Problem Definitions and Evaluation Criteria for the CEC 2015 Competition on Learning-based Real-Parameter Single Objective Optimization

J. J. Liang[1], B. Y. Qu[2], P. N. Suganthan[3], Q. Chen[4]

[1]School of Electrical Engineering, Zhengzhou University, Zhengzhou, China
[2]School of Electric and Information Engineering, Zhongyuan University of Technology, Zhengzhou, China
[3] School of Electrical and Electronic Engineering, Nanyang Technological University, Singapore
[4]Facility Design and Instrument Institute, China Aerodynamic Research and Development Center, China

liangjing@zzu.edu.cn, qby1984@hotmail.com, epnsugan@ntu.edu.sg, chenqin1980@gmail.com

Technical Report 201411A, Computational Intelligence Laboratory,
Zhengzhou University, Zhengzhou China
And
Technical Report, Nanyang Technological University, Singapore

November 2014

1. Introduction

Single objective optimization algorithms are the basis of the more complex optimization algorithms such as multi-objective, niching, dynamic, constrained optimization algorithms and so on. Research on single objective optimization algorithms influence the development of the optimization branches mentioned above. In the recent years, various kinds of novel optimization algorithms have been proposed to solve real-parameter optimization problems.

This special session is devoted to the approaches, algorithms and techniques for solving real parameter single objective optimization without knowing the exact equations of the test functions (i.e. blackbox optimization). We encourage all researchers to test their algorithms on the CEC'15 test suites. The participants are required to send the final results(after submitting their final paper version in March 2015)in the format specified in this technical report to the organizers. The organizers will present an overall analysis and comparison based on these results. We will also use statistical tests on convergence performance to compare algorithms that eventually generate similar final solutions. Papers on novel concepts that help us in understanding problem characteristics are also welcome.

Results of 10D and 30D problems are acceptable for the first review submission. However, other dimensional results as specified in the technical report should also be included in the final version, if space permits. Thus, final results for all dimensions in the format introduced in the technical report should be zipped and sent to the organizers after the final version of the paper is submitted.

Please note that in this competition error values smaller than 10^{-8} will be taken as zero.

You can download the C, JAVA and Matlab codes for CEC'15 test suite from the website given below:

http://www.ntu.edu.sg/home/EPNSugan/index_files/CEC2015/CEC2015.htm

This technical report presents the details of benchmark suite used for CEC'15 competition on learning based single objective global optimization.

1.1 Introduction to Learning-Based Problems

As a relatively new solver for the optimization problems, evolutionary algorithm has attracted the attention of researchers in various fields. When testing the performance of a novel evolutionary algorithm, we always choose a group of benchmark functions and compare the proposed new algorithm with other existing algorithms on these benchmark functions. To obtain fair comparison results and to simplify the experiments, we always set the parameters of the algorithms to be the same for all test functions. In general, specifying different sets of parameters for different test functions is not allowed. Due to this approach, we lose the opportunity to analyze how to adjust the algorithm to solve a specified problem in the most effective manner. As we all know that there is no free lunch and for solving a particular real-world problem, we only need one most effective algorithm. In practice, it is hard to imagine a scenario whereby a researcher or engineer has to solve highly diverse problems at the same time. In other words, a practicing engineer is more likely to solve numerous instances of a particular problem. Under this consideration and by the fact that by shifting the position of the optimum and mildly changing the rotation matrix will not change the properties of the benchmark functions significantly, we propose a set of learning-based benchmark problems. In this competition, the participants are allowed to optimize the parameters of their proposed (hybrid) optimization algorithm for each problem. Although a completely different optimization algorithm might be used for solving each of the 15 problems, this approach is strongly discouraged, as our objective is to develop a highly tunable algorithm to solve diverse instances of real-world problems. In other words, our objective is not to identify the best algorithms for solving each of the 15 synthetic benchmark problems.

To test the generalization performance of the algorithm and associated parameters, the competition has two stages:

Stage 1: Infinite instances of shifted optima and rotation matrixes can be generated. The participants can optimize the parameters of their proposed algorithms for each problem with these data and write the paper. Adaptive learning methods are also allowed.

Stage 2: A different testing set of shifted optima and rotation matrices will be provided to test the algorithms with the optimized parameters in Stage 1. The performance on the testing set will be used for the final ranking.

1.2 Summary of the CEC'15 Learning-Based Benchmark Suite

TableI. Summary of the CEC'15 Learning-Based Benchmark Suite

	No.	Functions	$F_i*=F_i(x*)$
Unimodal Functions	1	Rotated High Conditioned Elliptic Function	100
	2	Rotated Cigar Function	200
Simple Multimodal Functions	3	Shifted and Rotated Ackley's Function	300
	4	Shifted and Rotated Rastrigin's Function	400
	5	Shifted and Rotated Schwefel's Function	500
Hybrid Functions	6	Hybrid Function 1 (N=3)	600
	7	Hybrid Function 2 (N=4)	700
	8	Hybrid Function 3(N=5)	800
Composition Functions	9	Composition Function 1 (N=3)	900
	10	Composition Function 2 (N=3)	1000
	11	Composition Function 3 (N=5)	1100
	12	Composition Function 4 (N=5)	1200
	13	Composition Function 5 (N=5)	1300
	14	Composition Function 6 (N=7)	1400
	15	Composition Function 7 (N=10)	1500
Search Range: $[-100,100]^D$			

***Please Note:**

1. These problems should be treated as **black-box problems**. The explicit equations of the problems are not to be used.

2. These functions are with **bounds constraints**. Searching beyond the search range is not allowed.

1.3 Some Definitions:

All test functions are minimization problems defined as following:

$$\text{Min } f(\mathbf{x}),\ \boldsymbol{x} = [x_1, x_2, ..., x_D]^{\mathrm{T}}$$

D: dimensions.

$\boldsymbol{o}_{i1} = [o_{i1}, o_{i2}, ..., o_{iD}]^{\mathrm{T}}$: the shifted global optimum (defined in “shift_data_x.txt”), which is randomly distributed in $[-80,80]^D$. Each function has a shift data for CEC’14.

All test functions are shifted to $\boldsymbol{o}$ and scalable.

For convenience, the same search ranges are defined for all test functions.

Search range: $[-100,100]^D$.

$\mathbf{M}_i$: rotation matrix. Different rotation matrices are assigned to each function and each basic function.

The variables are divided into subcomponents randomly. The rotation matrix for each subcomponents are generated from standard normally distributed entries by Gram-Schmidt ortho-normalization with condition number *c* that is equal to 1 or 2.

1.4 Definitions of the Basic Functions

1) High Conditioned Elliptic Function

$$f_1(\boldsymbol{x}) = \sum_{i=1}^{D} (10^6)^{\frac{i-1}{D-1}} \boldsymbol{x}_i^2 \tag{1}$$

2) Cigar Function

$$f_2(\boldsymbol{x}) = x_1^2 + 10^6 \sum_{i=2}^{D} x_i^2 \tag{2}$$

3) Discus Function

$$f_3(\boldsymbol{x}) = 10^6 x_1^2 + \sum_{i=2}^{D} x_i^2 \tag{3}$$

4) Rosenbrock's Function

$$f_4(\boldsymbol{x}) = \sum_{i=1}^{D-1}(100(x_i^2 - x_{i+1})^2 + (x_i - 1)^2) \tag{4}$$

5) Ackley's Function

$$f_5(\boldsymbol{x}) = -20\exp(-0.2\sqrt{\frac{1}{D}\sum_{i=1}^{D} x_i^2}) - \exp(\frac{1}{D}\sum_{i=1}^{D}\cos(2\pi x_i)) + 20 + e \tag{5}$$

6) Weierstrass Function

$$f_6(\boldsymbol{x}) = \sum_{i=1}^{D}(\sum_{k=0}^{k\max}[a^k \cos(2\pi b^k (x_i + 0.5))]) - D\sum_{k=0}^{k\max}[a^k \cos(2\pi b^k \cdot 0.5)] \tag{6}$$

a=0.5, b=3, $kmax$=20

7) Griewank's Function

$$f_7(\boldsymbol{x}) = \sum_{i=1}^{D}\frac{{x_i}^2}{4000} - \prod_{i=1}^{D}\cos(\frac{x_i}{\sqrt{i}}) + 1 \tag{7}$$

8) Rastrigin's Function

$$f_8(\boldsymbol{x}) = \sum_{i=1}^{D}({x_i}^2 - 10\cos(2\pi x_i) + 10) \tag{8}$$

9) Modified Schwefel's Function

$$f_9(\boldsymbol{x}) = 418.9829 \times D - \sum_{i=1}^{D} g(z_i), \qquad z_i = x_i + 4.209687462275036\text{e+}002$$

$$g(z_i) = \begin{cases} z_i \sin(|z_i|^{1/2}) & \text{if } |z_i| \le 500 \\ (500 - \text{mod}(z_i, 500))\sin(\sqrt{|500 - \text{mod}(z_i, 500)|}) - \dfrac{(z_i - 500)^2}{10000D} & \text{if } z_i > 500 \\ (\text{mod}(|z_i|, 500) - 500)\sin(\sqrt{|\text{mod}(|z_i|, 500) - 500|}) - \dfrac{(z_i + 500)^2}{10000D} & \text{if } z_i < -500 \end{cases} \tag{9}$$

10) Katsuura Function

$$f_{10}(\boldsymbol{x}) = \frac{10}{D^2}\prod_{i=1}^{D}(1 + i\sum_{j=1}^{32}\frac{|2^j x_i - round(2^j x_i)|}{2^j})^{\frac{10}{D^{1.2}}} - \frac{10}{D^2} \tag{10}$$

11) HappyCat Function

$$f_{11}(\boldsymbol{x}) = \left|\sum_{i=1}^{D} x_i^2 - D\right|^{1/4} + (0.5\sum_{i=1}^{D} x_i^2 + \sum_{i=1}^{D} x_i)/D + 0.5 \tag{11}$$

12) HGBatFunction

$$f_{12}(\boldsymbol{x}) = \left|(\sum_{i=1}^{D} x_i^2)^2 - (\sum_{i=1}^{D} x_i)^2\right|^{1/2} + (0.5\sum_{i=1}^{D} x_i^2 + \sum_{i=1}^{D} x_i)/D + 0.5 \tag{12}$$

13) Expanded Griewank's plus Rosenbrock's Function

$$f_{13}(\boldsymbol{x}) = f_7(f_4(x_1,x_2)) + f_7(f_4(x_2,x_3)) + ... + f_7(f_4(x_{D-1},x_D)) + f_7(f_4(x_D,x_1)) \tag{13}$$

14) Expanded Scaffer's F6 Function

$$\text{Scaffer's F6 Function: } g(x,y) = 0.5 + \frac{(\sin^2(\sqrt{x^2+y^2}) - 0.5)}{(1+0.001(x^2+y^2))^2}$$

$$f_{14}(\boldsymbol{x}) = g(x_1,x_2) + g(x_2,x_3) + ... + g(x_{D-1},x_D) + g(x_D,x_1) \tag{14}$$

1.5 Definitions of the CEC'15Learning-Based Benchmark Suite

A. Unimodal Functions:

1) Rotated High Conditioned Elliptic Function

$$F_1(\boldsymbol{x}) = f_1(\mathbf{M}_1(\boldsymbol{x} - \boldsymbol{o}_1)) + F_1{}^* \tag{15}$$

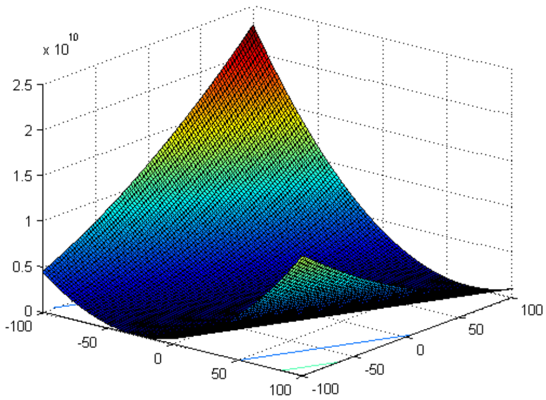

Figure 1.3-*D* map for 2-*D* function

Properties:

- Unimodal
- Non-separable
- Quadratic ill-conditioned

2) Rotated Cigar Function

$$F_2(\boldsymbol{x}) = f_2(\mathbf{M}_2(\boldsymbol{x}-\boldsymbol{o}_2)) + F_2{}^* \tag{16}$$

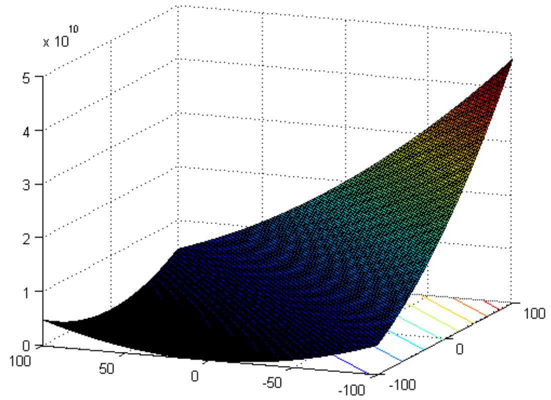

Figure 2. 3-*D* map for 2-*D* function

Properties:

- Unimodal
- Non-separable
- Smooth but narrow ridge

B. Multimodal Functions

3) Shifted and Rotated Ackley's Function

$$F_3(\boldsymbol{x}) = f_5(\mathbf{M}_3(\boldsymbol{x}-\boldsymbol{o}_3)) + F_3{}^* \tag{17}$$

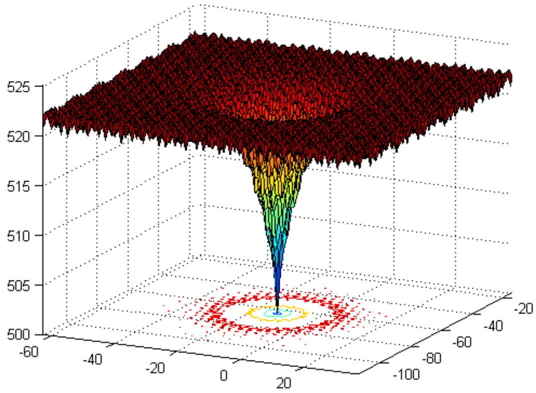

Figure 3. 3-*D* map for 2-*D* function

Properties:

- Multi-modal
- Non-separable

4) Shifted and Rotated Rastrigin's Function

$$F_4(\mathbf{x}) = f_8(\mathbf{M}_4(\frac{5.12(\mathbf{x}-\mathbf{o}_4)}{100})) + F_4{}^* \tag{18}$$

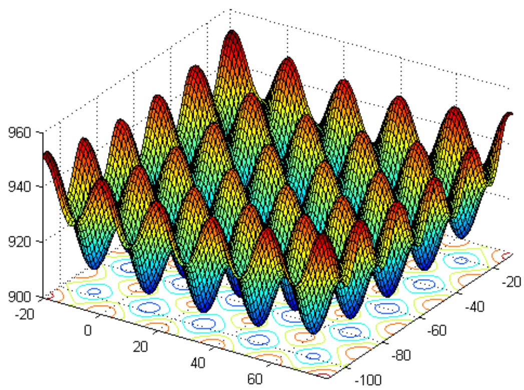

Figure 4. 3-*D* map for 2-*D* function

Properties:

- Multi-modal
- Non-separable
- Local optima's number is huge

5) Shifted and Rotated Schwefel's Function

$$F_5(\mathbf{x}) = f_9(\mathbf{M}_5(\frac{1000(\mathbf{x}-\mathbf{o}_5)}{100})) + F_5{}^* \tag{19}$$

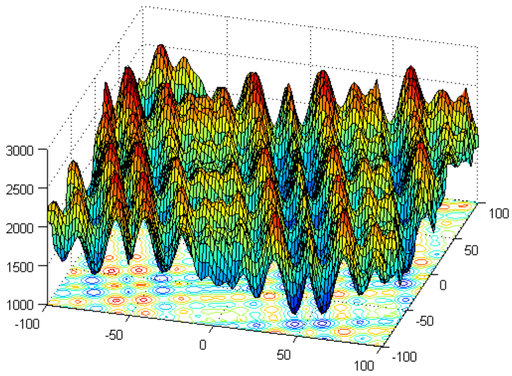

Figure 5(a). 3-*D* map for 2-*D* function

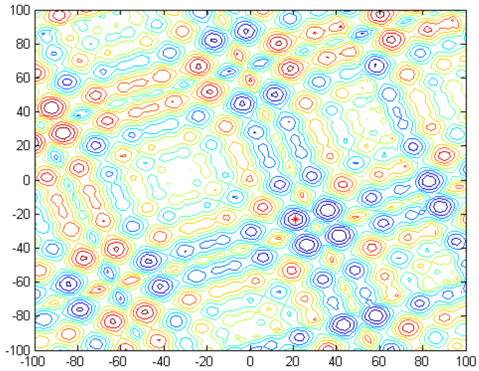

Figure 5(b).Contour map for 2-*D* function

Properties:

- Multi-modal
- Non-separable
- Local optima's number is huge and second better local optimum is far from the global optimum.

C. Hybrid Functions

Considering that in the real-world optimization problems, different subcomponents of the variables may have different properties[8]. In this set of hybrid functions, the variables are randomly divided into some subcomponents and then different basic functions are used for different subcomponents.

$$F(\boldsymbol{x}) = g_1(\mathbf{M}_1\boldsymbol{z}_1) + g_2(\mathbf{M}_2\boldsymbol{z}_2) + ... + g_N(\mathbf{M}_N\boldsymbol{z}_N) + F^*(\boldsymbol{x}) \tag{20}$$

$F(\boldsymbol{x})$: hybrid function

$g_i(\boldsymbol{x})$: i^{th} basic function used to construct the hybrid function

N: number of basic functions

$\boldsymbol{z} = [\boldsymbol{z}_1, \boldsymbol{z}_2, ..., \boldsymbol{z}_N]$

$\boldsymbol{z}_1 = [\boldsymbol{y}_{S_1}, \boldsymbol{y}_{S_2}, ..., \boldsymbol{y}_{S_{n_1}}], \boldsymbol{z}_2 = [\boldsymbol{y}_{S_{n_1+1}}, \boldsymbol{y}_{S_{n_1+2}}, ..., \boldsymbol{y}_{S_{n_1+n_2}}], ..., \boldsymbol{z}_N = [\boldsymbol{y}_{S_{\sum_{i=1}^{N-1} n_i+1}}, \boldsymbol{y}_{S_{\sum_{k=1}^{N-1} n_i+2}}, ..., \boldsymbol{y}_{S_D}]$

$\boldsymbol{y} = \boldsymbol{x} - \boldsymbol{o}_i, S = randperm(1:D)$

p_i: used to control the percentage of $g_i(\boldsymbol{x})$

n_i: dimension for each basic function $\sum_{i=1}^{N} n_i = D$

$$n_1 = \lceil p_1 D \rceil, n_2 = \lceil p_2 D \rceil, ..., n_{N-1} = \lceil p_{N-1} D \rceil, n_N = D - \sum_{i=1}^{N-1} n_i$$

Properties:

- Multi-modal or Unimodal, depending on the basic function
- Non-separable subcomponents
- Different properties for different variables subcomponents

6) Hybrid Function 1

N= 3

p = [0.3,0.3,0.4]

g_1: Modified Schwefel's Function f_9

g_2: Rastrigin's Function f_8

g_3: High Conditioned Elliptic Function f_1

7) Hybrid Function 3

N= 4

p =[0.2,0.2,0.3,0.3]

g_1: Griewank's Function f_7

g_2: Weierstrass Function f_6

g_3: Rosenbrock's Function f_4

g_4: Scaffer's F6 Function f_{14}

8) Hybrid Function 5

N= 5

p = [0.1,0.2,0.2,0.2,0.3]

g_1: Scaffer's F6 Function f_{14}

g_2: HGBat Function f_{12}

g_3: Rosenbrock's Function f_4

g_4: Modified Schwefel's Function f_9

g_5: High Conditioned Elliptic Function f_1

D. Composition Functions

$$F(\boldsymbol{x}) = \sum_{i=1}^{N} \{\omega_i * [\lambda_i g_i(\boldsymbol{x}) + bias_i]\} + F^* \tag{21}$$

$F(\boldsymbol{x})$: composition function

$g_i(\boldsymbol{x})$: i^{th} basic function used to construct the composition function

N: number of basic functions

o_i: new shifted optimum position for each $g_i(\boldsymbol{x})$, define the global and local optima's position

$bias_i$: defines which optimum is global optimum

σ_i : used to control each $g_i(\boldsymbol{x})$'s coverage range, a small σ_i give a narrow range for that $g_i(x)$

λ_i : used to control each $g_i(\boldsymbol{x})$'s height

w_i : weight value for each $g_i(\boldsymbol{x})$, calculated as below:

$$w_i = \frac{1}{\sqrt{\sum_{j=1}^{D}(x_j - o_{ij})^2}} \exp\left(-\frac{\sum_{j=1}^{D}(x_j - o_{ij})^2}{2D\sigma_i^2}\right) \tag{22}$$

Then normalize the weight $\omega_i = w_i / \sum_{i=1}^{n} w_i$

So when $\boldsymbol{x} = \boldsymbol{o}_i$, $\omega_j = \begin{cases} 1 & j = i \\ 0 & j \neq i \end{cases}$ for $j = 1, 2, ..., N$, $f(x) = bias_i + f^*$

The local optimum which has the smallest bias value is the global optimum. The composition function merges the properties of the sub-functions better and maintains continuity around the global/local optima.

Functions $F_i' = F_i - F_i^*$ are used as g_i. In this way, the function values of global optima of g_i are equal to 0 for all composition functions in this report.

For some composition functions, the hybrid functions are also used as the basic functions. With hybrid functions as the basic functions, the composition function can have different properties for different variables subcomponents.

9) Composition Function 1

$N = 3$

$\sigma = [20,20,20]$

$\lambda = [1, 1, 1]$

$bias = [0, 100, 200] + F_9^*$

g_1

- Schwefel's Function

g_2, g_3:

- Rotated Rastrigin's Function
- Rotated HGBat Function

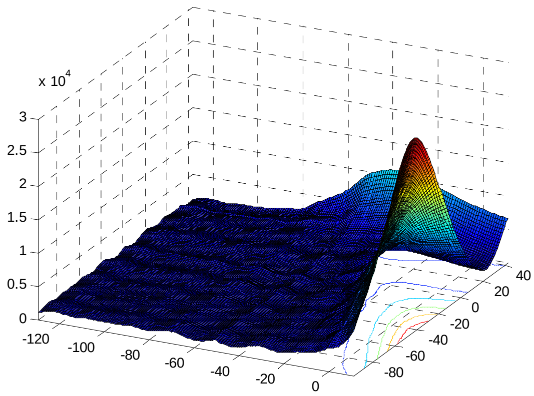

Figure6(a). 3-*D* map for 2-*D* function (example)

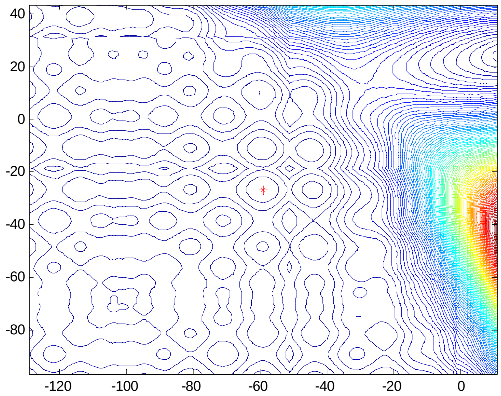

Figure6(b).Contour map for 2-*D* function (example)

Properties:

- Multi-modal
- Non-separable
- Different properties around different local optima
- The basic function of which the global optimum belongs to is fixed. The sequence of the other basic functions can be randomly generated.

10) Composition Function 2

$N = 3$

$\sigma = [10, 30, 50]$

$\lambda = [1, 1, 1]$

$bias$ =[0, 100, 200]+F_{10}*

g_1, g_2, g_3:

- Hybrid Function 1
- Hybrid Function 2
- Hybrid Function 3

Properties:

- Multi-modal
- Non-separable
- Asymmetrical
- Different properties around different local optima
- Different properties for different variables subcomponents

- The sequence of the basic functions can be randomly generated.

11) Composition Function 3

$N = 5$

$\sigma = [10, 10, 10, 20, 20]$

$\lambda = [10, 10, 2.5, 25, 1e\text{-}6]$

$bias$ =[0, 100, 200, 300, 400]+F_{11}*

g_1:

- Rotated HGBat Function

g_2, g_3,g_4,g_5:

- Rotated Rastrigin’s Function
- Rotated Schwefel's Function
- Rotated Weierstrass Function
- Rotated High Conditioned Elliptic Function

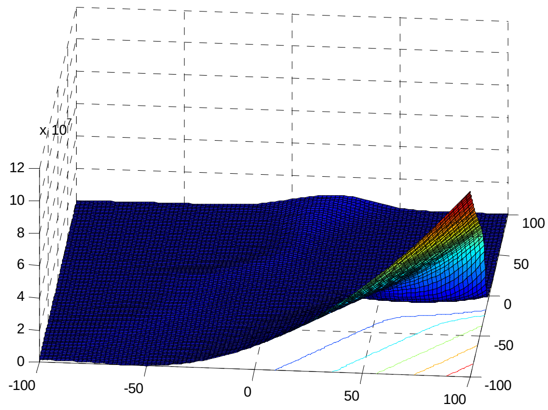

Figure 8(a). 3-*D* map for 2-*D* function (example)

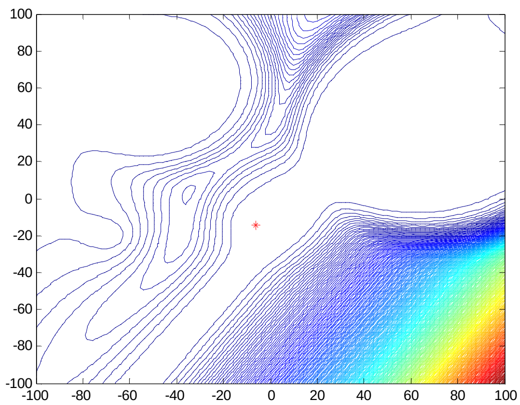

Figure8(b).Contour map for 2-*D* function (example)

Properties:

- Multi-modal
- Non-separable
- Asymmetrical
- Different properties around different local optima
- The basic function of which the global optimum belongs to is fixed. The sequence of the other basic functions can be randomly generated.

12) Composition Function 4

$N = 5$

$\sigma = [10,20,20,30,30]$

$\lambda = [0.25, 1, 1e\text{-}7, 10, 10]$

$bias$ =[0, 100, 100, 200, 200]+F_{12}*

g_1, g_2, g_3, g_4, g_5:

- Rotated Schwefel's Function
- Rotated Rastrigin's Function
- Rotated High Conditioned Elliptic Function
- Rotated Expanded Scaffer's F6 Function
- Rotated HappyCat Function

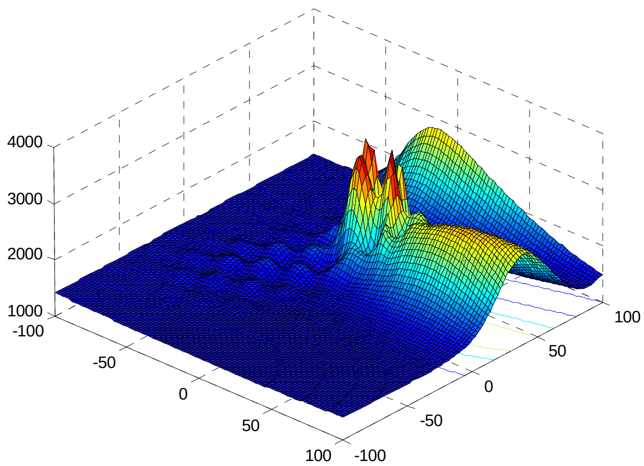

Figure9(a). 3-*D* map for 2-*D* function (example)

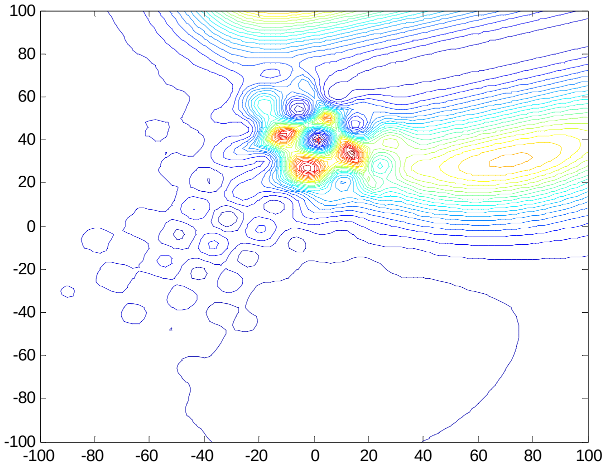

Figure9(b).Contour map for 2-*D* function (example)

Properties:

- Multi-modal
- Non-separable
- Asymmetrical
- Different properties around different local optima
- Different properties for different variables subcomponents
- The sequence of the basic functions can be randomly generated

13) Composition Function 5

$N = 5$

$\sigma = [10, 10, 10, 20, 20]$

$\lambda = [1, 10, 1, 25, 10]$

$bias = [0, 100, 200, 300, 400] + F_{13}*$

g_1, g_2, g_3, g_4, g_5:

- Hybrid Function 3
- Rotated Rastrigin's Function
- Hybrid Function 1
- Rotated Schwefel's Function
- Rotated Expanded Scaffer's F6 Function

Properties:

- Multi-modal
- Non-separable
- Asymmetrical
- Different properties around different local optima
- The sequence of the basic functions can be randomly generated

14) Composition Function 6

$N = 7$

$\sigma = [10, 20, 30, 40, 50, 50, 50]$

$\lambda = [10, 2.5, 2.5, 10, 1e\text{-}6, 1e\text{-}6, 10]$

$bias = [0, 100, 200, 300, 300, 400, 400] + F_{14}{}^*$

g_1:

- Rotated HappyCat Function

g_2, g_3, g_4, g_5, g_6, g_7:

- Rotated Expanded Griewank's plus Rosenbrock's Function
- Rotated Schwefel's Function
- Rotated Expanded Scaffer's F6 Function
- Rotated High Conditioned Elliptic Function
- Rotated Cigar Function
- Rotated Rastrigin's Function

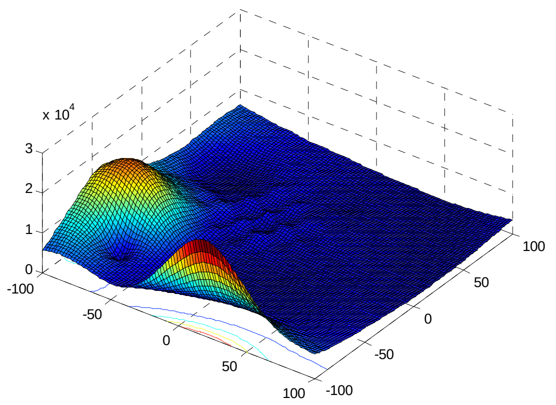

Figure 10(a). 3-*D* map for 2-*D* function (example)

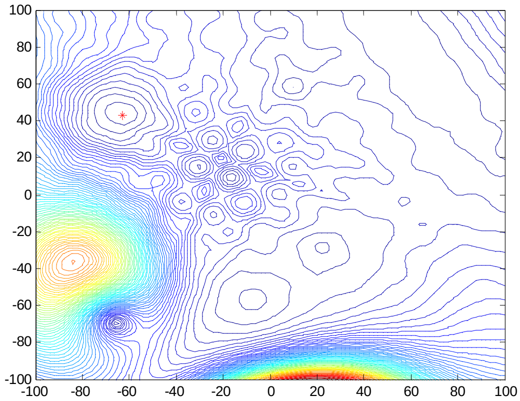

Figure 10(b).Contour map for 2-*D* function (example)

Properties:

- Multi-modal
- Non-separable
- Asymmetrical
- Different properties around different local optima
- The basic function of which the global optimum belongs to is fixed. The sequence of the other basic functions can be randomly generated.

15) Composition Function 7

$N = 10$

$\sigma = [10, 10, 20, 20, 30, 30, 40, 40, 50, 50]$

$\lambda = [0.1, 2.5\text{e-}1, 0.1, 2.5\text{e-}2, 1\text{e-}3, 0.1, 1\text{e-}5, 10, 2.5\text{e-}2, 1\text{e-}3]$

$bias$ =[0, 100, 100, 200, 200, 300, 300, 400, 400, 500]+F_{15}*

$g_1, g_2, g_3, g_4, g_5, g_6, g_7, g_8, g_9, g_{10}$:

- Rotated Rastrigin's Function
- Rotated Weierstrass Function
- Rotated HappyCat Function
- Rotated Schwefel's Function
- Rotated Rosenbrock's Function
- Rotated HGBat Function
- Rotated Katsuura Function

- Rotated Expanded Scaffer's F6 Function
- Rotated Expanded Griewank's plus Rosenbrock's Function
- Rotated Ackley Function

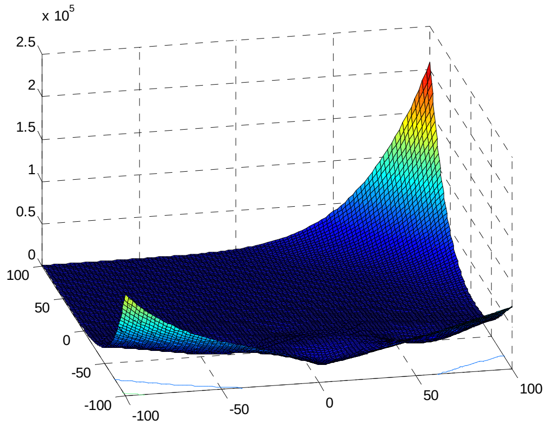

Figure 11(a). 3-*D* map for 2-*D* function (example)

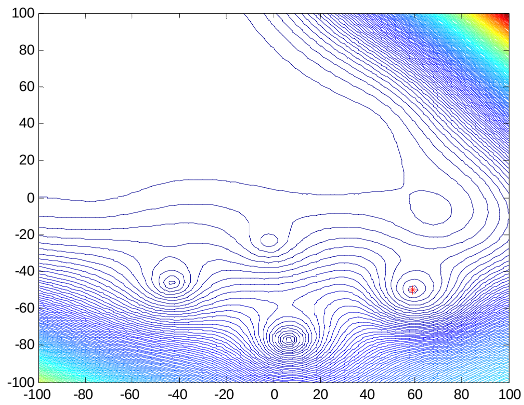

Figure 11(b).Contour map for 2-*D* function (example)

Properties:

- Multi-modal
- Non-separable
- Asymmetrical
- Different properties around different local optima
- The sequence of the basic functions can be randomly generated

2.Evaluation Criteria

2.1 Experimental Setting

Problems: 15 minimization problems

Dimensions: D=10, 30, 50, 100 (Results only for 10D and 30D are acceptable for the initial submission; but 50D and 100D should be included in the final version)

Runs / problem:51 (**Do not run many 51 runs to pick the best run**)

MaxFES: 10000*D (Max_FES for 10D= 100000; for 30D=300000; for 50D = 500000; for 100D = 1000000)

SearchRange: $[-100,100]^D$

Initialization: Uniform random initialization within the search space. Random seed is based on time, Matlab users can use rand('state', sum(100*clock)).

Global Optimum: All problems have the global optimum within the given bounds and there is no need to perform search outside of the given bounds for these problems. $F_i(\boldsymbol{x}^*) = F_i(\boldsymbol{o}_i) = F_i{}^*$

Termination: Terminate when reaching MaxFES or the error value is smaller than 10^{-8}.

2.1 Results Record

1) **Record function error value ($F_i(x)$-$F_i(x^*)$) after (0.0001, 0.001, 0.01, 0.02, 0.03, 0.04, 0.05, 0.1, 0.2, 0.3,0.4,0.5,0.6,0.7,0.8,0.9, 1.0)*MaxFES for each run.**

   In this case, **17** error values are recorded for each function for each run. Sort the error values achieved after MaxFES in 51 runs from the smallest (best) to the largest (worst**)** and present the **best, worst, mean, median** and **standard variance** values of function error values for the 51 runs.

   Please Notice: Error value smaller than 10^{-8} will be taken as zero.

2) **Algorithm Complexity**

a) Run the test program below:

for i=1:1000000

x= 0.55 + (double)*i*;

x=*x* + *x*; *x*=*x*/2; *x=x*x; x=sqrt(x); x=log(x); x=exp(x); x=x/(x*+2*);*

end

Computing time for the above=*T0*;

b) Evaluate the computing time just for **Function 1**. For 200000 evaluations of a certain dimension *D*, it gives *T1*;

c) The complete computing time for the algorithm with 200000 evaluations of the same *D* dimensional **Function 1** is *T2*.

d) Execute step c **five** times and get **five** *T2* values. $\hat{T}2$ =Mean(*T2*)

The complexity of the algorithm is reflected by: $\hat{T}2$,*T1*, *T0*, and ($\hat{T}2$ -*T1*)/*T0*

The algorithm complexities are calculated on 10, 30, 50 and 100 dimensions, to show the algorithm complexity's relationship with dimension. Also provide sufficient details on the computing system and the programming language used. In step c, we execute the complete algorithm **five**times to accommodate variations in execution time due adaptive nature of some algorithms.

Please Note: Similar programming styles should be used for all *T0, T1* and *T2*.

(For example, if *m* individuals are evaluated at the same time in the algorithm, the same style should be employed for calculating *T1*; if parallel calculation is employed for calculating *T2*, the same way should be used for calculating *T0* and *T1*. In other word, the complexity calculation should be fair.)

3) Parameters

Participants are allowed to search for a distinct set of parameters for each problem. Please provide details on the following whenever applicable:

a) All parameters to be adjusted;

b) Corresponding dynamic ranges;

c) Guidelines on how to adjust the parameters;

d) Estimated cost of parameter tuning in terms of number of FEs;

e) Actual parameter values used for each problem.

4) Results Format

The participants are required to send the final results as the following format to the organizers and the organizers will present an overall analysis and comparison based on these results.

Create one txt document with the name "AlgorithmName_FunctionNo._D.txt" for each test function and for each dimension.

For example, PSO results for test function 5 and D=30, the file name should be "PSO_5_30.txt".

Then save the results matrix (***the gray shadowing part***) as Table II in the file:

Table II. Information Matrix for *D* Dimensional Function X

***.txt	Run 1	Run 2	…	Run 51
Function error values when FES=0.0001*MaxFES				
Function error values when FES=0.001*MaxFES				
Function error values when FES=0.01*MaxFES				
Function error values when FES=0.02*MaxFES				
Function error values when FES=0.03*MaxFES				
Function error values when FES=0.04*MaxFES				
Function error values when FES=0.05*MaxFES				
……				
Function error values when FES=0.9*MaxFES				
Function error values when FES=MaxFES				

Thus **15*4**(10D, 30D, 50D and 100D)files (each file contains a **17*51**matrix.) and a list of the parameters used for each function should be zipped and sent to the organizers.

Notice: All participants are allowed to improve their algorithms further after submitting the initial version of their papers to CEC2014. And they are required to submit their results in the introduced format to the organizers after submitting the **final** version of paper as soon as possible.

2.3ResultsTemple

Language: Matlab 2013a

Algorithm: Particle Swarm Optimizer (PSO)

Results

Notice:

Considering the length limit of the paper, only Error Values Achieved with MaxFES are need to be listed. While the authors are required to send all results (15*4 files described in section 2.2) to the organizers for a better comparison among the algorithms.

Table III. Results for 10D

Func.	Best	Worst	Median	Mean	Std
1					
2					
3					
4					
5					
6					
7					
8					
9					
10					
11					
12					
13					
14					
15					

Table IV. Results for 30D

…

Table V. Results for 50D

…

Table VI. Results for 100D

…

Algorithm Complexity

Table VII. Computational Complexity

	$T0$	$T1$	$\hat{T}2$	$(\hat{T}2 - T1)/T0$
D=10				
D=30				
D=50				
D=100				

Parameters

a) All parameters to be adjusted

b) Corresponding dynamic ranges

c) Guidelines on how to adjust the parameters

d) Estimated cost of parameter tuning in terms of number of FES

e) Actual parameter values used.

References

[1] Q. Chen, B. Liu and Q. Zhang, J. J. Liang, P. N. Suganthan, B. Y. Qu Problem Definitions and Evaluation Criteria for CEC 2015 Special Session on Computationally Expensive Single Objective Optimization, Technical Report, 2014.

[2] B. Y. Qu, J. J. Liang, P. N. Suganthan, Q. Chen, "Problem Definitions and Evaluation Criteria for the CEC 2015 Special Session and Competition on Single Objective Multi-Niche Optimization", Technical Report201411B,Computational Intelligence Laboratory, Zhengzhou University, Zhengzhou China and Technical Report, Nanyang Technological University, Singapore, November 2014

[3] P. N. Suganthan, N. Hansen, J. J. Liang, K. Deb, Y.-P. Chen, A. Auger & S. Tiwari, "Problem Definitions and Evaluation Criteria for the CEC 2005 Special Session on Real-Parameter Optimization," Technical Report, Nanyang Technological University, Singapore, May 2005 and KanGAL Report #2005005, IIT Kanpur, India, 2005.

[4] J. J. Liang, B. Y. Qu, P. N. Suganthan, Alfredo G. Hernández-Díaz, "Problem Definitions and Evaluation Criteria for the CEC 2013 Special Session and Competition on Real-Parameter Optimization", Technical Report 201212, Computational Intelligence

Laboratory, Zhengzhou University, Zhengzhou China and Technical Report, Nanyang Technological University, Singapore, January 2013.

[5] J. J. Liang, B-Y. Qu, P. N. Suganthan, "Problem Definitions and Evaluation Criteria for the CEC 2014 Special Session and Competition on Single Objective Real-Parameter Numerical Optimization",Technical Report201311,Computational Intelligence Laboratory, Zhengzhou University, Zhengzhou China and Technical Report, Nanyang Technological University, Singapore, December 2013

[6] Joaquin Derrac, Salvador Garcia, Sheldon Hui, Francisco Herrera, Ponnuthurai N. Suganthan, "Statistical analysis of convergence performance throughout the search: A case study with SaDE-MMTS and Sa-EPSDE-MMTS," IEEE Symp. On DE 2013, IEEE SSCI2013, Singapore.

[7] Nikolaus Hansen, Steffen Finck, Raymond Ros and Anne Auger, "Real-Parameter Black-Box Optimization Benchmarking 2010: Noiseless Functions Definitions" INRIA research report RR-6829,March 24,2012.

[8] Xiaodong Li, Ke Tang, Mohammad N. Omidvar, Zhenyu Yang, and Kai Qin, Benchmark Functions for the CEC'2013 Special Session and Competition on Large-Scale Global Optimization, Technical Report,2013