FK9 4LA, Scotland, email: mge@cs.stir.ac.uk

[1]URL: http://goanna.cs.rmit.edu.au/~xiaodong/cec13-niching/

and F_{10} are separable, and non-symmetric, while F_{11} and F_{12} are non-separable, non-symmetric complex multimodal functions. The number of global optima in all composition functions is independent from D, therefore can be controlled by the user.

II. Definitions of the Test Functions

A. F_1: Five-Uneven-Peak Trap

$$F_1(x) = \begin{cases} 80(2.5 - x) & \text{for } 0 \leq \text{x} < 2.5, \\ 64(x - 2.5) & \text{for } 2.5 \leq \text{x} < 5.0, \\ 64(7.5 - x) & \text{for } 5.0 \leq \text{x} < 7.5, \\ 28(x - 7.5) & \text{for } 7.5 \leq \text{x} < 12.5, \\ 28(17.5 - x) & \text{for } 12.5 \leq \text{x} < 17.5, \\ 32(x - 17.5) & \text{for } 17.5 \leq \text{x} < 22.5, \\ 32(27.5 - x) & \text{for } 22.5 \leq \text{x} < 27.5, \\ 80(x - 27.5) & \text{for } 27.5 \leq \text{x} \leq 30. \end{cases}$$

Properties:

- Variable ranges: $x \in [0, 30]$;
- No. of global optima: 2;
- No. of local optima:3.

This function was originally proposed in [10]. See Fig. 1.

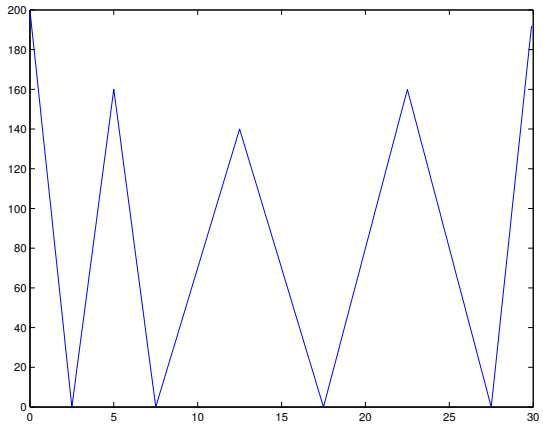

Fig. 1. Five-Uneven-Peak Trap.

B. F_2: Equal Maxima

$$F_2(x) = sin^6(5\pi x).$$

Properties:

- Variable ranges: $x \in [0, 1]$;
- No. of global optima: 5;
- No. of local optima: 0;

This function was originally proposed in [15]. See Fig. 2.

C. F_3: Uneven Decreasing Maxima

$$F_3(x) = exp\left(-2log(2)\left(\frac{x - 0.08}{0.854}\right)^2\right) sin^6(5\pi(x^{3/4} - 0.05)).$$

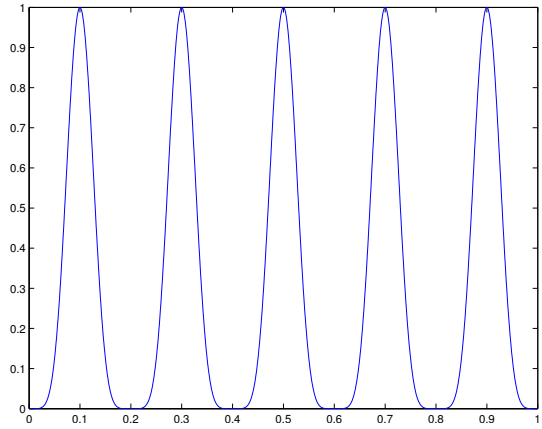

Fig. 2. Equal Maxima.

Properties:

- Variable ranges: $x \in [0, 1]$;
- No. of global optima: 1;
- No. of local optima: 4;

This function was originally proposed in [15]. See Fig. 3.

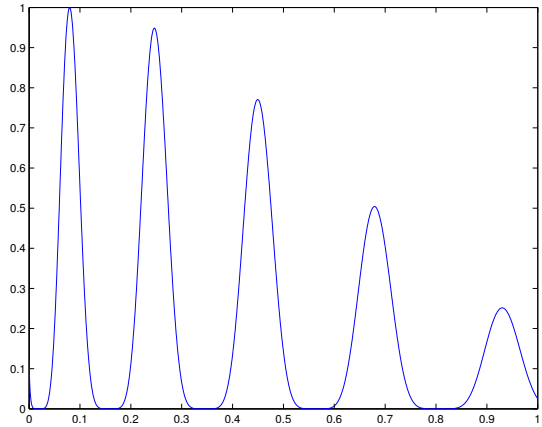

Fig. 3. Uneven Decreasing Maxima.

D. F_4: Himmelblau

$$F_4(x, y) = 200 - (x^2 + y - 11)^2 - (x + y^2 - 7)^2.$$

Properties:

- Variable ranges: $x, y \in [-6, 6]$;
- No. of global optima: 4;
- No. of local optima: 0;

This is an inverted version of Himmelblau function [15]. It has 4 global optima with 2 closer to each other than the other 2. There are no local optima, as shown in Fig. 4.

E. F_5: Six-Hump Camel Back

$$F_5(x, y) = -4[(4 - 2.1x^2 + \frac{x^4}{3})x^2 + xy + (4y^2 - 4)y^2].$$

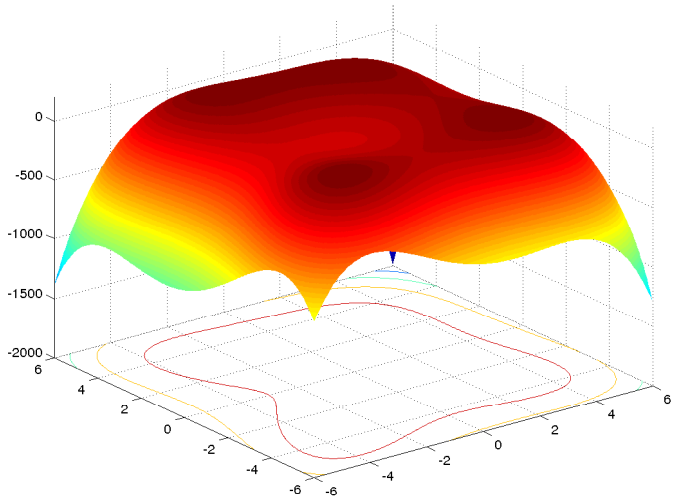

Fig. 4. Himmelblau.

Properties:

- Variable ranges: $x \in [-1.9, 1.9]$; $y \in [-1.1, 1.1]$;
- No. of global optima: 2;
- No. of local optima: 2;

This function was originally proposed in [16]. The function has 2 global optima as well as 2 local optima. See Fig. 5.

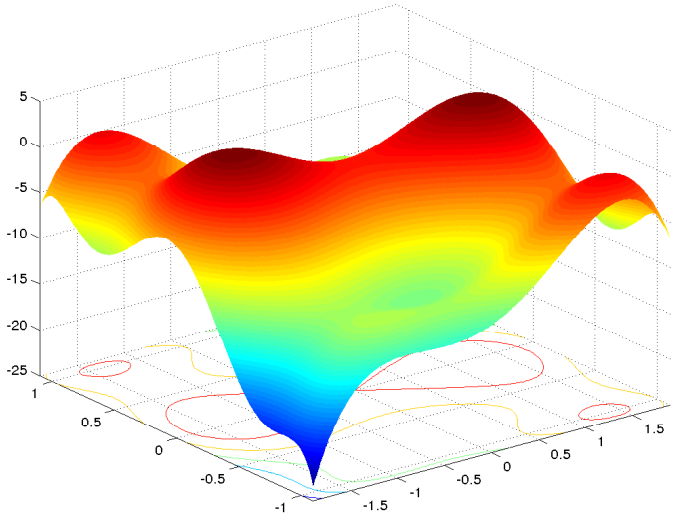

Fig. 5. Six-Hump Camel Back.

F. F_6: Shubert

$$F_6(\vec{\mathbf{x}}) = -\prod_{i=1}^{D}\sum_{j=1}^{5} jcos[(j+1)x_i + j].$$

Properties:

- Variable ranges: $x_i \in [-10, 10]^D, i = 1, 2, \ldots, D$;
- No. of global optima: $D \cdot 3^D$;
- No. of local optima: many;

This function is an inverted version of the Shubert function [16], [10], where there are $n3^D$ global optima unevenly distributed. These global optima are divided into 3^D groups, with each group having D global optima being close to each other. Fig. 6 shows an example of the Shubert 2D function, where there are 18 global optima in 9 pairs (each with an objective function value of 186.73), with each pair very close to each other, but the distance between any pair is much greater. There are in total 760 global and local optima.

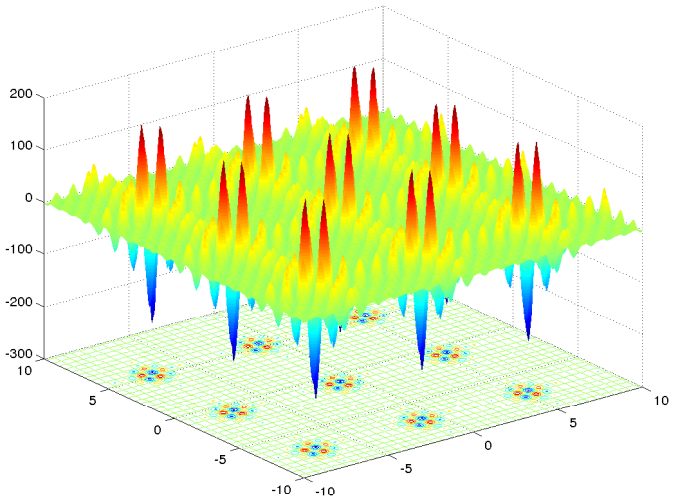

Fig. 6. Shubert 2D function.

G. F_7: Vincent

$$F_7(\vec{\mathbf{x}}) = \frac{1}{D}\sum_{i=1}^{D} sin(10log(x_i))$$

Properties:

- Variable range: $x_i \in [0.25, 10]^D, i = 1, 2, \ldots, D$
- No. of global optima: 6^D;
- No. of local optima: 0;

This is an inverted version of the Vincent function [17], which has 6^D global optima (each with an objective function value of 1.0), but unlike the regular distances between global optima in F_6, in Vincent function the global optima have vastly different spacing between them. In addition, the Vincent function has no local optima. Fig. 7 shows an example of the Vincent 2D function.

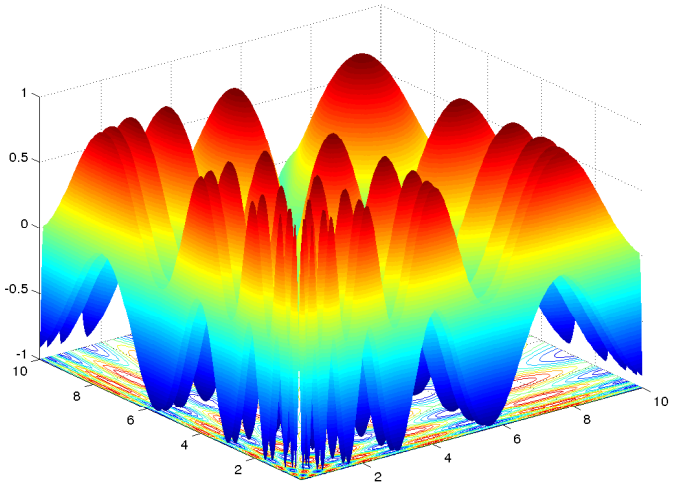

Fig. 7. Vincent 2D function.

H. F_8: Modified Rastrigin - All Global Optima

$$F_9(\vec{\mathbf{x}}) = -\sum_{i=1}^{D}(10 + 9cos(2\pi k_i x_i)).$$

Properties:

- Variable ranges: $x_i \in [0,1]^D, i = 1, 2, \ldots, D$;
- No. of global optima: $\prod_{i=1}^{D} k_i$;
- No. of local optima: 0;

This is a modified Rastrigin function according to [13]. The total number of global optima is $M = \prod_{i=1}^{D} k_i$. In [13], a problem instance $D = 16$ was used, with the following setting: $k_i = 1$, for $i = 1 - 3, 5 - 7, 9 - 11, 13 - 15$, and $k_4 = 2, k_8 = 2, k_{12} = 3, k_{16} = 4$. In this case, there are 48 evenly distributed global optima, each having an identical objective value of $f = 16$. Fig. 8 provides a problem instance of this function in 2D (where $D = 2, k_1 = 3, k_2 = 4$), in which case the total number of optima is 12.

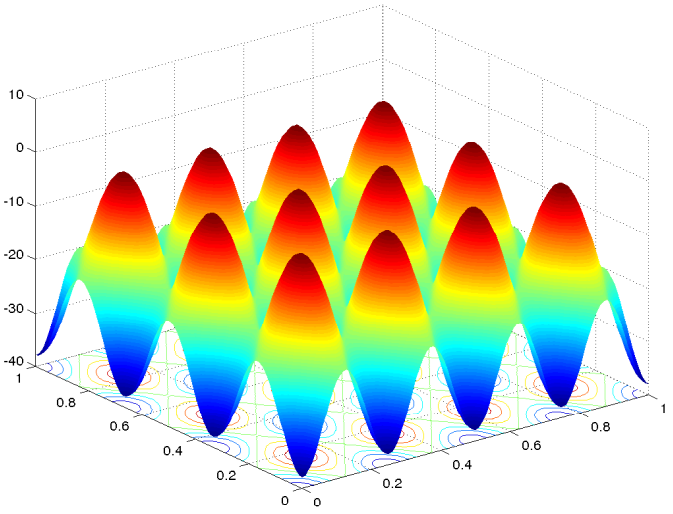

Fig. 8. Modified Rastrigin - All Global Optima 2D function.

I. Composition functions

In this section, we first describe the general framework for constructing multimodal composition functions with several global optima, then present the new composition functions used in this technical report.

More specifically, a D-dimensional, composition function $\mathrm{CF}_j : \mathcal{A}_D \subset \mathbb{R}^D \rightarrow \mathbb{R}$ can be generally constructed as a weighted aggregation of n basic functions $f_i : \mathcal{A}_D \subset \mathbb{R}^D \rightarrow \mathbb{R}$. Each basic function is shifted to a new position inside the optimization space $\mathcal{A}_D$ and can be either rotated through a linear transformation matrix or used as is. Thus, a composition function CF_j is calculated according the following equation:

$$\mathrm{CF}_j(\vec{x}) = \sum_{i=1}^{n} w_i \left(\hat{f}_i \left((\vec{x} - \vec{o_i})/\lambda_i \cdot M_i \right) + \mathrm{bias}_i \right) + f^j_{\mathrm{bias}},$$

where n is the number of basic functions used to construct the composition function, $\hat{f}_i$ denotes a normalization of the i-th basic function, $i \in \{1, 2, \ldots, n\}$, w_i is the corresponding weight, $\vec{o_i}$ is the new shifted optimum of each $\hat{f}_i$, M_i is the linear transformation (rotation) matrix of each $\hat{f}_i$, and λ_i is a parameter which is used to stretch ($\lambda_i > 1$) or compress ($\lambda_i < 1$) each $\hat{f}_i$ function. The composition function includes two bias parameters bias_i and f^j_{bias}. The former defines a function value bias for each basic function and denotes which optimum is the global optimum, while the latter defines a function value bias for the constructed composition function. Here, we set the $\mathrm{bias}_i = 0, \forall i \in \{1, 2, \ldots, n\}$, thus the global optimum of each basic function is a global optimum of the composition function. In addition, we set $f^j_{\mathrm{bias}} = 0$, as such in each composition function, all global optima have fitness values equal to zero.

The weight w_i of each basic function can be easily calculated based on the following equations:

$$w_i = \exp\left(-\frac{\sum_{k=1}^{D}(x_k - o_{ik})^2}{2D\sigma_i^2} \right),$$

$$w_i = \begin{cases} w_i & w_i = \max(w_i) \\ w_i(1 - \max(w_i)^{10}) & \text{otherwise.} \end{cases}$$

Finally, the weights are normalized according to $w_i = w_i / \sum_{i=1}^{n} w_i$. The parameter σ_i controls the coverage range of each basic function, with small values to produce a narrow coverage range to the corresponding $\hat{f}_i$.

The pool of basic functions may include functions with different properties, characteristics and heights. As such to have a better mixture of the basic functions a normalization procedure is incorporated. The normalized function $\hat{f}_i$, can be defined as: $\hat{f}_i(\cdot) = C f_i(\cdot)/|f^i_{\max}|$, where C is a predefined constant ($C = 2000$) and $f^i_{\max}$ is estimated using: $f^i_{\max} = f_i\left((x^\star/\lambda_i)\, M_i\right)$, with $x^\star = [5, 5, \ldots, 5]$.

The pool of basic functions that we have used to construct the composition functions includes the following:

- Sphere function:

$$f_S(\vec{x}) = \sum_{i=1}^{D} x_i^2.$$

- Grienwank's function:

$$f_G(\vec{x}) = \sum_{i=1}^{D} \frac{x_i^2}{4000} - \prod_{i=1}^{D} \cos\left(\frac{x_i}{\sqrt{i}}\right) + 1.$$

- Rastrigin's function:

$$f_R(\vec{x}) = \sum_{i=1}^{D} \left(x_i^2 - 10\cos(2\pi x_i) + 10\right).$$

- Weierstrass function:

$$f_W(\vec{x}) = \sum_{i=1}^{D} \left(\sum_{k=0}^{\mathrm{kmax}} \alpha^k \cos\left(2\pi\beta^k(x_i + 0.5)\right) \right) - D\sum_{k=0}^{\mathrm{kmax}} \alpha^k \cos\left(2\pi\beta^k(0.5)\right),$$

where $\alpha = 0.5, \beta = 3$, and $\mathrm{kmax} = 20$.

– Expanded Griewank's plus Rosenbrock's function (EF8F2):

$$F8(\vec{x}) = \sum_{i=1}^{D} \frac{x_i^2}{4000} - \prod_{i=1}^{D} \cos\left(\frac{x_i}{\sqrt{i}}\right) + 1,$$
$$F2(\vec{x}) = \sum_{i=1}^{D-1} \left(100(x_i^2 - x_{i+1})^2 + (x_i - 1)^2\right),$$
$$EF8F2(\vec{x}) = F8F2(x_1, x_2, \ldots, x_D)$$
$$= F8(F2(x_1, x_2)) + F8(F2(x_2, x_3)) + \ldots$$
$$+ F8(F2(x_{D-1}, x_D)) + F8(F2(x_D, x_1))$$

It is clear that the aforementioned basic functions do not incorporate either shifted positions, or linear transformations (rotations). Thus, in order to calculate for example $f_S\left((\vec{x} - \vec{o_i})/\lambda_i \cdot M_i\right)$, one can easily first calculate $\vec{z} = (\vec{x} - \vec{o_i})/\lambda_i \cdot M_i$ and subsequently $f_S(\vec{z})$. It has to be noted that all composition functions are formulated as maximization problems.

Note on the implementation: The implementation of the composition functions has been mostly based on the source code of the IEEE CEC 2005 competition on real-parameter optimization [14]. However, the source code has been re-implemented from scratch in a simplified and scalable way. It has to be noted that, similar implementation of composition functions with multiple global optima was proposed in [18]. However, our implementation of the composition functions are different in many ways: new structures, parameters, rotation matrices, and shifted optima positions.

J. F_9 : *Composition Function 1*

Composition Function 1 (CF_1) is constructed based on six basic functions ($n = 6$), thus it has six global optima in the optimization box $\mathcal{A}_D = [-5, 5]^D$. The basic functions used here include the following:

– $f_1 - f_2$: Grienwank's function,
– $f_3 - f_4$: Weierstrass function, and
– $f_5 - f_6$: Sphere function.

The composition function is constructed based on the following parameter settings:

– $\sigma_i = 1, \forall i \in \{1, 2, \ldots, n\}$,
– $\vec{\lambda} = [1, 1, 8, 8, 1/5, 1/5]$,
– M_i are identity matrices $\forall i \in \{1, 2, \ldots, n\}$.

Fig. 9 shows the 2D version of CF_1.

Properties:

– Multi-modal,
– Shifted,
– Non-Rotated,
– Non-symmetric,
– Separable near the global optima,
– Scalable,
– Numerous local optima,
– Different function's properties are mixed together,
– Sphere Functions give two flat areas for the function,
– In the optimization box $\mathcal{A}_D = [-5, 5]^D$, there are six global optima $\vec{x_i^\star} = \vec{o_i}, i \in \{1, 2, \ldots, n\}$ with $CF_1(\vec{x_i^\star}) = 0, \forall i \in \{1, 2, \ldots, n\}$.

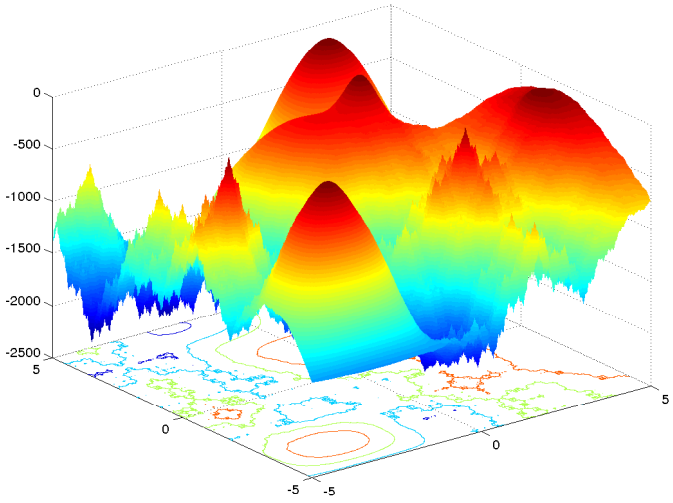

Fig. 9. Composition Function 1.

K. F_{10} : *Composition Function 2*

Composition Function 2 (CF_2) is constructed based on eight basic functions ($n = 8$), thus it has eight global optima in the optimization box $\mathcal{A}_D = [-5, 5]^D$. The basic functions used here include the following:

– $f_1 - f_2$: Rastrigin's function,
– $f_3 - f_4$: Weierstrass function,
– $f_5 - f_6$: Griewank's function, and
– $f_7 - f_8$: Sphere function.

The composition function is constructed based on the following parameter settings:

– $\sigma_i = 1, \forall i \in \{1, 2, \ldots, n\}$,
– $\vec{\lambda} = [1, 1, 10, 10, 1/10, 1/10, 1/7, 1/7]$,
– M_i are identity matrices $\forall i \in \{1, 2, \ldots, n\}$.

Fig. 10 shows the 2D version of CF_2.

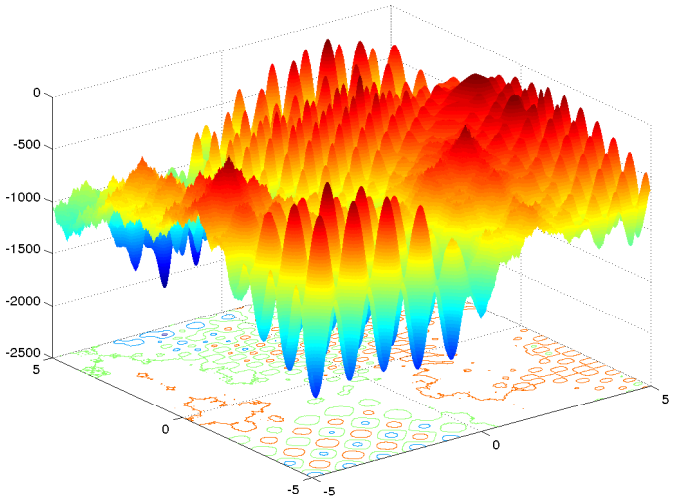

Fig. 10. Composition Function 2.

Properties:

– Multi-modal,
– Shifted,

- Non-Rotated,
- Non-symmetric,
- Separable near the global optima,
- Scalable,
- Numerous local optima,
- Different function's properties are mixed together,
- In the optimization box $\mathcal{A}_D = [-5,5]^D$, there are eight global optima $\vec{x_i^\star} = \vec{o_i}, i \in \{1,2,\ldots,n\}$ with $\text{CF}_2(\vec{x_i^\star}) = 0, \forall i \in \{1,2,\ldots,n\}$.

L. F_{11} : *Composition Function 3*

Composition Function 3 (CF_3) is constructed based on six basic functions ($n = 6$), thus it has six global optima in the optimization box $\mathcal{A}_D = [-5,5]^D$. The basic functions used here include the following:

- $f_1 - f_2$: EF8F2 function,
- $f_3 - f_4$: Weierstrass function, and
- $f_5 - f_6$: Griewank's function.

The composition function is constructed based on the following parameter settings:

- $\vec{\sigma} = [1,1,2,2,2,2]$,
- $\vec{\lambda} = [1/4, 1/10, 2, 1, 2, 5]$,
- M_i are different linear transformation (rotation) matrices with condition number one.

Fig. 11 shows the 2D version of the CF_3.

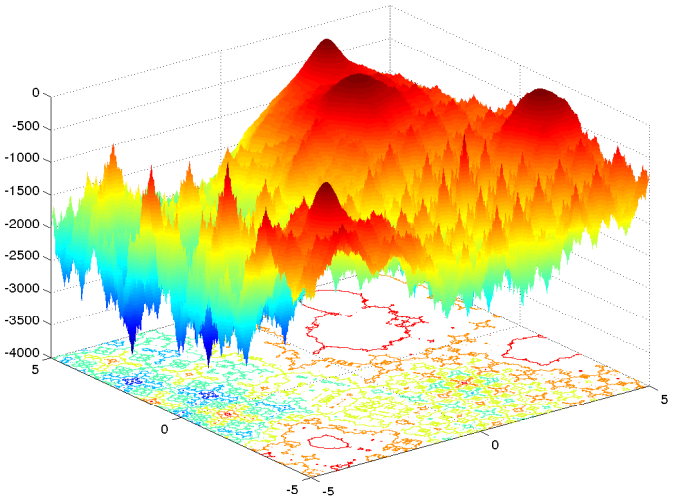

Fig. 11. Composition Function 3.

Properties:

- Multi-modal,
- Shifted,
- Rotated,
- Non-symmetric,
- Non-separable,
- Scalable,
- A huge number of local optima,
- Different function's properties are mixed together,
- In the optimization box $\mathcal{A}_D = [-5,5]^D$, there are six global optima $\vec{x_i^\star} = \vec{o_i}, i \in \{1,2,\ldots,n\}$ with $\text{CF}_3(\vec{x_i^\star}) = 0, \forall i \in \{1,2,\ldots,n\}$.

M. F_{12} : *Composition Function 4*

Composition Function 4 (CF_4) is constructed based on eight basic functions ($n = 8$), thus it has eight global optima in the optimization box $\mathcal{A}_D = [-5,5]^D$. The basic functions used here include the following:

- $f_1 - f_2$: Rastrigin's function,
- $f_3 - f_4$: EF8F2 function,
- $f_5 - f_6$: Weierstrass function, and
- $f_7 - f_8$: Griewank's function.

The composition function is constructed based on the following parameter settings:

- $\vec{\sigma} = [1,1,1,1,1,2,2,2]$,
- $\vec{\lambda} = [4,1,4,1,1/10,1/5,1/10,1/40]$,
- M_i are different linear transformation (rotation) matrices with condition number one.

Fig. 12 shows the 2D version of the CF_4.

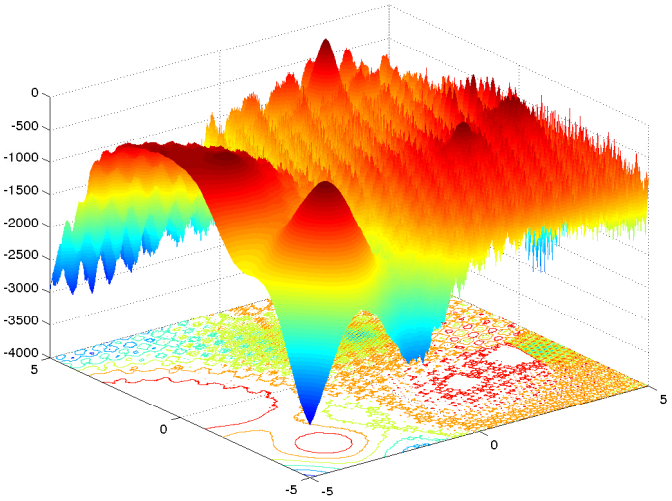

Fig. 12. Composition Function 4.

Properties:

- Multi-modal,
- Shifted,
- Rotated,
- Non-symmetric,
- Non-separable,
- Scalable,
- A huge number of local optima,
- Different function's properties are mixed together,
- In the optimization box $\mathcal{A}_D = [-5,5]^D$, there are eight global optima $\vec{x_i^\star} = \vec{o_i}, i \in \{1,2,\ldots,n\}$ with $\text{CF}_4(\vec{x_i^\star}) = 0, \forall i \in \{1,2,\ldots,n\}$.

III. Performance Measures

A. *How to determine if all global optima are found?*

Our objective is to compare different niching algorithms' capability to locate all global optima. To achieve this, first we need to specify a *level of accuracy* (typically $0 < \epsilon \leq 1$), a threshold value under which we would consider a global optimum is found. Second, we assume that for each test function, the following information is available:

- The number of global optima;
- The fitness of the global optima (or peak height), which is known or can be estimated;
- A niche radius value that can sufficiently distinguish two closest global optima.

This information is required by Algorithm 1 (based on a previously proposed procedure [19]), to determine if a niching algorithm has located all the global optima (Table IV provides an example of how this information is used for performance measures in this competition). Basically, at the end of an optimization run, Algorithm 1 is invoked to check all the individuals on the sorted list L_{sorted}, starting from the best-fit individual. In the first iteration, as long as this best-fit individual is within ϵ distance from the fitness of the global optima ph, it will be added to the solution list S (which is initially empty). In the next iteration, the next best-fit individual from L_{sorted} is first assigned to p. We then check if the fitness of p is close enough to that of the global optima (i.e., if $d(ph, fit(p)) \leq \epsilon$). If p is close enough (in other words, p is a potential solution), then next we check if p is within the niche radius r from all the current solutions in the solution list S. If it is not, then p is considered as a new solution (i.e., a new global optimum is found), and will be added to S. Otherwise p is not considered as a distinct solution, and will be skipped. This process is repeated until all the individuals on L_{sorted} have been checked. It is important to note that **Algorithm 1 is only used for performance measurement** in determining if a sufficient number of global optima has been found, but not in any part of the optimization procedure.

The output of Algorithm 1 is S, a solution list containing all the *distinct* global optima found. As long as r is set to a value not greater than the distance between 2 closest global optima, individuals on two found global optima would be treated as two different solutions. Since the exact number of global optima is known *a priori*, we can measure a niching algorithm's performance in terms of the *peak ratio*, *success rate*, and *averaged number of evaluations* required to achieve a given accuracy ϵ for locating all global optima over multiple runs. These will be described in the following sections.

input : L_{sorted} - a list of individuals (candidate solutions) sorted in decreasing fitness values;
ϵ - accuracy level; r - niche radius;
ph - the fitness of global optima (or peak height)
output: S - a list of best-fit individuals identified as solutions

```
begin
    S = ∅;
    while not reaching the end of L_sorted do
        Get best unprocessed p ∈ L_sorted;
        found ← FALSE;
        if d(ph, fit(p)) ≤ ϵ) then
            for all s ∈ S do
                if d(s, p) ≤ r then
                    found ← TRUE;
                    break;
                end
            end
            if not found then
                let S ← S ∪ {p};
            end
        end
    end
end
```

Algorithm 1: The algorithm for determining if all global optima are found.

B. Peak ratio and success rate

We use *peak ratio* (PR) [20] and *success rate* (SR) as two performance measures, to evaluate the performance of a niching algorithm over multiple runs. Given a fixed maximum number of function evaluations (MaxFEs) and a required *accuracy level* ϵ, PR measures the average percentage of all known global optima found over multiple runs:

$$PR = \frac{\sum_{run=1}^{NR} NPF_i}{NKP * NR}, \tag{1}$$

where NPF_i denotes the number of global optima found in the end of the i-th run, NKP the number of known global optima, and NR the number of runs. SR measures the percentage of successful runs (a successful run is defined as a run where all known global optima are found) out of all runs:

$$SR = \frac{NSR}{NR}, \tag{2}$$

where NSR denotes the number of successful runs.

C. Convergence speed

We measure the convergence speed of a niching algorithm by counting the number of function evaluations (FEs) required to locate all known global optima, at a specified accuracy level ϵ. We calculate the *average FEs* over multiple runs:

$$AveFEs = \frac{\sum_{run=1}^{NR} FE_i}{NR}, \tag{3}$$

where FE_i denotes the number of evaluations used in the i-th run. If the algorithm cannot locate all the global optima by the MaxFEs, then MaxFEs is used when calculating the average FEs.

IV. Experimental Settings

We evaluate niching methods by using a fixed amount of MaxFEs. A user is allowed to use any population size, but only a fixed amount of MaxFEs is allowed to be used as a given computational budget.

A. Peak ratio and success rate

PR and SR are calculated according to equations (1) and (2). The following parameter settings are used:

- Level of accuracy (ϵ): {1.0E-01,1.0E-02, . . . , 1.0E-05};
- Niche radius (r): See Table IV;

- Initialization: Uniform random initialization within the search space;
- Termination of a run: Terminate when reaching MaxFEs.
- Number of runs: 50.

Note that ϵ and r are to be used only at the end of a run, for evaluations of the final solutions. Table I shows different MaxFEs used for the 3 ranges of test functions.

TABLE I
MAXFES USED FOR 3 RANGES OF TEST FUNCTIONS.

Range of functions	MaxFEs
F_1 to F_5 (1D or 2D)	5.0E+04
F_6 to F_{11} (2D)	2.0E+05
F_6 to F_{12} (3D or higher)	4.0E+05

For F_8 (2D), we set $k_1 = 3$ and $k_2 = 4$, as shown in Fig. 8. Table II shows an example of presenting PR and SR values of a typical niching algorithm. Note that in this example the PR and SR values are calculated based on the results of a baseline model, DE/nrand/1/bin algorithm (see details in a section below).

B. Convergence speed

Convergence speed is calculated according to equation (3), using the same MaxFEs settings in Table I. The accuracy level ϵ is set to 1.0E-04. Other parameters are the same as in Table IV. Table V presents the convergence speed results of the DE/nrand/1/bin algorithm.

To further illustrate the behaviour of the niching algorithm, we can record the number of global optima found at different iteration steps of a run. We recommend to use figures to show the mean global optima found averaged over 50 runs, on 5 or 6 different test functions of your choice. We encourage authors to follow a recent paper on niching [21] on how to better present results.

C. Baseline models

To facilitate easy comparisons for participants in the competition, we use as baseline models two Differential Evolution (DE) niching variants, the recently proposed DE/nrand/1/bin algorithm [21] and the well known Crowding DE/rand/1/bin [20]. DE/nrand/1/bin is a simple DE algorithm which incorporates spatial information about the neighborhood of each potential solution to produce a niching formation. On the other hand, Crowding DE/rand/1/bin produces niching formation by incorporating the crowding technique to maintain a better population diversity and therefore to prevent premature convergence to an optimum. The results of the two baseline models are presented in Tables II, III, and Tables V, VI. Please note that we did not conduct any fine-tuning on the parameters of the baseline algorithms. Instead, we used the following default parameters: population size $NP = 100$, $F = 0.5$, $CR = 0.9$, and the crowding factor equals to the population size $CF = NP$.

TABLE IV
PARAMETERS USED FOR PERFORMANCE MEASUREMENT.

Function	r	Peak height	No. global optima
F_1 (1D)	0.01	200.0	2
F_2 (1D)	0.01	1.0	5
F_3 (1D)	0.01	1.0	1
F_4 (2D)	0.01	200.0	4
F_5 (2D)	0.5	1.03163	2
F_6 (2D)	0.5	186.731	18
F_7 (2D)	0.2	1.0	36
F_6 (3D)	0.5	2709.0935	81
F_7 (3D)	0.2	1.0	216
F_8 (2D)	0.01	-2.0	12
F_9 (2D)	0.01	0	6
F_{10} (2D)	0.01	0	8
F_{11} (2D)	0.01	0	6
F_{11} (3D)	0.01	0	6
F_{12} (3D)	0.01	0	8
F_{11} (5D)	0.01	0	6
F_{12} (5D)	0.01	0	8
F_{11} (10D)	0.01	0	6
F_{12} (10D)	0.01	0	8
F_{12} (20D)	0.01	0	8

TABLE V
CONVERGENCE SPEEDS OF THE DE/NRAND/1/BIN ALGORITHM (WITH ACCURACY LEVEL ϵ = 1.0E-04).

Function	F_1(1D)	F_2(1D)	F_3(1D)	F_4(2D)	F_5(2D)
Mean	22886.0	1552.0	1258.0	13610.0	3806.0
St. D.	2689.056	386.106	781.179	1399.453	618.890
Function	F_6(2D)	F_7(2D)	F_6(3D)	F_7(3D)	F_8(2D)
Mean	200000.0	200000.0	400000.0	400000.0	9858.0
St. D.	0.000	0.000	0.000	0.000	833.015
Function	F_9(2D)	F_{10}(2D)	F_{11}(2D)	F_{11}(3D)	F_{12}(3D)
Mean	200000.0	181658.0	200000.0	400000.0	400000.0
St. D.	0.000	42543.630	0.000	0.000	0.000
Function	F_{11}(5D)	F_{12}(5D)	F_{11}(10D)	F_{12}(10D)	F_{12}(20D)
Mean	400000.0	400000.0	400000.0	400000.0	400000.0
St. D.	0.000	0.000	0.000	0.000	0.000

TABLE VI
CONVERGENCE SPEEDS OF THE CROWDING DE/RAND/1/BIN ALGORITHM (WITH ACCURACY LEVEL ϵ = 1.0E-04).

Function	F_1(1D)	F_2(1D)	F_3(1D)	F_4(2D)	F_5(2D)
Mean	50000.0	3386.0	2576.0	41666.0	12980.0
St. D.	0.000	1368.749	2625.974	3772.598	2046.799
Function	F_6(2D)	F_7(2D)	F_6(3D)	F_7(3D)	F_8(2D)
Mean	200000.0	200000.0	400000.0	400000.0	30306.0
St. D.	0.000	0.000	0.000	0.000	1984.677
Function	F_9(2D)	F_{10}(2D)	F_{11}(2D)	F_{11}(3D)	F_{12}(3D)
Mean	200000.0	200000.0	200000.0	400000.0	400000.0
St. D.	0.000	0.000	0.000	0.000	0.000
Function	F_{11}(5D)	F_{12}(5D)	F_{11}(10D)	F_{12}(10D)	F_{12}(20D)
Mean	400000.0	400000.0	400000.0	400000.0	400000.0
St. D.	0.000	0.000	0.000	0.000	0.000

V. RANKING METHOD

We will use *peak ratio* values in Table II as our key criterion to rank algorithms submitted to this competition. The top algorithm is the one that obtains the best average peak ratio, across all test functions and 5 accuracy levels. If there is a tie, then the algorithm having the lower AveFEs in Table V will be the winner.

REFERENCES

[1] K. A. De Jong, "An analysis of the behavior of a class of genetic adaptive systems." Ph.D. dissertation, University of Michigan, 1975.

[2] D. E. Goldberg and J. Richardson, "Genetic algorithms with sharing for multimodal function optimization," in *Proc. of the Second International Conference on Genetic Algorithms*, J. Grefenstette, Ed., 1987, pp. 41–49.

[3] S. W. Mahfoud, "Crowding and preselection revisited," in *Parallel problem solving from nature 2*, R. Männer and B. Manderick, Eds. Amsterdam: North-Holland, 1992, pp. 27–36. [Online]. Available: citeseer.ist.psu.edu/mahfoud92crowding.html

[4] D. Beasley, D. R. Bull, and R. R. Martin, "A sequential niche technique for multimodal function optimization," *Evolutionary Computation*, vol. 1, no. 2, pp. 101–125, 1993. [Online]. Available: citeseer.ist.psu.edu/beasley93sequential.html

[5] G. R. Harik, "Finding multimodal solutions using restricted tournament selection," in *Proc. of the Sixth International Conference on Genetic Algorithms*, L. Eshelman, Ed. San Francisco, CA: Morgan Kaufmann, 1995, pp. 24–31. [Online]. Available: citeseer.ist.psu.edu/harik95finding.html

[6] M. Bessaou, A. Pétrowski, and P. Siarry, "Island model cooperating with speciation for multimodal optimization," in *Parallel Problem Solving from Nature - PPSN VI 6th International Conference*, H.-P. S. et al., Ed. Paris, France: Springer Verlag, 16-20 2000. [Online]. Available: citeseer.ist.psu.edu/bessaou00island.html

[7] K. E. Parsopoulos and M. N. Vrahatis, "On the computation of all global minimizers through particle swarm optimization," *IEEE Transactions on Evolutionary Computation*, vol. 8, no. 3, pp. 211–224, 2004.

[8] X. Yin and N. Germay, "A fast genetic algorithm with sharing scheme using cluster analysis methods in multi-modal function optimization," in *the International Conference on Artificial Neural Networks and Genetic Algorithms*, 1993, pp. 450–457.

[9] A. Pétrowski, "A clearing procedure as a niching method for genetic algorithms," in *Proc. of the 3rd IEEE International Conference on Evolutionary Computation*, 1996, pp. 798–803.

[10] J.-P. Li, M. E. Balazs, G. T. Parks, and P. J. Clarkson, "A species conserving genetic algorithm for multimodal function optimization," *Evol. Comput.*, vol. 10, no. 3, pp. 207–234, 2002.

[11] X. Li, "Niching without niching parameters: Particle swarm optimization using a ring topology," *IEEE Trans. on Evol. Comput.*, vol. 14, no. 1, pp. 150 – 169, February 2010.

[12] K. Deb and A. Saha, "Finding multiple solutions for multimodal optimization problems using a multi-objective evolutionary approach," in *Proceedings of the 12th annual conference on Genetic and evolutionary computation*, ser. GECCO '10. New York, NY, USA: ACM, 2010, pp. 447–454.

[13] A. Saha and K. Deb, "A bi-criterion approach to multimodal optimization: self-adaptive approach," in *Proceedings of the 8th international conference on Simulated evolution and learning*, ser. SEAL'10. Berlin, Heidelberg: Springer-Verlag, 2010, pp. 95–104.

[14] P. N. Suganthan, N. Hansen, J. J. Liang, K. Deb, Y. P. Chen, A. Auger, and S. Tiwari, "Problem definitions and evaluation criteria for the CEC 2005 special session on real-parameter optimization," Nanyang Technological University and KanGAL Report #2005005, IIT Kanpur, India., Tech. Rep., 2005.

[15] K. Deb, "Genetic algorithms in multimodal function optimization (master thesis and tcga report no. 89002)," Ph.D. dissertation, Tuscaloosa: University of Alabama, The Clearinghouse for Genetic Algorithms, 1989.

[16] Z. Michalewicz, *Genetic Algorithms + Data Structures = Evolution Programs*. New York: Springer-Verlag, New York, 1996.

[17] O. Shir and T. Bäck, "Niche radius adaptation in the cms-es niching algorithm," in *Parallel Problem Solving from Nature - PPSN IX, 9th International Conference (LNCS 4193)*. Reykjavik, Iceland: Springer, 2006, pp. 142 – 151.

[18] B.-Y. Qu and P. N. Suganthan, "Novel multimodal problems and differential evolution with ensemble of restricted tournament selection," in *Proceedings of the IEEE Congress on Evolutionary Computation, CEC 2010*. Barcelona, Spain: IEEE, 18-23 July 2010, pp. 1–7.

[19] D. Parrott and X. Li, "Locating and tracking multiple dynamic optima by a particle swarm model using speciation," *IEEE Trans. on Evol. Comput.*, vol. 10, no. 4, pp. 440–458, August 2006.

[20] R. Thomsen, "Multimodal optimization using crowding-based differential evolution," in *Proceedings of the 2004 IEEE Congress on Evolutionary Computation*. Portland, Oregon: IEEE Press, 20-23 Jun. 2004, pp. 1382–1389.

[21] M. Epitropakis, V. Plagianakos, and M. Vrahatis, "Finding multiple global optima exploiting differential evolution's niching capability," in *IEEE Symposium on Differential Evolution, 2011. SDE 2011. (IEEE Symposium Series on Computational Intelligence)*, Paris, France, April 2011, p. 80–87.

TABLE II
PEAK RATIOS AND SUCCESS RATES OF THE DE/NRAND/1/BIN ALGORITHM.

Accuracy level ϵ	F_1 (1D)		F_2 (1D)		F_3 (1D)		F_4 (2D)		F_5 (2D)	
	PR	SR	PR	SR	PR	SR	PR	SR	PR	SR
1.0E-01	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000
1.0E-02	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000
1.0E-03	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000
1.0E-04	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000
1.0E-05	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000
Accuracy level ϵ	F_6 (2D)		F_7 (2D)		F_6 (3D)		F_7 (3D)		F_8 (2D)	
	PR	SR	PR	SR	PR	SR	PR	SR	PR	SR
1.0E-01	0.450	0.000	0.347	0.000	0.108	0.000	0.097	0.000	1.000	1.000
1.0E-02	0.438	0.000	0.346	0.000	0.105	0.000	0.095	0.000	1.000	1.000
1.0E-03	0.440	0.000	0.349	0.000	0.113	0.000	0.099	0.000	0.998	0.980
1.0E-04	0.434	0.000	0.337	0.000	0.112	0.000	0.095	0.000	1.000	1.000
1.0E-05	0.000	0.000	0.333	0.000	0.113	0.000	0.094	0.000	1.000	1.000
Accuracy level ϵ	F_9 (2D)		F_{10} (2D)		F_{11} (2D)		F_{11} (3D)		F_{12} (3D)	
	PR	SR	PR	SR	PR	SR	PR	SR	PR	SR
1.0E-01	0.683	0.000	0.855	0.240	0.667	0.000	0.667	0.000	0.522	0.000
1.0E-02	0.673	0.000	0.837	0.220	0.667	0.000	0.667	0.000	0.535	0.000
1.0E-03	0.683	0.000	0.815	0.140	0.667	0.000	0.667	0.000	0.507	0.000
1.0E-04	0.673	0.000	0.815	0.160	0.667	0.000	0.667	0.000	0.502	0.000
1.0E-05	0.670	0.000	0.777	0.100	0.667	0.000	0.667	0.000	0.507	0.000
Accuracy level ϵ	F_{11} (5D)		F_{12} (5D)		F_{11} (10D)		F_{12} (10D)		F_{12} (20D)	
	PR	SR	PR	SR	PR	SR	PR	SR	PR	SR
1.0E-01	0.677	0.000	0.345	0.000	0.403	0.000	0.227	0.000	0.130	0.000
1.0E-02	0.663	0.000	0.325	0.000	0.343	0.000	0.167	0.000	0.127	0.000
1.0E-03	0.663	0.000	0.295	0.000	0.323	0.000	0.152	0.000	0.130	0.000
1.0E-04	0.663	0.000	0.290	0.000	0.270	0.000	0.125	0.000	0.125	0.000
1.0E-05	0.657	0.000	0.287	0.000	0.250	0.000	0.127	0.000	0.123	0.000

TABLE III
PEAK RATIOS AND SUCCESS RATES OF THE CROWDING DE/RAND/1/BIN ALGORITHM.

Accuracy level ϵ	F_1 (1D)		F_2 (1D)		F_3 (1D)		F_4 (2D)		F_5 (2D)	
	PR	SR	PR	SR	PR	SR	PR	SR	PR	SR
1.0E-01	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000
1.0E-02	0.710	0.500	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000
1.0E-03	0.090	0.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000
1.0E-04	0.020	0.000	1.000	1.000	1.000	1.000	0.995	0.980	1.000	1.000
1.0E-05	0.000	0.000	1.000	1.000	1.000	1.000	0.420	0.040	1.000	1.000
Accuracy level ϵ	F_6 (2D)		F_7 (2D)		F_6 (3D)		F_7 (3D)		F_8 (2D)	
	PR	SR	PR	SR	PR	SR	PR	SR	PR	SR
1.0E-01	1.000	1.000	0.703	0.000	0.847	0.000	0.271	0.000	1.000	1.000
1.0E-02	0.999	0.980	0.724	0.000	0.835	0.000	0.272	0.000	1.000	1.000
1.0E-03	0.972	0.740	0.715	0.000	0.716	0.000	0.274	0.000	1.000	1.000
1.0E-04	0.107	0.000	0.709	0.000	0.290	0.000	0.274	0.000	1.000	1.000
1.0E-05	0.000	0.000	0.716	0.000	0.038	0.000	0.270	0.000	1.000	1.000
Accuracy level ϵ	F_9 (2D)		F_{10} (2D)		F_{11} (2D)		F_{11} (3D)		F_{12} (3D)	
	PR	SR	PR	SR	PR	SR	PR	SR	PR	SR
1.0E-01	0.937	0.720	0.380	0.000	0.837	0.400	0.683	0.000	0.730	0.000
1.0E-02	0.690	0.040	0.055	0.000	0.683	0.020	0.667	0.000	0.690	0.000
1.0E-03	0.667	0.000	0.007	0.000	0.667	0.000	0.667	0.000	0.627	0.000
1.0E-04	0.667	0.000	0.007	0.000	0.667	0.000	0.667	0.000	0.490	0.000
1.0E-05	0.667	0.000	0.002	0.000	0.667	0.000	0.667	0.000	0.375	0.000
Accuracy level ϵ	F_{11} (5D)		F_{12} (5D)		F_{11} (10D)		F_{12} (10D)		F_{12} (20D)	
	PR	SR	PR	SR	PR	SR	PR	SR	PR	SR
1.0E-01	0.697	0.000	0.567	0.080	0.517	0.080	0.000	0.000	0.502	0.380
1.0E-02	0.667	0.000	0.425	0.000	0.250	0.000	0.000	0.000	0.013	0.000
1.0E-03	0.667	0.000	0.280	0.000	0.200	0.000	0.000	0.000	0.000	0.000
1.0E-04	0.667	0.000	0.115	0.000	0.173	0.000	0.000	0.000	0.000	0.000
1.0E-05	0.667	0.000	0.047	0.000	0.170	0.000	0.000	0.000	0.000	0.000