Problem Definitions and Evaluation Criteria for CEC 2015 Special Session on Computationally Expensive Single Objective Optimization

Q. Chen[1], B. Liu[2] and Q. Zhang[3], J. J. Liang[4], P. N. Suganthan[5], B. Y. Qu[6]

[1] Facility Design and Instrument Institute, China Aerodynamic Research and Development Center, China
[2] Department of Computing, Glyndwr University, UK
[3] Department of Computer Science, City University of Hong Kong, Hong Kong & School of Computer Science and Electronic Engineering, University of Essex, UK.
[4] School of Electrical Engineering, Zhengzhou University, Zhengzhou, China
[5] School of EEE, Nanyang Technological University, Singapore
[6] School of Electric and Information Engineering, Zhongyuan University of Technology, Zhengzhou, China

b.liu@glyndwr.ac.uk, qingfu.zhang@cityu.edu.hk, chenqin1980@gmail.com
liangjing@zzu.edu.cn, epnsugan@e.ntu.edu.sg, qby1984@hotmail.com

Many real-world optimization problems require computationally expensive computer or physical simulations for evaluating their candidate solutions. For practical reasons like computation resource constraints and project time requirements, optimization algorithms specialized for computationally expensive problems are essential for the success in industrial design. Often, canonical evolutionary algorithms (EA) cannot directly solve them since a large number of function evaluations are required which is unaffordable in many cases. In recent years, various kinds of novel methods for computationally expensive optimization problems have been proposed and good results with only limited function evaluations have been published in literatures.

To promote research on expensive optimization, we successfully organized a competition focusing on small- to medium-scale (from 10 to 30 decision variables) real parameter bound constrained single-objective computationally expensive optimization within CEC 2014. For 8 functions with 10/20/30 decision variables, 6 valid results were collected and compared. In the special session of CEC2014 on computationally expensive optimization sector, we exchanged ideas and visions. These encourage us make further effort on this direction. This year, we choose 15 new benchmark problems and propose a more challenging competition within CEC 2015, which include composite problems and hybrid problems introduced by J.J. Liang and P.N. Suganthan with more features.

We encourage all participants to test their algorithms on the 15 black-box benchmark functions with 10 and 30 dimensions. The participants are required to send the final results in the format given in this technical report to the organizers. The organizers will conduct an overall analysis and comparison. The participants with the best results should also be willing to release their codes for verification before declaring the eventual winners of the competition.

The JAVA, C and Matlab codes for CEC'15 test suite can be downloaded from the website given below:

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

1. Introduction of the CEC'15 expensive optimization test problems

1.1 Common definitions

All test functions are minimization problems defined as following:

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

where D is the dimension of the problem, $\boldsymbol{o}_{i1} = [o_{i1}, o_{i2}, ..., o_{iD}]^{\mathrm{T}}$ is 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'15. All test functions are shifted to $\boldsymbol{o}$ and scalable. For convenience, the same search ranges are defined for all test functions as $[-100,100]^{D}$.

Considering that in the real-world problems, it is seldom that there exist linkages among all variables. The decision variables are divided into subcomponents randomly. The rotation matrix for each sub-components are generated from standard normally distributed entries by Gram-Schmidt ortho-normalization with condition number c that is equal to 1 or 2.

1.2 Summary of CEC'15 expensive optimization test problems

Table I. Summary of the CEC' 15 expensive optimization test problems

Categories	No	Functions	Related basic functions	F_i^*
Unimodal functions	1	Rotated Bent Cigar Function	Bent Cigar Function	100
	2	Rotated Discus Function	Discus Function	200
Simple Multimodal functions	3	Shifted and Rotated Weierstrass Function	Weierstrass Function	300
	4	Shifted and Rotated Schwefel's Function	Schwefel's Function	400
	5	Shifted and Rotated Katsuura Function	Katsuura Function	500
	6	Shifted and Rotated HappyCat Function	HappyCat Function	600
	7	Shifted and Rotated HGBat Function	HGBat Function	700
	8	Shifted and Rotated Expanded Griewank's plus Rosenbrock's Function	Griewank's Function Rosenbrock's Function	800
	9	Shifted and Rotated Expanded Scaffer's F6 Function	Expanded Scaffer's F6 Function	900
Hybrid funtions	10	Hybrid Function 1 (*N*=3)	Schwefel's Function Rastrigin's Function High Conditioned Elliptic Function	1000
	11	Hybrid Function 2 (*N*=4)	Griewank's Function Weierstrass Function Rosenbrock's Function Scaffer's F6 Function	1100
	12	Hybrid Function 3 (*N*=5)	Katsuura Function HappyCat Function Griewank's Function Rosenbrock's Function Schwefel's Function Ackley's Function	1200
Composition functions	13	Composition Function 1 (N=5)	Rosenbrock's Function High Conditioned Elliptic Function Bent Cigar Function Discus Function High Conditioned Elliptic Function	1300
	14	Composition Function 2 (*N*=3)	Schwefel's Function Rastrigin's Function High Conditioned Elliptic Function	1400
	15	Composition Function 3 (*N*=5)	HGBat Function Rastrigin's Function Schwefel's Function Weierstrass Function High Conditioned Elliptic Function	1500

Please notice: These problems should be treated as black-box optimization problems and without any prior knowledge. Neither the analytical equations nor the problem landscape characters extracted from analytical equations are allowed to be used. However, the dimensionality and the number of available function evaluations can be considered as known values and can be used.

1.2 Definitions of basic functions

1) Bent Cigar Function

$$f_1(\boldsymbol{x}) = x_1^2 + 10^6 \sum_{i=2}^{D} x_i^2 \tag{1}$$

2) Discus Function

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

3) Weierstrass Function

$$f_3(\boldsymbol{x}) = \sum_{i=1}^{D} \left(\sum_{k=0}^{k\max} [a^k \cos(2\pi b^k (x_i + 0.5))] \right) - D \sum_{k=0}^{k\max} [a^k \cos(2\pi b^k \cdot 0.5)] \tag{3}$$

where a=0.5, b=3, and kmax=20.

4) Modified Schwefel's Function

$$f_4(\boldsymbol{x}) = 418.9829 \times D - \sum_{i=1}^{D} g(z_i), \qquad z_i = x_i + 4.209687462275036\text{e+002}$$

$$g(z_i) = \begin{cases} z_i \sin(|z_i|^{1/2}) & \text{if } |z_i| \le 500 \\ (500 - \operatorname{mod}(z_i, 500)) \sin\left(\sqrt{|500 - \operatorname{mod}(z_i, 500)|}\right) - \dfrac{(z_i - 500)^2}{10000D} & \text{if } z_i > 500 \\ (\operatorname{mod}(|z_i|, 500) - 500) \sin\left(\sqrt{|\operatorname{mod}(|z_i|, 500) - 500|}\right) - \dfrac{(z_i + 500)^2}{10000D} & \text{if } z_i < -500 \end{cases} \tag{4}$$

5) Katsuura Function

$$f_5(\boldsymbol{x}) = \frac{10}{D^2} \prod_{i=1}^{D} \left(1 + i \sum_{j=1}^{32} \frac{|2^j x_i - round(2^j x_i)|}{2^j}\right)^{\frac{10}{D^{1.2}}} - \frac{10}{D^2} \tag{5}$$

6) HappyCat Function

$$f_6(\boldsymbol{x}) = \left| \sum_{i=1}^{D} x_i^2 - D \right|^{1/4} + \left(0.5 \sum_{i=1}^{D} x_i^2 + \sum_{i=1}^{D} x_i\right) / D + 0.5 \tag{6}$$

7) HGBat Function

$$f_7(\boldsymbol{x}) = \left| \left(\sum_{i=1}^{D} x_i^2\right)^2 - \left(\sum_{i=1}^{D} x_i\right)^2 \right|^{1/2} + \left(0.5 \sum_{i=1}^{D} x_i^2 + \sum_{i=1}^{D} x_i\right) / D + 0.5 \tag{7}$$

8) Expanded Griewank's plus Rosenbrock's Function

$$f_8(\boldsymbol{x}) = f_{11}(f_{10}(x_1, x_2)) + f_{11}(f_{10}(x_2, x_3)) + \ldots + f_{11}(f_{10}(x_{D-1}, x_D)) + f_{11}(f_{10}(x_D, x_1)) \tag{8}$$

9) Expanded Scaffer's F6 Function

Scaffer's F6 Function:

$$g(x, y) = 0.5 + \frac{(\sin^2(\sqrt{x^2 + y^2}) - 0.5)}{(1 + 0.001(x^2 + y^2))^2}$$

$$f_9(\boldsymbol{x}) = g(x_1, x_2) + g(x_2, x_3) + \ldots + g(x_{D-1}, x_D) + g(x_D, x_1) \tag{9}$$

10) Rosenbrock's Function

$$f_{10}(\boldsymbol{x}) = \sum_{i=1}^{D-1}(100(x_i^2 - x_{i+1})^2 + (x_i - 1)^2) \tag{10}$$

11) Griewank's Function

$$f_{11}(\boldsymbol{x}) = \sum_{i=1}^{D} \frac{x_i^{\,2}}{4000} - \prod_{i=1}^{D} \cos(\frac{x_i}{\sqrt{i}}) + 1 \tag{11}$$

12) Rastrigin's Function

$$f_{12}(\boldsymbol{x}) = \sum_{i=1}^{D} (x_i^{\,2} - 10\cos(2\pi x_i) + 10) \tag{12}$$

13) High Conditioned Elliptic Function

$$f_{13}(\boldsymbol{x}) = \sum_{i=1}^{D} (10^6)^{\frac{i-1}{D-1}} \boldsymbol{x}_i^2 \tag{13}$$

14) Ackley's Function

$$f_{14}(\boldsymbol{x}) = -20\exp(-0.2\sqrt{\frac{1}{D}\sum_{i=1}^{D} x_i^2}) - \exp(\frac{1}{D}\sum_{i=1}^{D}\cos(2\pi x_i)) + 20 + e \tag{14}$$

2. Definitions of the CEC'14 Test Suite

2.1 Unimodal Functions

1) Rotated Bent Cigar Function

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

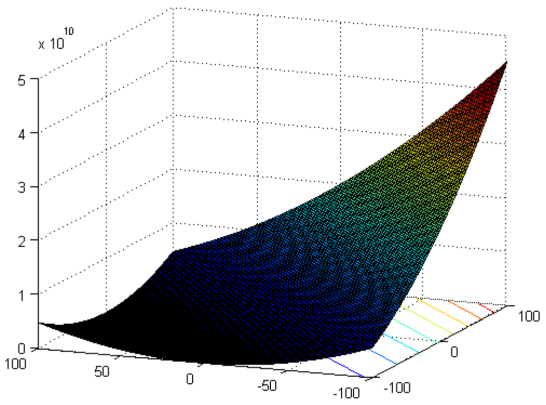

Figure 1. 3-*D* map for 2-*D* function

Properties:

- Unimodal
- Non-separable
- Smooth but narrow ridge

2) Rotated Discus Function

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

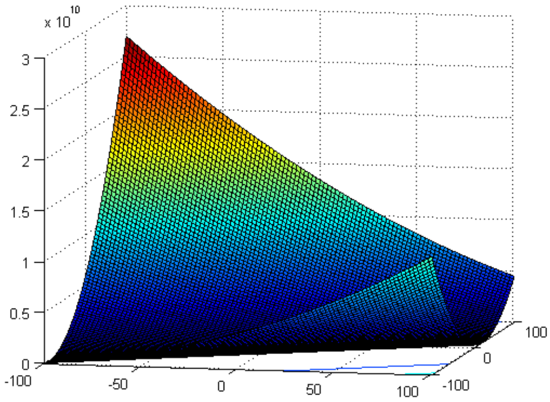

Figure 2. *3-D map for 2-D function*

Properties:

- Unimodal
- Non-separable
- With one sensitive direction

2.2 Simple Multimodal Functions

3) Shifted and Rotated Weierstrass Function

$$F_3(\boldsymbol{x}) = f_3(\mathbf{M}(\frac{0.5(\boldsymbol{x}-\boldsymbol{o}_3)}{100})) + F_3{}^* \quad \textbf{(17)}$$

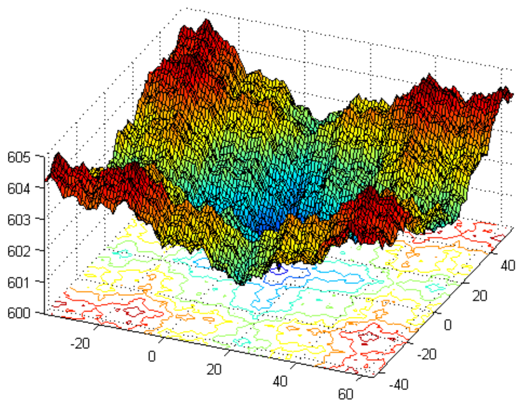

Figure 3. *3-D map for 2-D function*

Properties:

- Multi-modal
- Non-separable
- Continuous but differentiable only on a set of points

4) Shifted and Rotated Schwefel's Function

$$F_4(\boldsymbol{x}) = f_4(\mathbf{M}(\frac{1000(\boldsymbol{x}-\boldsymbol{o}_4)}{100})) + F_4{}^* \quad \textbf{(18)}$$

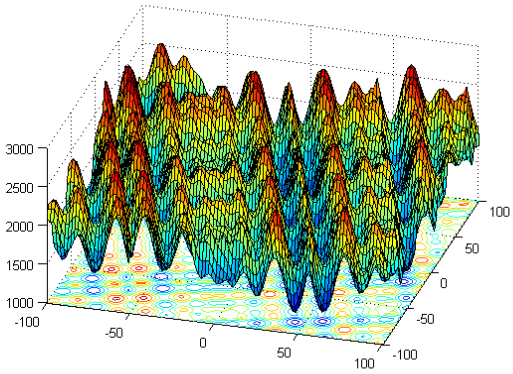

Figure 4(a). *3-D map for 2-D function*

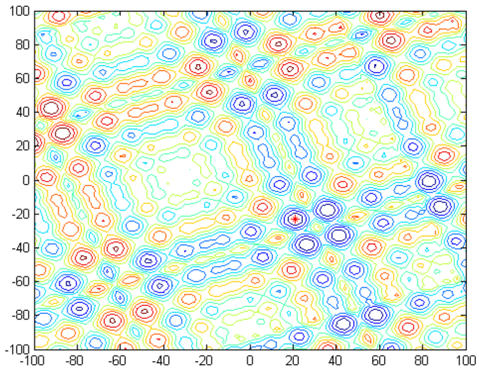

Figure 4(b).*Contour map for 2-D function*

Properties:

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

5) Shifted and Rotated Katsuura Function

$$F_5(\boldsymbol{x}) = f_5(\mathbf{M}(\frac{5(\boldsymbol{x}-\boldsymbol{o}_5)}{100})) + F_5{}^* \tag{19}$$

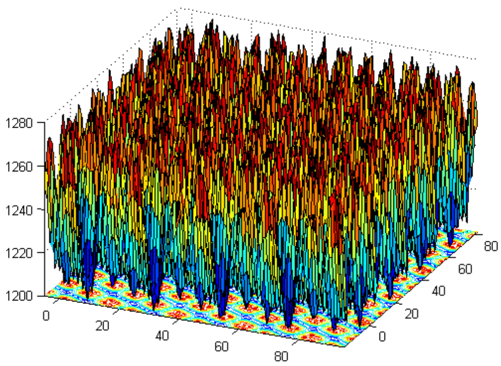

Figure 5(a). *3-D map for 2-D function*

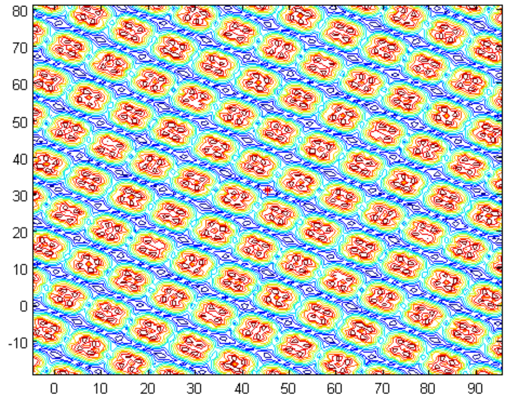

Figure 5(b). *Contour map for 2-D function*

Properties:

- Multi-modal
- Non-separable
- Continuous everywhere yet differentiable nowhere

6) Shifted and Rotated HappyCat Function

$$F_6(\boldsymbol{x}) = f_6(\mathbf{M}(\frac{5(\boldsymbol{x}-\boldsymbol{o}_6)}{100})) + F_6{}^* \quad \textbf{(20)}$$

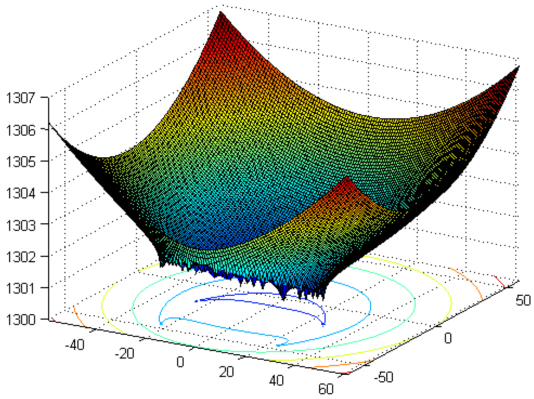

Figure 6(a). *3-D map for 2-D function*

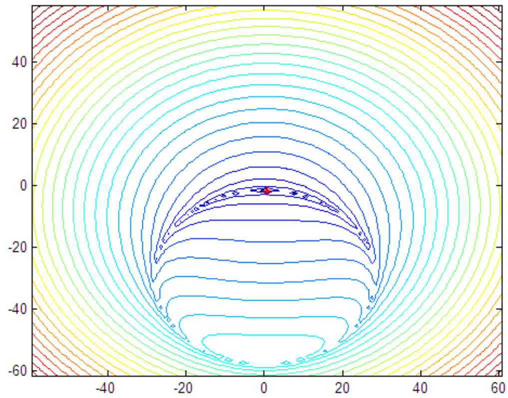

Figure 6(b). *Contour map for 2-D function*

Properties:

- Multi-modal
- Non-separable

7) Shifted and Rotated HGBat Function

$$F_7(\boldsymbol{x}) = f_7(\mathbf{M}(\frac{5(\boldsymbol{x}-\boldsymbol{o}_7)}{100})) + F_7{}^* \tag{21}$$

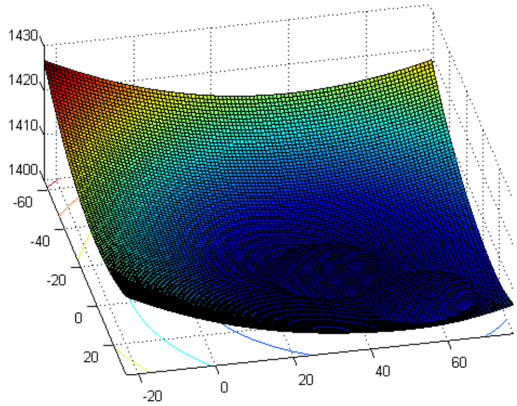

Figure 7(a). *3-D map for 2-D function*

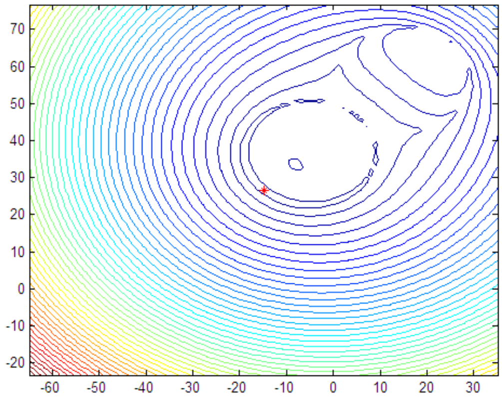

Figure 7(b). *Contour map for 2-D function*

Properties:

- Multi-modal
- Non-separable

8) Shifted and Rotated Expanded Griewank's plus Rosenbrock's Function

$$F_8(\boldsymbol{x}) = f_8(\mathbf{M}(\frac{5(\boldsymbol{x}-\boldsymbol{o}_8)}{100}) + 1) + F_8{}^* \tag{22}$$

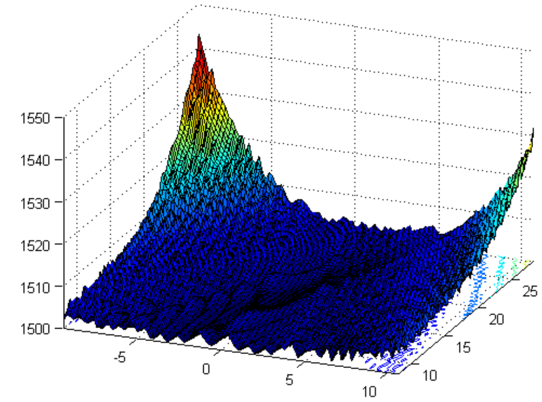

Figure 8(a). *3-D map for 2-D function*

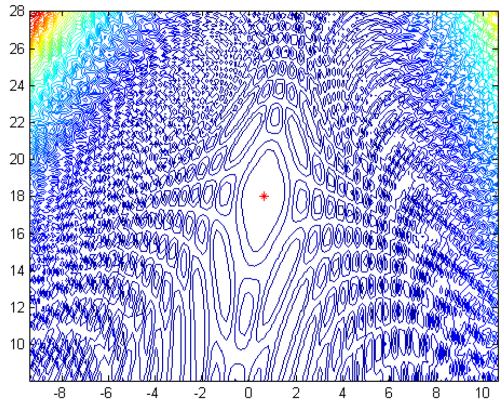

Figure 8(b).*Contour map for 2-D function*

Properties:

- Multi-modal
- Non-separable

9) Shifted and Rotated Expanded Scaffer's F6 Function

$$F_9(\boldsymbol{x}) = f_9(\mathbf{M}(\boldsymbol{x} - \boldsymbol{o}_9) + 1) + F_9 * \qquad \textbf{(23)}$$

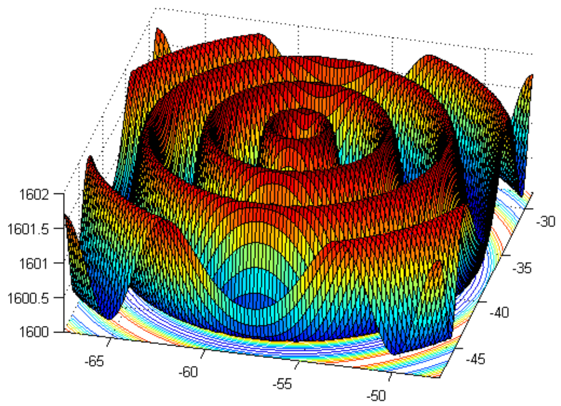

Figure 9. *3-D map for 2-D function*

Properties:

- Multi-modal
- Non-separable

2.3 Hybrid Functions

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

$$F(\boldsymbol{x}) = g_1(\mathbf{M}_1\boldsymbol{z}_1) + g_2(\mathbf{M}_2\boldsymbol{z}_2) + ... + g_N(\mathbf{M}_N\boldsymbol{z}_N) + F^*(\boldsymbol{x}) \quad \textbf{(24)}$$

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

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

$$\boldsymbol{z}_1 = [\boldsymbol{y}_{S_1}, \boldsymbol{y}_{S_2}, ..., \boldsymbol{y}_{S_{n_1}}], \boldsymbol{z}_2 = [\boldsymbol{y}_{S_{n_1+1}}, \boldsymbol{y}_{S_{n_1+2}}, ..., \boldsymbol{y}_{S_{n_1+n_2}}], ..., \boldsymbol{z}_N = [\boldsymbol{y}_{S_{\sum_{i=1}^{N-1} n_i+1}}, \boldsymbol{y}_{S_{\sum_{k=1}^{N-1} n_i+2}}, ..., \boldsymbol{y}_{S_D}] \quad \textbf{(25)}$$

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

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

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

$$n_1 = \lceil p_1 D \rceil, n_2 = \lceil p_2 D \rceil, ..., n_{N-1} = \lceil p_{N-1} D \rceil, n_N = D - \sum_{i=1}^{N-1} n_i \quad \textbf{(26)}$$

10) Hybrid Function 1 (N=3)

p = [0.3, 0.3, 0.4]
g_1: Modified Schwefel's Function f_4
g_2: Rastrigin’s Function f_{12}
g_3: High Conditioned Elliptic Function f_{13}

11) Hybrid Function 2 (N=4)

p = [0.2, 0.2, 0.3, 0.3]
g_1: Griewank’s Function f_{11}
g_2: Weierstrass Function f_3
g_3: Rosenbrock’s Function f_{10}
g_4: Scaffer’s F6 Function f_9

12) Hybrid Function 3 (N=5)

p = [0.1, 0.2, 0.2, 0.2, 0.3]
g_1: Katsuura Function f_5
g_2: HappyCat Function f_6
g_3: Expanded Griewank’s plus Rosenbrock’s Function f_8

g_4: Modified Schwefel's Function f_4
g_5: Ackley's Function f_{14}

2.4 Composite Functions

$$F(\mathbf{x}) = \sum_{i=1}^{N} \{\omega_i * [\lambda_i g_i(\mathbf{x}) + bias_i]\} + f^* \qquad \textbf{(27)}$$

$F(\mathbf{x})$: composition function
$g_i(\mathbf{x})$: i^{th} basic function used to construct the composition function
N: number of basic functions
o_i: new shifted optimum position for each $g_i(\mathbf{x})$, define the global and local optima's position
$bias_i$: defines which optimum is global optimum
σ_i: used to control each $g_i(\mathbf{x})$'s coverage range, a small σ_i give a narrow range for that $g_i(x)$
λ_i: used to control each $g_i(\mathbf{x})$'s height
w_i: weight value for each $g_i(\mathbf{x})$, calculated as below:

$$w_i = \frac{1}{\sqrt{\sum_{j=1}^{D}(x_j - o_{ij})^2}} \exp\left(-\frac{\sum_{j=1}^{D}(x_j - o_{ij})^2}{2D\sigma_i^2}\right) \qquad \textbf{(28)}$$

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

So when $\mathbf{x} = \mathbf{o}_i$, $\omega_j = \begin{cases} 1 & j = i \\ 0 & j \neq i \end{cases}$ 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.

13) Composition Function 1 (N=5)

N= 5, σ = [10, 20, 30, 40, 50]
λ = [1, 1e-6, 1e-26, 1e-6, 1e-6]
$bias$ = [0, 100, 200, 300, 400]
g_1: Rotated Rosenbrock's Function f_{10}
g_2: High Conditioned Elliptic Function f_{13}
g_3: Rotated Bent Cigar Function f_1
g_4: Rotated Discus Function f_2
g_5: High Conditioned Elliptic Function f_{13}

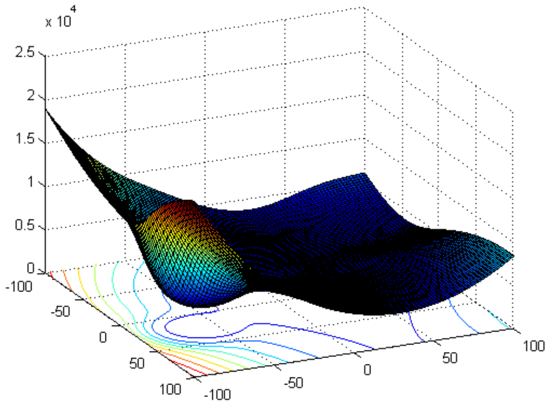

Figure 10(a). *3-D map for 2-D function*

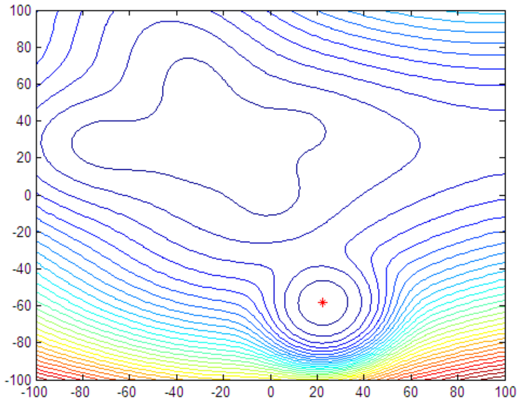

Figure 10 (b).*Contour map for 2-D function*

Properties:

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

14) Composition Function 2 (N=3)

N = 3
σ = [10, 30, 50]
λ = [0.25, 1, 1e-7]
bias = [0, 100, 200]
g_1: Rotated Schwefel's Function f_4
g_2: Rotated Rastrigin's Function f_{12}
g_3: Rotated High Conditioned Elliptic Function f_{13}

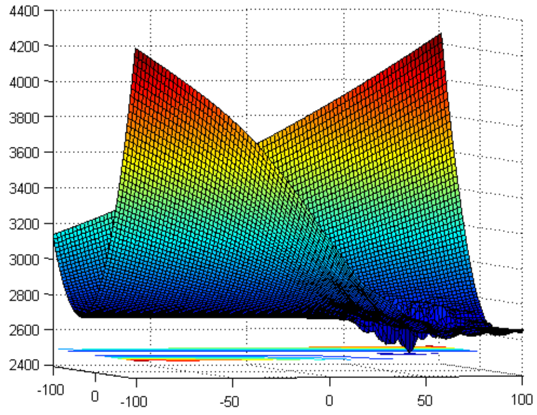

Figure 11(a). *3-D map for 2-D function*

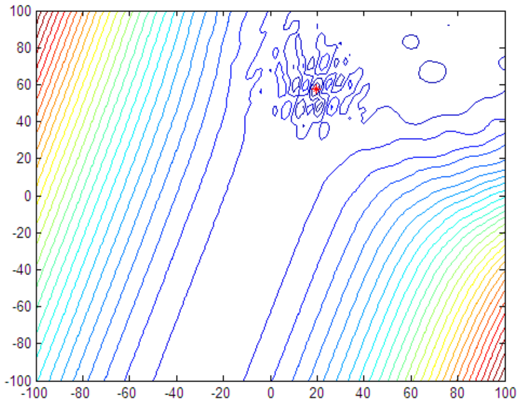

Figure 11(b).*Contour map for 2-D function*

Properties:

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

15) Composition Function 3 (N=5)

N = 5
σ = [10, 10, 10, 20, 20]
λ = [10, 10, 2.5, 25, 1e-6]
bias = [0, 100, 200, 300, 400]
g_1: Rotated HGBat Function f_7
g_2: Rotated Rastrigin's Function f_{12}
g_3: Rotated Schwefel's Function f_4
g_4: Rotated Weierstrass Function f_3
g_5: Rotated High Conditioned Elliptic Function f_{13}

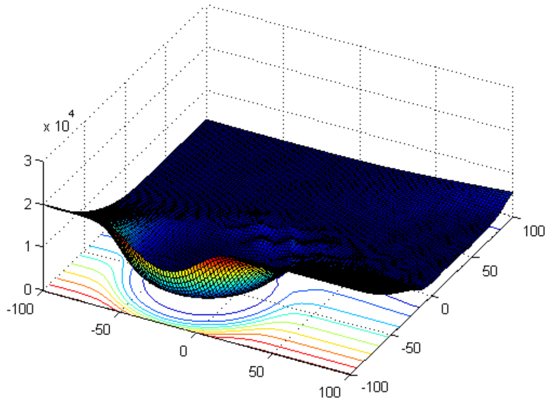

Figure 12(a). *3-D map for 2-D function*

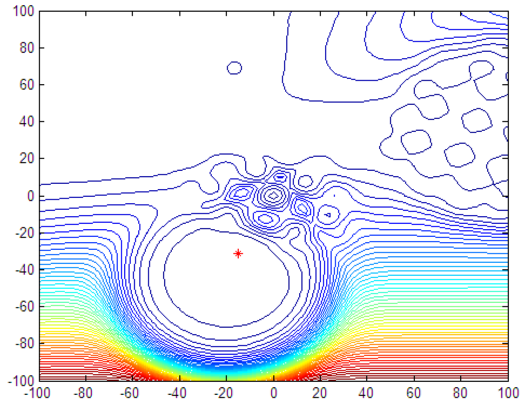

Figure 12(b). *Contour map for 2-D function*

Properties:

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

3. Evaluation criteria

3.1 Experimental setting:

- Number of independent runs: 20
- Maximum number of exact function evaluations:
  - o 10-dimensional problems: 500
  - o 30-dimensional problems: 1,500
- Initialization: Any problem-independent initialization method is allowed.
- 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.
- Termination: Terminate when reaching the maximum number of exact function evaluations or the error value ($F_i^* - F_i(x^*)$) is smaller than 10^{-3}.

3.2 Results to record:

(1) Current best function values:

Record current best function values using $0.1\times\mathbf{MaxFES}$, $0.2\times\mathbf{MaxFES}$, …, $\mathbf{MaxFES}$ for each run. Sort the obtained best function values after the maximum number of exact function evaluations from the smallest (best) to the largest (worst) and present the best, worst, mean, median and standard deviation values for the 20 runs. Error values smaller than 10^{-8} are taken as zero.

(2) Algorithm complexity:

For expensive optimization, the criterion to judge the efficiency is the obtained best result vs. number of exact function evaluations. But the computational overhead on surrogate modeling and search is also considered as a secondary evaluation criterion. Considering that for different data sets, the computational overhead for a surrogate modeling method can be quite different, the computational overhead of each problem is necessary to be reported. Often, compared to the computational cost on surrogate modeling, the cost on 500, 1000 and 1500 function evaluations can almost be ignored. Hence, the following method is used:

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) The average complete computing time for the algorithm = $\widehat{T}1$

The complexity of the algorithm is measured by: $\widehat{T}1/T0$.

(3) Parameters:

Participants are requested not to search for the best distinct set of parameters for each problem/dimension/etc. 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.

(4) Encoding

If the algorithm requires encoding, then the encoding scheme should be independent of the specific problems and governed by generic factors such as the search ranges, dimensionality of the problems, etc.

(5) 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_expensive.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_expensive.txt”.

The txt document should contain the mean and median values of current best function values when $0.1\times\mathbf{MaxFES}$, $0.2\times\mathbf{MaxFES}$, …, $\mathbf{MaxFES}$ are used of all the 20 runs. The participant can save the results in the matrix shown in Table II and extracts the mean and median values.

Table II Information matrix for function X

	$0.1\times\mathbf{MaxFES}$	$0.2\times\mathbf{MaxFES}$	…	**MaxFES**
Run 1				
Run 2				
…				
Run 20				

Notice: All participants are allowed to improve their algorithms further after submitting the initial version of their papers to CEC2015. 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. Considering the surrogate modeling for 30 dimensional functions is often time consuming, especially for MATLAB users, results using 10 runs are requested for initial submission.

3.3 Results template

Language: Matlab 2008a

Algorithm: Surrogate model assisted evolutionary algorithm A

Results
Notice:
Considering the length limit of the paper, only Error Values Achieved with MaxFES are need to be listed.

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

…

Algorithm Complexity

Table V. Computational Complexity

Func.	$\hat{T}1 / T0$
1	
2	
3	
…	
14	
15	

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

3.4 Sorting method

In CEC 2014, the mean and median values at the maximum allowed number of evaluations were used to score algorithms. For each problem, the algorithm with the best result scored 9, the second best scored 6, the third best scored 3 and all the others score 0.

$$\text{Total score} = \sum_{i=1}^{24} score_i \text{ (using mean value)} + \sum_{i=1}^{24} score_i \text{ (using median value)}$$

This scoring favours those algorithms which get better results for relatively simpler problems. While for computationally expensive optimisation, it's more important to get acceptable results for complicate problems like real-world multimodal cases. It has been proposed that we directly sort the sum of all mean values of 15 problems for two dimensions (10/30). Thus, in this competition, mean values and median values obtained by each algorithm on all 15 problems for 10/30 dimension will be summed up as the final score of the algorithm.

$$\text{Total score} = \sum_{i=1}^{15} mean(\mathrm{f}^*)\Big|_{D=10} + \sum_{i=1}^{15} mean(\mathrm{f}^*)\Big|_{D=30} + \sum_{i=1}^{15} median(\mathrm{f}^*)\Big|_{D=10} + \sum_{i=1}^{15} median(\mathrm{f}^*)\Big|_{D=30}$$

Special attention will be paid to which algorithm has advantages on which kind of problems, considering dimensionality and problem characteristics.