Swarm and Evolutionary Computation 26 (2016) 23–34

Contents lists available at ScienceDirect

Swarm and Evolutionary Computation

journal homepage: www.elsevier.com/locate/swevo

Regular Paper

Novel benchmark functions for continuous multimodal optimization with comparative results

B.Y. Qu [a,c], J.J. Liang [a,b,*], Z.Y. Wang [c], Q. Chen [d], P.N. Suganthan [e]

[a] School of Electrical and Information Engineering, Zhongyuan University of Technology, Zhengzhou 450007, China
[b] School of Electrical Engineering, Zhengzhou University, Zhengzhou 450001, China
[c] School of Information Engineering, Zhengzhou University, Zhengzhou 450001, China
[d] Facility Design and Instrument Institute, China Aerodynamic Research and Development Center, Mianyang, China
[e] School of Electrical and Electronic Engineering, Nanyang Technological University, Singapore 639798, Singapore

ARTICLE INFO

Article history:
Received 27 January 2015
Received in revised form
2 May 2015
Accepted 23 July 2015
Available online 1 August 2015

Keywords:
Multimodal optimization
Niching
Numerical optimization
Benchmark problems

ABSTRACT

Multi-modal optimization is concerned with locating multiple optima in one single run. Finding multiple solutions to a multi-modal optimization problem is especially useful in engineering, as the best solution may not always be the best realizable due to various practical constraints. To compare the performances of multi-modal optimization algorithms, multi-modal benchmark problems are always required. In this paper, 15 novel scalable multi-modal and real parameter benchmark problems are proposed. Among these 15 problems, 8 are extended simple functions while the rest are composition functions. These functions coordinate rotation and shift operations to create linkage among different dimensions and to place the optima at different locations, respectively. Four typical niching algorithms are used to solve the proposed problems. As shown by the experimental results, the proposed problems are challenging to these four recent algorithms.

1. Introduction

In real world optimization, many problems often contain multiple global/local optima. Multimodal optimization aims to find a number of global or good local optima so that the user is able to select the most appropriate one among these potential solutions. Multiple solutions could also be analyzed to discover hidden properties or relationships of the concerned functional landscape [1,23]. In recent decades, Evolutionary Algorithms (EAs) have become a common choice for solving multimodal problems due to their use of a population, which allows multiple solutions to be searched simultaneously in one execution [2,3].

Detection and maintenance of multiple solutions are two challenging tasks for the EAs while solving multimodal optimization problems as EAs are originally designed to solve single-peak global optimization problems. Various techniques that commonly referred to as niching methods are incorporated in the original EAs to make them suitable for solving multimodal optimization problems [4–15]. Niching is a technique of finding and preserving multiple stable subpopulations or niches and preventing the population from converging to a single point. The primary objective of any niching technique is to preserve the diversity of the population. The techniques for multimodal optimization are usually based on diversity maintenance techniques borrowed from other domains [16]. Many niching methods have been developed in the past few years, including crowding [8], fitness sharing [10], restricted tournament selection [9], speciation [11], and geographical neighborhood-based mutation [6]. Crowding [8] technique is inspired by the competition among similar individuals in a natural population. In order to maintain the diversity of the population, it compares the offspring with several randomly sampled individuals from the current population. Restricted Tournament Selection [10] is similar to crowding. It selects a random sample of '*w*' (window size) individuals from the population and compares the offspring with the closest one within the sampled individuals. In fitness sharing [9], the population is partitioned into different subgroups according to the similarity of the individuals and individuals share their information within the same subgroup. Speciation [11] divides the population into different species based on their similarity and the size of the specie is determined by a user specified value. Geographical neighborhood-based mutation [6,22] restricts the production of offspring as well as the competition within a local area (most often a compact Euclidean ball surrounding the current parent) which can maintain the diversity of the population and improve independent multiple local convergences.

Multimodal benchmark test functions are needed to comparatively assess the effectiveness of a newly proposed niching technique. It is noticeable that most of the commonly used test functions are relatively too simple to meaningfully compare high

* Corresponding author. Tel.: +86 13526781788.
E-mail address: liangjing@zzu.edu.cn (J.J. Liang).

http://dx.doi.org/10.1016/j.swevo.2015.07.003
2210-6502/

performing algorithms. The dimensions of these problems are small and non-scalable as well as the positions of the optima are also mostly periodic without linkages among the decision variables. Although some scalable test functions were proposed in literature [17], they do not contain different rotations for different subset of decision variables. Without these operations, the test functions can be biased to certain algorithms. For these reasons, this paper constructs 15 novel scalable multi-modal benchmark functions. The characteristics of these functions can be changed independently by using rotation and shift operations. The linkages among different dimensions are also created by these operations. The guidelines for designing these functions are listed as follows:

1. The functions should be constructed or obtained by using commonly used simple functions.
2. The functions should be scalable. (As the dimensions increase, the number of optima also increases which will pose extra difficulties).
3. Rotation and shift operations should be included in the problems.
4. The positions of optima should be changeable.

As an illustration, four recent niching algorithms are selected to demonstrate the levels of difficulties of these benchmark functions. The remainder of this paper is organized as follows. Section 2 presents the proposed benchmark functions. Section 3 introduces the optimization methods used in the experimental part. Experimental setup and experimental results are presented in Sections 4 and 5, respectively. Section 6 concludes the paper.

2. Novel test functions

This section introduces the novel multimodal benchmark test functions proposed in this paper. (Note that the sources codes of these functions can be downloaded from http://www3.ntu.edu.sg/home/epnsugan/) The Contour maps of these novel test functions are presented in Figs. 1–15. These functions are divided into two categories as extended simple functions and composition functions. Simple functions are constructed by extending some commonly used multimodal problems while composition functions are formed by combining different basic functions [17–20]. All test functions are minimization problems defined as follows:

$$\text{Min } F(\boldsymbol{x}),\ \boldsymbol{x}=[x_1,x_2,\ldots,x_D]^T$$

where D is the dimensions of the function. For these test functions, the following notations are used. $\boldsymbol{o}_{i1}=[o_{i1},o_{i2},\ldots,o_{iD}]^T$: the shifted global optimum, $\mathbf{M}_i$: rotation matrix. $F*$: user specified optima value.

The shifted global optima are randomly distributed in $[-80,80]^D$. All test functions are shifted to o and scalable. For convenience, the same search ranges are used for all test functions as $[-100,100]^D$. Different rotation matrixes are assigned for each function. The variables are divided into subcomponents randomly. The rotation matrixes for each subcomponent are generated from standard normally distributed entries by Gram–Schmidt orthonormalization with condition number c that is equal to 1.

2.1. Extended simple functions

1. Expanded Two-Peak Trap

$$f_1(\boldsymbol{x})=\sum_{i=1}^{D} t_i+200D$$

$$t_i=\begin{cases} -160+y_i^2, & \text{if } y_i<0\\ \frac{160}{15}(y_i-15), & \text{if } 0\le y_i\le 15\\ \frac{200}{5}(15-y_i), & \text{if } 15\le y_i\le 20\\ -200+(y_i-20)^2, & \text{if } y_i>20 \end{cases}$$

$$\boldsymbol{y}=\boldsymbol{x}+20$$

$$f_1(x^*)=0,\quad x^*=[0,0,\ldots,0]^D$$

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

2. Expanded Five-Uneven-Peak Trap

$$f_2(\boldsymbol{x})=\sum_{i=1}^{D} t_i+200D$$

$$t_i=\begin{cases} -200+x_i^2, & \text{if } x_i<0\\ -80(2.5-x_i), & \text{if } 0\le x_i<2.5\\ -64(x_i-2.5), & \text{if } 2.5\le x_i<5\\ -64(7.5-x_i), & \text{if } 5\le x_i<7.5\\ -28(x_i-7.5), & \text{if } 7.5\le x_i<12.5\\ -28(17.5-x_i), & \text{if } 12.5\le x_i<17.5\\ -32(x_i-17.5), & \text{if } 17.5\le x_i<22.5\\ -32(27.5-x_i), & \text{if } 22.5\le x_i<27.5\\ -80(x_i-27.5), & \text{if } 27.5\le x_i\le 30\\ -200+(x_i-30)^2, & \text{if } x_i>30 \end{cases}$$

$$f_2(x^*)=0,\quad \boldsymbol{x}_i^*=0 \text{ or } 30 \text{ for } i=1,2,\ldots,D$$

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

3. Expanded Equal Minima

$$f_3(\boldsymbol{x})=\sum_{i=1}^{D} t_i+D$$

$$t_i=\begin{cases} y_i^2, & \text{if } y_i<0 \text{or} y_i>1\\ -\sin^6(5\pi y_i), & \text{if } 0\le y_i\le 1 \end{cases} \quad i=1,2,\ldots,D$$

$$\boldsymbol{y}=\boldsymbol{x}+0.1$$

$$f_3(x^*)=0,\quad \boldsymbol{x}_i^*=0.0,\ 0.2,\ 0.4,\ 0.6 \text{ or } 0.8 \text{ for} i=1,2,\ldots,D$$

$$F_3(\boldsymbol{x})=f_3\left(\mathbf{M}_3\left(\frac{\boldsymbol{x}-\boldsymbol{o}_3}{20}\right)\right)+F_3^*$$

4. Expanded Decreasing Minima

$$f_4(\boldsymbol{x})=\sum_{i=1}^{D} t_i+D$$

$$t_i=\begin{cases} y_i^2, & \text{if } y_i<0 \text{ or } y_i>1\\ -\exp\left[-2\log(2)\cdot\left(\frac{y_i-0.1}{0.8}\right)^2\right]\cdot\sin^6(5\pi y_i), & \text{if } 0\le y_i\le 1 \end{cases}$$

$$i=1,2,\ldots,D$$

$$\boldsymbol{y}=\boldsymbol{x}+0.1$$

$$f_4(x^*)=0,\quad x^*=[0,0,\ldots,0]^D$$

$$F_4(\boldsymbol{x})=f_4\left(\mathbf{M}_4\left(\frac{\boldsymbol{x}-\boldsymbol{o}_4}{20}\right)\right)+F_4^*$$

5. Expanded Uneven Minima

$$f_5(\boldsymbol{x})=\sum_{i=1}^{D} t_i-D$$

$$t_i=\begin{cases} y_i^2, \\ -\sin^6(5\pi(y_i^{3/4}-0.05)), & \text{if } y_i<0 \text{or} y_i>1 \end{cases}$$

$$\text{if } 0\le y_i\le 1$$

$$i=1,2,\ldots,D$$

$$\boldsymbol{y}=\boldsymbol{x}+0.079699392688696$$

$$f_5(x^*)=0,\quad \boldsymbol{x}_i^*=\begin{cases}0\\ \text{or } 0.166955\\ \text{or } 0.370927 \text{ for } i=1,2,\ldots,D\\ \text{or } 0.601720\\ \text{or } 0.854195\end{cases}$$

$$F_5(\boldsymbol{x})=f_5\left(\mathbf{M}_5\left(\frac{\boldsymbol{x}-\boldsymbol{o}_5}{20}\right)\right)+F_5^*$$

6. Expanded Himmelblau's Function

$$f_6(\boldsymbol{x})=\sum_{i=1,3,5,\ldots}^{D-1}\left[\left(y_i{}^2+y_{i+1}-11\right)^2+\left(y_i+y_{i+1}{}^2-7\right)^2\right]$$

$$y_i=\begin{cases}x_i+3, & \text{if } i \text{ is odd number}\\ x_i+2, & \text{if } i \text{ is even number}\end{cases},\quad i=1,2,\ldots,D$$

D must be an even number.

$$f_6(x^*)=0$$

$$x^*=[\boldsymbol{y}_1,\ldots,\boldsymbol{y}_{D/2}]$$

$$\boldsymbol{y}_i=\begin{cases}[0,0]\\ \text{or } [0.584428,-3.848126]\\ \text{or } [-6.779310,-5.283186]\\ \text{or } [-5.805118,1.131312]\end{cases}\text{for } i=1,2,\ldots,\frac{D}{2}$$

$$F_6(\boldsymbol{x})=f_6\left(\mathbf{M}_6\left(\frac{\boldsymbol{x}-\boldsymbol{o}_6}{5}\right)\right)+F_6^*$$

7. Expanded Six-Hump Camel Back

$$f_7(\boldsymbol{x})=\sum_{i=1,3,5,\ldots}^{D-1}\left\{-4[(4-2.1y_i{}^2+\frac{y_i{}^4}{3})y_i{}^2+y_iy_{i+1}+(-4+4y_{i+1}{}^2)y_{i+1}{}^2]\right\}+4.126514*\frac{D}{2}$$

$$y_i=\begin{cases}x_i-0.089842, & \text{if } i \text{ is odd number}\\ x_i+0.712656, & \text{if } i \text{ is even number}\end{cases},\quad i=1,2,\ldots,D$$

D must be an even number

$$f_7(x^*)=0$$

$$x^*=[\boldsymbol{y}_1,\ldots,\boldsymbol{y}_{D/2}]$$

$$\boldsymbol{y}_i=\begin{cases}[0,0]\\ \text{or}[-0.179684,1.425312]\end{cases}\text{for } i=1,2,\ldots,\frac{D}{2}$$

$$F_7(\boldsymbol{x})=f_7(\mathbf{M}_7(\boldsymbol{x}-\boldsymbol{o}_7))+F_7^*$$

8. Modified Vincent Function

$$f_8(\boldsymbol{x})=\frac{1}{D}\sum_{i=1}^{D}(t_i+1.0)$$

$$t_i=\begin{cases}\sin(10\log(y_i)) & \text{if } 0.25\le y_i\le 10\\ (0.25-y_i)^2+\sin(10\log(2.5)) & \text{if } y_i<0.25\\ (y_i-10)^2+\sin(10\log(10)) & \text{if } y_i>10\end{cases},\quad i=1,2,\ldots,D$$

$$\boldsymbol{y}=\boldsymbol{x}+4.1112$$

$$f_8(x^*)=0,\quad \boldsymbol{x}_i^*=\begin{cases}-3.7782\\ \text{or}-3.4870\\ \text{or}-2.9411\\ \text{or}-1.9179\\ \text{or}0\\ \text{or}3.5951\end{cases}\text{for } i=1,2,\ldots,D$$

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

The number of optima for these expanded functions are provided in Table 1.

Table 1
No. of optima for expanded functions.

Function	Type of optima		
	Global optimum	Local optima	Second and third best optima
$f_1(\boldsymbol{x})$ Expanded Two-Peak Trap	1	2^D-1	$C_D^1+C_D^2$ *
$f_2(\boldsymbol{x})$ Five-Uneven-Peak Trap	2^D	5^D-2^D	–
$f_3(\boldsymbol{x})$ Expanded Equal Minima	5^D	0	–
$f_4(\boldsymbol{x})$ Expanded Decreasing Minima	1	5^D-1	$C_D^1+C_D^2$*
$f_5(\boldsymbol{x})$ Expanded Uneven Minima	5^D	0	–
$f_6(\boldsymbol{x})$ Expanded Himmelblau's Function	$4^{D/2}$	0	–
$f_7(\boldsymbol{x})$ Expanded Six-Hump Camel Back	$2^{D/2}$	0	–
$f_8(\boldsymbol{x})$ Modified Vincent Function	6^D	0	–

* $C_m^n=\frac{m!}{n!(m-n)!}$.

2.2. Basic functions used to construct composition problems

The following basic functions are used to construct the composition functions.

(a) Sphere Function

$$f_9(\boldsymbol{x})=\sum_{i=1}^{D}x_i{}^2$$

(b) High Conditioned Elliptic Function

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

(c) Bent Cigar Function

$$f_{11}(\boldsymbol{x})=x_1^2+10^6\sum_{i=2}^{D}x_i^2$$

(d) Discus Function

$$f_{12}(\boldsymbol{x})=10^6x_1^2+\sum_{i=2}^{D}x_i^2$$

(e) Different Powers Function

$$f_{13}(\boldsymbol{x})=\sqrt{\sum_{i=1}^{D}|x_i|^{2+4\frac{i-1}{D-1}}}$$

(f) Rosenbrock's Function

$$f_{14}(\boldsymbol{x})=\sum_{i=1}^{D-1}(100(x_i^2-x_{i+1})^2+(x_i-1)^2)$$

(g) Ackley's Function

$$f_{15}(\boldsymbol{x})=-20\exp(-0.2\sqrt{\frac{1}{D}\sum_{i=1}^{D}x_i^2})-\exp(\frac{1}{D}\sum_{i=1}^{D}\cos(2\pi x_i))+20+e$$

(h) Weierstrass Function

$$f_{16}(\boldsymbol{x})=\sum_{i=1}^{D}\left(\sum_{k=0}^{k\max}[a^k\cos(2\pi b^k(x_i+0.5))]\right)-D\sum_{k=0}^{k\max}[a^k\cos(2\pi b^k\cdot 0.5)]$$
$$a=0.5, b=3, kmax=20$$

(i) Griewank's Function

$$f_{17}(\boldsymbol{x})=\sum_{i=1}^{D}\frac{x_i^2}{4000}-\prod_{i=1}^{D}\cos\left(\frac{x_i}{\sqrt{i}}\right)+1$$

(j) Rastrigin's Function

$$f_{18}(\boldsymbol{x})=\sum_{i=1}^{D}(x_i^2-10\cos(2\pi x_i)+10)$$

(k) Modified Schwefel's Function

$$f_{19}(\boldsymbol{x})=418.9829\times D-\sum_{i=1}^{D}g(z_i),\qquad z_i=x_i+420.9687462275036$$
$$g(z_i)=\begin{cases}z_i\sin(|z_i|^{1/2}) & \text{if } |z_i|\le 500\\ (500-\bmod(z_i,500))\sin\left(\sqrt{|500-\bmod(z_i,500)|}\right)-\frac{(z_i-500)^2}{10000D} & \text{if } z_i>500\\ (\bmod(|z_i|,500)-500)\sin\left(\sqrt{|\bmod(|z_i|,500)-500|}\right)-\frac{(z_i+500)^2}{10000D} & \text{if } z_i<-500\end{cases}$$

(l) Katsuura Function

$$f_{20}(\boldsymbol{x})=\frac{10}{D^2}\prod_{i=1}^{D}\left(1+i\sum_{j=1}^{32}\frac{\left|2^jx_i-round(2^jx_i)\right|}{2^j}\right)^{\frac{10}{D^{1.2}}}-\frac{10}{D^2}$$

(m) HappyCat Function

$$f_{21}(\boldsymbol{x})=\left|\sum_{i=1}^{D}x_i^2-D\right|^{1/4}+(0.5\sum_{i=1}^{D}x_i^2+\sum_{i=1}^{D}x_i)/D+0.5$$

(n) HGBat Function

$$f_{22}(\boldsymbol{x})=\left|(\sum_{i=1}^{D}x_i^2)^2-(\sum_{i=1}^{D}x_i)^2\right|^{1/2}+(0.5\sum_{i=1}^{D}x_i^2+\sum_{i=1}^{D}x_i)/D+0.5$$

(o) Expanded Griewank's plus Rosenbrock's Function

$$f_{23}(\boldsymbol{x})=f_7(f_4(x_1,x_2))+f_7(f_4(x_2,x_3))+\cdots+f_7(f_4(x_{D-1},x_D))+f_7(f_4(x_D,x_1))$$

(p) Expanded Scaffer's F6 Function

$$\text{Scaffer's F6 Function}:\ g(x,y)=0.5+\frac{(\sin^2(\sqrt{x^2+y^2})-0.5)}{(1+0.001(x^2+y^2))^2}$$
$$f_{24}(\boldsymbol{x})=g(x_1,x_2)+g(x_2,x_3)+\cdots+g(x_{D-1},x_D)+g(x_D,x_1)$$

2.3. Construction of composition problems

The idea of constructing composition function is borrowed from [18]. The motivations of proposing the composition function are to avoid algorithms taking advantage of the known property of the benchmark functions, such as local optima lying along the coordinate axes, global optimum having the same values for many decision variables, optimum near the centre of the search space and so on [18]. Based on these reasons, 7 composition functions are constructed as the benchmark functions for multi-modal optimization.

The composition functions take the following form:

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

$F(\boldsymbol{x})$:composition function, $g_i(\boldsymbol{x})$: ith 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 basic function's height, ω_i : weight value for each $g_i(\boldsymbol{x})$, calculated as below:

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

Then the weight is normalized as $\omega_i=w_i/\sum_{i=1}^{n}w_i$
Thus, when

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

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 paper.

The seven composition functions are constructed as follows:

9. Composition Function 1

$N=10$
$\sigma=[10,20,10,20,10,20,10,20,10,20]$
$\lambda=[1,1,1e-6,1e-6,1e-6,1e-6,1e-6,1e-4,1e-4,1e-5,1e-5]$
$bias=[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]+F_9^*$
g_{1-2}: Rotated Sphere Function

$$g_i(\boldsymbol{x})=f_9(\mathbf{M}_i(\boldsymbol{x}-\boldsymbol{o}_i))+bias_i, i=1,\ 2$$

g_{3-4}: Rotated High Conditioned Elliptic Function

$$g_i(\boldsymbol{x})=f_{10}(\mathbf{M}_i(\boldsymbol{x}-\boldsymbol{o}_i))+bias_i, i=3,\ 4$$

g_{5-6}: Rotated Bent Cigar Function

$$g_i(\boldsymbol{x})=f_{11}(\mathbf{M}_i(\boldsymbol{x}-\boldsymbol{o}_i))+bias_i, i=5,\ 6$$

g_{7-8}: Rotated Discus Function

$$g_i(\boldsymbol{x})=f_{12}(\mathbf{M}_i(\boldsymbol{x}-\boldsymbol{o}_i))+bias_i, i=7,\ 8$$

g_{9-10}: Rotated Different Powers Function

$$g_i(\boldsymbol{x})=f_{13}(\mathbf{M}_i(\boldsymbol{x}-\boldsymbol{o}_i))+bias_i, i=9,\ 10$$

10. Composition Function 2

$N=10$
$\sigma=[10,20,30,40,50,60,70,80,90,100]$
$\lambda=[1e-5,1e-5,1e-6,1e-6,1e-6,1e-6,1e-6,1e-4,1e-4,1,1]$
$bias=[0, 10, 20, 30, 40, 50, 60, 70, 80, 90]+F_{10}^*$

g_{1-2}: Rotated High Conditioned Elliptic Function

$g_i(\boldsymbol{x}) = f_{10}(\mathbf{M}_i(\boldsymbol{x}-\boldsymbol{o}_i)) + bias_i, i = 1, 2$

g_{3-4}: Rotated Different Powers Function

$g_i(\boldsymbol{x}) = f_{13}(\mathbf{M}_i(\boldsymbol{x}-\boldsymbol{o}_i)) + bias_i, i = 3, 4$

g_{5-6}: Rotated Bent Cigar Function

$g_i(\boldsymbol{x}) = f_{14}(\mathbf{M}_i(\boldsymbol{x}-\boldsymbol{o}_i)) + bias_i, i = 5, 6$

g_{7-8}: Rotated Discus Function

$g_i(\boldsymbol{x}) = f_{12}(\mathbf{M}_i(\boldsymbol{x}-\boldsymbol{o}_i)) + bias_i, i = 7, 8$

g_{9-10}: Rotated Sphere Function

$g_i(\boldsymbol{x}) = f_9(\mathbf{M}_i(\boldsymbol{x}-\boldsymbol{o}_i)) + bias_i, i = 9, 10$

11. Composition Function 3

$N = 10$

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

$\lambda = [0.1, 0.1, 10, 10, 10, 10, 100, 100, 1, 1]$

$bias = [0, 0, 0, 0, 0, 0, 0, 0, 0, 0] + F_{11}^*$

g_{1-2}: Rotated Rosenbrock's Function

$$g_i(\boldsymbol{x}) = f_{14}(\mathbf{M}_i\frac{2.048(\boldsymbol{x}-\boldsymbol{o}_i)}{100}+1) + bias_i, i = 1, 2$$

g_{3-4}: Rotated Rastrigin's Function

$$g_i(\boldsymbol{x}) = f_{18}(\mathbf{M}_i\frac{5.12(\boldsymbol{x}-\boldsymbol{o}_i)}{100}) + bias_i, i = 3, 4$$

g_{5-6}: Rotated HappyCat Function

$$g_i(\boldsymbol{x}) = f_{21}(\mathbf{M}_i\frac{5(\boldsymbol{x}-\boldsymbol{o}_i)}{100}) + bias_i, i = 5, 6$$

g_{7-8}: Rotated Scaffer's F6 Function

$g_i(\boldsymbol{x}) = f_{24}(\mathbf{M}_i(\boldsymbol{x}-\boldsymbol{o}_i)) + bias_i, i = 7, 8$

g_{9-10}: Rotated Expanded Modified Schwefel's Function

$$g_i(\boldsymbol{x}) = f_{19}(\mathbf{M}_i\frac{1000(\boldsymbol{x}-\boldsymbol{o}_i)}{100}) + bias_i, i = 9, 10$$

12. Composition Function 4

$N = 10$

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

$\lambda = [0.1, 0.1, 10, 10, 10, 10, 100, 100, 1, 1]$

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

g_{1-2}: Rotated Rosenbrock's Function

$$g_i(\boldsymbol{x}) = f_{14}(\mathbf{M}_i\frac{2.048(\boldsymbol{x}-\boldsymbol{o}_i)}{100}+1) + bias_i, i = 1, 2$$

g_{3-4}: Rotated Rastrigin's Function

$$g_i(\boldsymbol{x}) = f_{15}(\mathbf{M}_i\frac{5.12(\boldsymbol{x}-\boldsymbol{o}_i)}{100}) + bias_i, i = 3, 4$$

g_{5-6}: Rotated HappyCat Function

$$g_i(\boldsymbol{x}) = f_{21}(\mathbf{M}_i\frac{5(\boldsymbol{x}-\boldsymbol{o}_i)}{100}) + bias_i, i = 5, 6$$

g_{7-8}: Rotated Scaffer's F6 Function

$g_i(\boldsymbol{x}) = f_{24}(\mathbf{M}_i(\boldsymbol{x}-\boldsymbol{o}_i)) + bias_i, i = 7, 8$

g_{9-10}: Rotated Expanded Modified Schwefel's Function

$$g_i(\boldsymbol{x}) = f_{19}(\mathbf{M}_i\frac{1000(\boldsymbol{x}-\boldsymbol{o}_i)}{100}) + bias_i, i = 9, 10$$

13. Composition Function 5

$N = 10$

$\sigma = [10,20,30,40,50,60,70,80,90,100]$

$\lambda = [0.1, 10, 10, 0.1, 2.5, 1e-3, 100, 2.5, 10, 1]$

$bias = [0, 0, 0, 0, 0, 0, 0, 0, 0, 0] + F_{13}^*$

g_1: Rotated Rosenbrock's Function

$$g_1(\boldsymbol{x}) = f_{14}(\mathbf{M}_1\frac{2.048(\boldsymbol{x}-\boldsymbol{o}_1)}{100}+1) + bias_1$$

g_2: Rotated HGBat Function

$$g_2(\boldsymbol{x}) = f_{22}(\mathbf{M}_2\frac{5(\boldsymbol{x}-\boldsymbol{o}_2)}{100}) + bias_2$$

g_3: Rotated Rastrigin's Function

$$g_3(\boldsymbol{x}) = f_{18}(\mathbf{M}_3\frac{5.12(\boldsymbol{x}-\boldsymbol{o}_3)}{100}) + bias_3$$

g_4: Rotated Ackley's Function

$g_4(\boldsymbol{x}) = f_{15}(\mathbf{M}_4(\boldsymbol{x}-\boldsymbol{o}_4)) + bias_4$

g_5: Rotated Weierstrass Function

$$g_5(\boldsymbol{x}) = f_{16}(\mathbf{M}_5\frac{0.5(\boldsymbol{x}-\boldsymbol{o}_5)}{100}) + bias_5$$

g_6: Rotated Katsuura Function

$$g_6(\boldsymbol{x}) = f_{20}(\mathbf{M}_6\frac{5(\boldsymbol{x}-\boldsymbol{o}_6)}{100}) + bias_6$$

g_7: Rotated Scaffer's F6 Function

$g_7(\boldsymbol{x}) = f_{24}(\mathbf{M}_7(\boldsymbol{x}-\boldsymbol{o}_7)) + bias_7$

g_8: Rotated Expanded Griewank's plus Rosenbrock's Function

$$g_8(\boldsymbol{x}) = f_{23}(\mathbf{M}_8\frac{5(\boldsymbol{x}-\boldsymbol{o}_8)}{100}) + bias_8$$

g_9: Rotated HappyCat Function

$$g_9(\boldsymbol{x}) = f_{21}(\mathbf{M}_9\frac{5(\boldsymbol{x}-\boldsymbol{o}_9)}{100}) + bias_9$$

g_{10}: Rotated Expanded Modified Schwefel's Function

$$g_{10}(\boldsymbol{x}) = f_{19}(\mathbf{M}_{10}\frac{1000(\boldsymbol{x}-\boldsymbol{o}_{10})}{100}) + bias_{10}$$

14. Composition Function 6

$N = 10$

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

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

$bias = [0, 20, 40, 60, 80, 100, 120, 140, 160, 180] + F_{14}^*$

$g_{1,3,5,7,9}$: Rotated Rastrigin's Function

$$g_i(\boldsymbol{x}) = f_{18}(\mathbf{M}_i \frac{5.12(\boldsymbol{x}-\boldsymbol{o}_i)}{100}) + bias_i, i = 1,\ 3,\ 5,\ 7,\ 9$$

$g_{2,4,6,8,10}$: Rotated Expanded Modified Schwefel's Function

$$g_i(\boldsymbol{x}) = f_{19}(\mathbf{M}_i \frac{1000(\boldsymbol{x}-\boldsymbol{o}_i)}{100}) + bias_i, i = 2,\ 4,\ 6,\ 8,\ 10$$

15. Composition Function 7

$N=10$

$\sigma=[10,20,30,40,50,60,70,80,90,100]$

$\lambda=[0.1,\ 10,\ 10,\ 0.1,\ 2.5,\ 1e-3,\ 100,\ 2.5,\ 10,\ 1]$

$bias=[0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0]+F_{15}^*$

g_1: Rotated Rosenbrock's Function

$$g_1(\boldsymbol{x}) = f_{14}(\mathbf{M}_1 \frac{2.048(\boldsymbol{x}-\boldsymbol{o}_1)}{100} + 1) + bias_1$$

g_2: Rotated HGBat Function

$$g_2(\boldsymbol{x}) = f_{22}(\mathbf{M}_2 \frac{5(\boldsymbol{x}-\boldsymbol{o}_2)}{100}) + bias_2$$

g_3: Rotated Rastrigin's Function

$$g_3(\boldsymbol{x}) = f_{18}(\mathbf{M}_3 \frac{5.12(\boldsymbol{x}-\boldsymbol{o}_3)}{100}) + bias_3$$

g_4: Rotated Ackley's Function

$$g_4(\boldsymbol{x}) = f_{15}(\mathbf{M}_4(\boldsymbol{x}-\boldsymbol{o}_4)) + bias_4$$

g_5: Rotated Weierstrass Function

$$g_5(\boldsymbol{x}) = f_{16}(\mathbf{M}_5 \frac{0.5(\boldsymbol{x}-\boldsymbol{o}_5)}{100}) + bias_5$$

g_6: Rotated Katsuura Function

$$g_6(\boldsymbol{x}) = f_{20}(\mathbf{M}_6 \frac{5(\boldsymbol{x}-\boldsymbol{o}_6)}{100}) + bias_6$$

g_7: Rotated Scaffer's F6 Function

$$g_7(\boldsymbol{x}) = f_{24}(\mathbf{M}_7(\boldsymbol{x}-\boldsymbol{o}_7)) + bias_7$$

g_8: Rotated Expanded Griewank's plus Rosenbrock's Function

$$g_8(\boldsymbol{x}) = f_{23}(\mathbf{M}_8 \frac{5(\boldsymbol{x}-\boldsymbol{o}_8)}{100}) + bias_8$$

g_9: Rotated HappyCat Function

$$g_9(\boldsymbol{x}) = f_{21}(\mathbf{M}_9 \frac{5(\boldsymbol{x}-\boldsymbol{o}_9)}{100}) + bias_9$$

g_{10}: Rotated Expanded Modified Schwefel's Function

$$g_{10}(\boldsymbol{x}) = f_{19}(\mathbf{M}_{10} \frac{1000(\boldsymbol{x}-\boldsymbol{o}_{10})}{100}) + bias_{10}$$

3. Optimization algorithms

This section introduces the four optimization algorithms used in the experimental section. These algorithms are compared using the 15 new benchmark functions to evaluate their ability to find multiple optima.

3.1. Crowding differential evolution [8]

Crowding Differential Evolution (CDE) was proposed by Thomsen to extend DE with a crowding scheme which restricts the comparison between the similar individuals and increase the diversity of the population [8]. It allows DE to handle multimodal optimization problems. The crowding factor of CDE is equal to the population size, which means the newly generated offspring compete with the most similar (the similarity is measured by Euclidean distance) individual in the current population. Although the concept of CDE is rather simple, it has been proven to be effective in solving multimodal optimization problems [8].

3.2. Neighborhood based crowding differential evolution

Neighborhood based crowding differential evolution (NCDE) [17] is a recently introduced niching algorithm. It modifies the mutation method of the original CDE by restricting all evolutionary operations within the Euclidean distance based neighborhood. The neighborhood mutation is able to maintain the multiple peaks found during the evolution and evolve toward the respective global/local optima [17]. The neighborhood mutation allows a higher exploitation of the areas that potentially contain optima, thereby facilitating multiple convergences to different optima.

3.3. Ring topology PSO

Use of ring topology PSO (rpso) [21] to solve multimodal optimization problems was introduced by Li in 2010 [21]. The algorithm uses an *lbest* PSO with ring topology to guide the particles and each particle interacts only with its immediate neighbors. Different niches are formed by using the ring topology and subsequently the optimization of multiple peaks are realized. One major disadvantage of rpso is that the ring topology links members from different niches also. However, Li has demonstrated that rpso is able to form stable niches across different local neighborhoods, eventually locating multiple global/local optima on a set of selected problems. Note that only one version of rpso named r2pso is used in the experimental section, as the other three versions perform similarly.

3.4. Locally informed particle swarm

A distance-based Locally Informed Particle Swarm [1] was introduced by Qu et al. to eliminate the need for niching parameter and enhance the fine search ability of PSO [1]. LIPS uses several local bests instead of the global best to guide the search of particles independently in several promising neighborhoods. Hence, it is able to maintain different stable niches by using the information in the neighborhoods. The two main advantages of LIPS are [1]: 1. Benefit of neighborhood based velocity update equation to ensure good usage of neighborhood information especially during the later stages of the search process thereby leading to fast convergence with better accuracy. 2. Euclidean distance based neighborhood selection [22] to ensure the neighbors are from the same niche thereby increasing the algorithm's ability to perform finer local search.

Table 2
Properties and settings of the test functions.

	No.	Functions	Dimension	Goal optima no. global/local*	$F_i^*=F_i(x^*)$
Expanded scalable simple functions	1	Shifted and rotated expanded two-peak trap	5	1/15	100
			10	1/55	
			20	1/210	
	2	Shifted and rotated expanded Five-Uneven-Peak Trap	2	4/21	200
			5	32/0	
			8	256/0	
	3	Shifted and rotated Expanded Equal Minima	2	25/0	300
			3	125/0	
			4	625/0	
	4	Shifted and rotated Expanded Decreasing Minima	5	1/15	400
			10	1/55	
			20	1/210	
	5	Shifted and rotated expanded uneven minima	2	25/0	500
			3	125/0	
			4	625/0	
	6	Shifted and rotated expanded Himmelblau's function	4	16/0	600
			6	64/0	
			8	256/0	
	7	Shifted and rotated expanded six-hump camel back	6	8/0	700
			10	32/0	
			16	256/0	
	8	Shifted and rotated modified Vincent function	2	36/0	800
			3	216/0	
			4	1296/0	
Composition functions	9	Composition Function 1	10, 20, 30	5	900
	10	Composition Function 2	10, 20, 30	5	1000
	11	Composition Function 3	10, 20, 30	5	1100
	12	Composition Function 4	10, 20, 30	5	1200
	13	Composition Function 5	10, 20, 30	5	1300
	14	Composition Function 6	10, 20, 30	5	1400
	15	Composition Function 7	10, 20, 30	5	1500

Search range: $[-100,100]^D$ level of accuracy = 0.1

Table 3
Results of expanded functions using CDE [8].

Func.	Dimension	Best	Worst	Mean	Std
	5	0	0	0	0
1	10	0	0	0	0
	20	0	0	0	0
	2	0	2	0.56	0.65
2	5	0	0	0	0
	8	0	0	0	0
	2	1	9	4.36	2.1385
3	3	0	3	0.68	0.8524
	4	0	4	2	1.2472
	5	0	0	0	0
4	10	0	0	0	0
	20	0	0	0	0
	2	1	7	3.76	1.5351
5	3	0	0	0	0
	4	0	0	0	0
	4	0	0	0	0
6	6	0	0	0	0
	8	0	0	0	0
	6	0	0	0	0
7	10	0	0	0	0
	16	0	0	0	0
	2	4	11	7.44	1.85
8	3	0	5	1.6	1.2583
	4	4	12	6.9	2.331

Table 4
Results of expanded functions using NCDE [6].

Func.	**Dimension**	**Worst**	**Best**	**Mean**	**Std**
	5	1	3	1.88	0.6658
1	10	0	0	0	0
	20	0	0	0	0
	2	2	6	3.84	1.1431
2	5	0	0	0	0
	8	0	0	0	0
	2	17	23	20.08	1.579
3	3	12	25	17.6	3.3166
	4	9	13	10.4	1.6733
	5	0	1	0.08	0.2769
4	10	0	0	0	0
	20	0	0	0	0
	2	13	20	16.24	1.8321
5	3	0	0	0	0
	4	0	0	0	0
	4	8	12	9.64	1.2871
6	6	0	3	1.44	0.8699
	8	0	2	0.8	0.8367
	6	0	3	0.92	0.8622
7	10	0	1	0.04	0.20
	16	0	0	0	0
	2	21	29	24	2.0817
8	3	33	52	42.68	5.4441
	4	30	40	36.4	3.7815

4. Experiments preparations

4.1. Experimental setting

All algorithms are implemented by using Matlab 2013 and executed on a PC with Intel® Core™ i5 CPU and 4 Gb of RAM. The operating system is Microsoft Windows 7. The settings of the 4 algorithms are adopted exactly as in their original works [1,8,17,21].

4.2. Performance measure

Various performance measures have been used in the literature such as average number of optima found, success rate and peak ratio [3]. Different measures have different advantages and

Table 5
Results of expanded functions using R2PSO [21].

Func.	Dimension	Best	Worst	Mean	Std
	5	0	0	0	0
1	10	0	0	0	0
	20	0	0	0	0
	2	0	3	5	3.96
2	5	0	0	0	0
	8	0	0	0	0
	2	18	23	20.48	19.96
3	3	4	18	8.44	3.453
	4	0	3	1.30	1.0593
	5	0	0	0	0
4	10	0	0	0	0
	20	0	0	0	0
	2	13	24	18.92	2.5153
5	3	0	0	0	0
	4	0	0	0	0
	4	0	1	0.04	0.2
6	6	0	0	0	0
	8	0	0	0	0
	6	0	0	0	0
7	10	0	0	0	0
	16	0	0	0	0
	2	12	25	17.52	2.8449
8	3	6	15	10	2.1985
	4	1	9	4.90	2.8848

Table 6
Results of expanded functions using LIPS [1].

Func.	Dimension	Best	Worst	Mean	Std
	5	5	11	8.24	1.2675
1	10	5	11	7.96	1.6197
	20	0	1	0.04	0.20
	2	4	6	4.8	0.6455
2	5	14	21	17.68	1.7010
	8	14	26	23	3.0412
	2	16	25	19.96	2.1502
3	3	21	38	32.12	3.2573
	4	31	45	37.92	3.5814
	5	0	3	1.48	0.8718
4	10	0	0	0	0
	20	0	0	0	0
	2	16	25	20.24	1.7626
5	3	0	0	0	0
	4	0	0	0	0
	4	10	14	12.12	1.0924
6	6	23	34	28.04	2.3359
	8	50	63	55.96	3.1817
	6	6	8	7.32	0.8524
7	10	9	15	12.04	1.4572
	16	16	26	21.32	2.9256
	2	13	20	17.04	1.6704
8	3	36	63	44.48	3.8954
	4	78	96	88.52	4.5655

disadvantages. If we consider the success rate as an example, it is easy to understand and implement. However, it cannot differentiate the performance of algorithms when the test functions are difficult as no algorithm is able locate all the optima and the success rates are just zeros for all algorithms. As for the peak ratio, it tests the quality of optima without considering the distribution of the population which may generate erroneous results as one solution may be used for multiple optima when the number of peaks located is smaller than the actual number of peaks.

Due to the observations listed above, the performance of all multimodal algorithms is measured in terms of the following criteria (25 independent runs of each algorithm are taken on each problem):

1. For the first 8 extended simple problems, average number of optima found is used as the assessment criterion using the

Table 7
Results of composition functions (5 best solutions separated at least by 10).

Func.		CDE	NCDE	R2PSO	LIPS
F9	Solution 1	952.14	900.03	2,073.3	900
	Solution 2	968.61	931.97	1,839.5	915.48
	Solution 3	990.52	936.85	1,233.9	911.84
	Solution 4	964.81	935.64	1,802.6	911.8
	Solution 5	976.75	974.83	1,762.4	920.67
	Mean	**970.566**	**935.864**	**1,742.34**	**911.958**
F10	Solution 1	9,564.1	10,490	15,243	1080.1
	Solution 2	12,551	12,267	20,496	5752.4
	Solution 3	12,460	12,026	18,870	6205.1
	Solution 4	10,834	1,699.3	17,634	6287.1
	Solution 5	2,701	1,139.2	18,915	1070.4
	Mean	**9622.02**	**7,524.3**	**18,231.6**	**4079.02**
F11	Solution 1	1105.3	1,100.3	1,131.3	1102.8
	Solution 2	1,106.7	1,102.9	1,117.7	1102
	Solution 3	1,104.6	1,103	1,102.2	1102.7
	Solution 4	1,107.3	1,103.3	1,131.8	1103.3
	Solution 5	1,104.6	1,111.8	1,145.5	1102.9
	Mean	**1,105.7**	**1,104.26**	**1,125.7**	**1102.74**
F12	Solution 1	1,462.8	2,129.7	2,441.9	1476.6
	Solution 2	1,313.9	1,221.8	2,058.2	1201.2
	Solution 3	1,492.9	1,282.5	1,967.3	1201.5
	Solution 4	1,992.8	1,898	2,073.7	1402.3
	Solution 5	1,853.8	1,731.2	2,091.4	1201.8
	Mean	**1,623.24**	**1,652.64**	**2,126.5**	**1296.68**
F13	Solution 1	1,647.9	1,593.8	225,790	1508.1
	Solution 2	1,658	1,616.4	181,140	1519.4
	Solution 3	1,598.2	1,615.7	70,593	1437.6
	Solution 4	1,580.9	1,503	269,710	1539.9
	Solution 5	1,488.1	1,656.6	5,261.5	1446.2
	Mean	**1,594.62**	**1,597.1**	**150,498.9**	**1490.24**
F14	Solution 1	2,855	1,950.3	3,415.7	2004.5
	Solution 2	2,469	2,207.5	3,287.8	1480
	Solution 3	3,041	2,320.2	2,628	2203.9
	Solution 4	2,512.8	2,959.6	2,765.7	1933.3
	Solution 5	2,528.4	2,472.8	2955.7	1480
	Mean	**2,681.24**	**2,382.08**	**3,010.58**	**1820.34**
F15	Solution 1	1,934	1,940.5	115,360	1778.5
	Solution 2	1,981.7	2,073.7	71,993	1722.6
	Solution 3	2,069.7	1,980.4	66,837	1640
	Solution 4	1,921.7	1,640.5	,2341.8	1824.1
	Solution 5	1,686.4	1,905.9	126,860	1745.3
	Mean	**1,918.7**	**1,908.2**	**76,678.36**	**1742.1**

Table 8
Ranks.

Func.	CDE	NCDE	R2PSO	LIPS
1	2.33	2.00	2.33	1.00
2	2.67	2.33	1.67	1.33
3	3.67	2.00	2.67	1.67
4	1.67	1.33	1.67	1.00
5	2.00	1.67	1.33	1.00
6	3.33	2.00	3.00	1.00
7	2.67	2.00	2.67	1.00
8	3.67	1.67	3.00	1.67
F1–F8	22.00	15.00	18.33	**9.67**
9	3.00	2.00	4.00	1.00
10	3.00	2.00	4.00	1.00
11	3.00	2.00	4.00	1.00
12	2.00	3.00	4.00	1.00
13	2.00	3.00	4.00	1.00
14	3.00	2.00	4.00	1.00
15	3.00	2.00	4.00	1.00
F9–F15	19.00	16.00	28.00	7.00
F1–F15	41.00	31.00	46.33	**16.67**

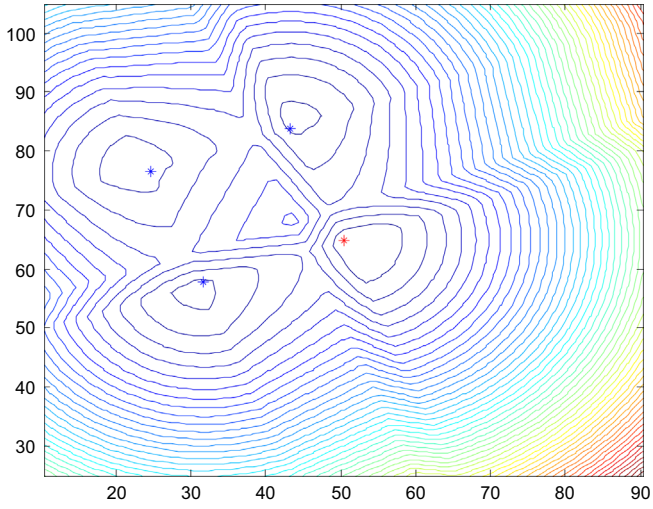

Fig. 1. Contour map for 2-*D* function $F_1(\boldsymbol{x})$.

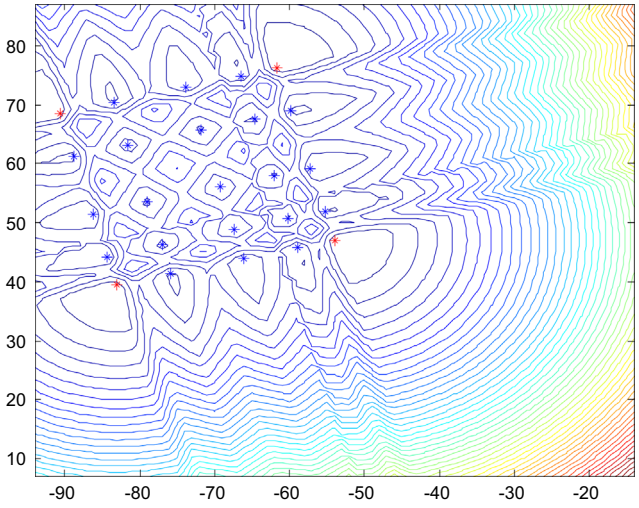

Fig. 2. Contour map for 2-D function $F_2(\boldsymbol{x})$.

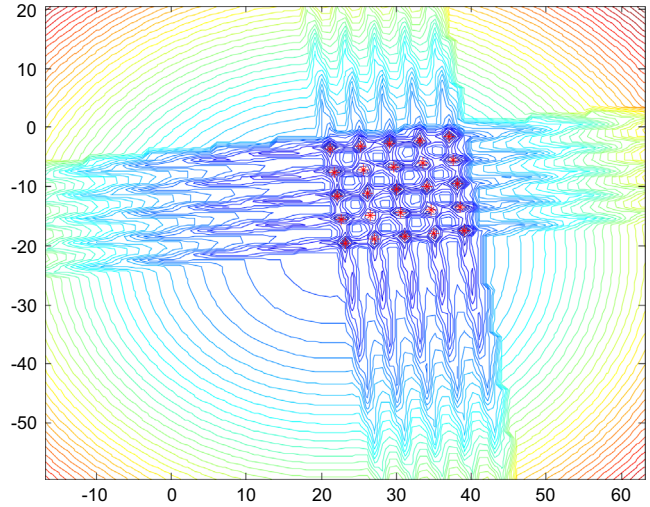

Fig. 3. Contour map for 2-D function $F_3(\boldsymbol{x})$.

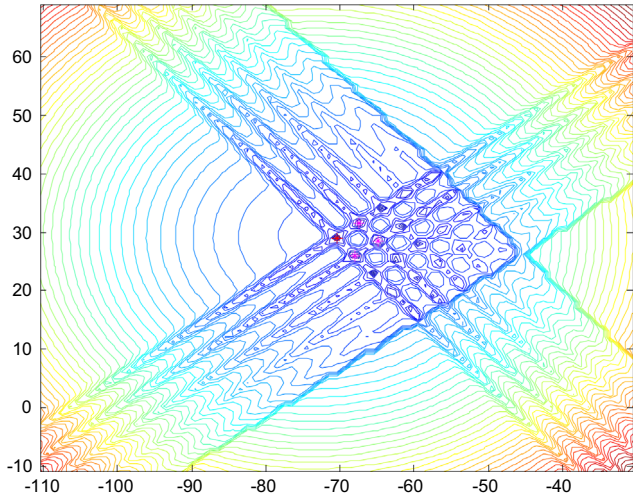

Fig. 4. Contour map for 2-D function $F_4(\boldsymbol{x})$.

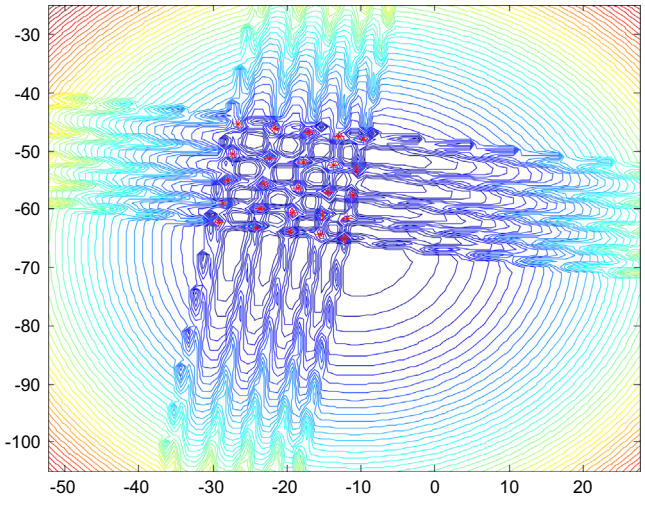

Fig. 5. Contour map for 2-D function $F_5(\boldsymbol{x})$.

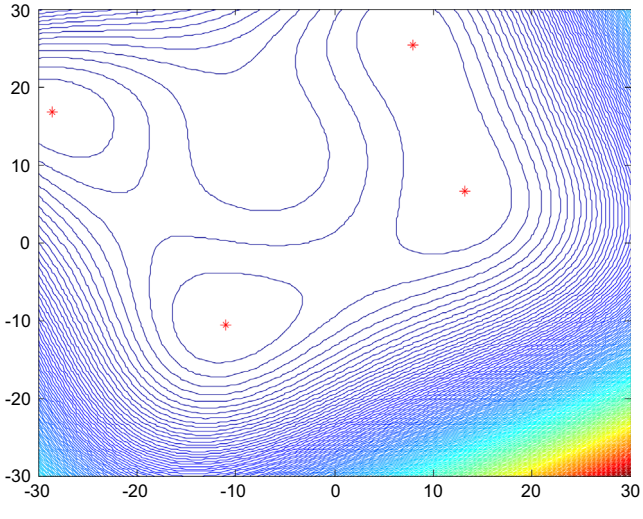

Fig. 6. Contour map for 2-D function $F_6(\boldsymbol{x})$.

given level of accuracy. A level of accuracy, typically $0 < \varepsilon < 1$, is a value indicating how close the computed solutions to the known global peaks are. If the difference from a computed solution to a known global optimum is below ε, then the peak is considered to have been found. Moreover, to ensure different peaks are located by different solutions, a distance restriction is also imposed. That is only when the distance between one solution and its respective optimum is less than $dis_{threshold}$ and

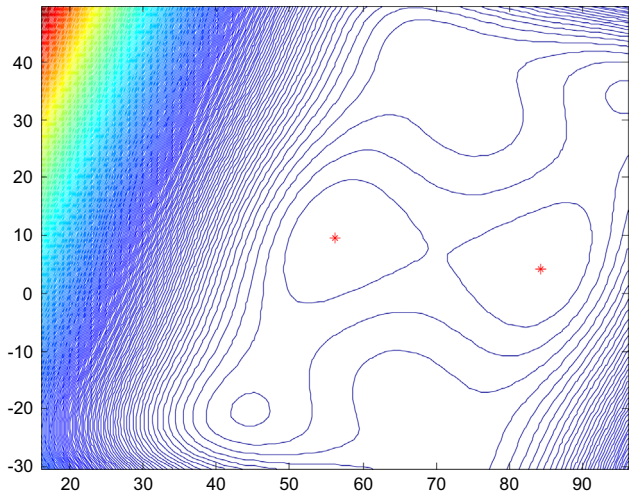

Fig. 7. Contour map for 2-D function $F_7(\boldsymbol{x})$

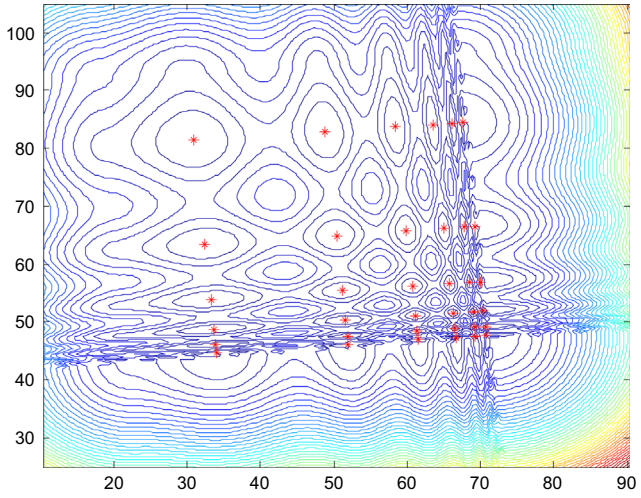

Fig. 8. Contour map for 2-D function $F_8(\boldsymbol{x})$.

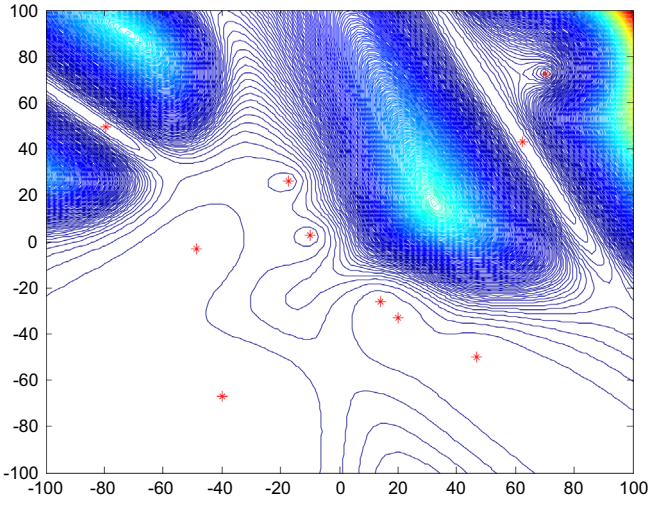

Fig. 9. Contour map for 2-D function $F_9(\boldsymbol{x})$

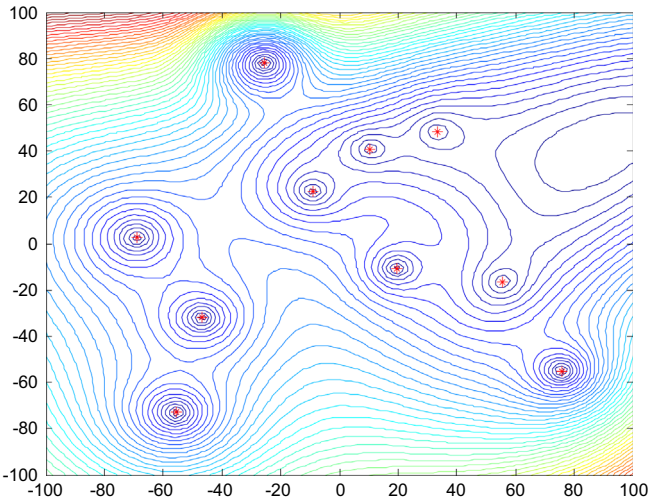

Fig. 10. Contour map for 2-D function $F_{10}(\boldsymbol{x})$.

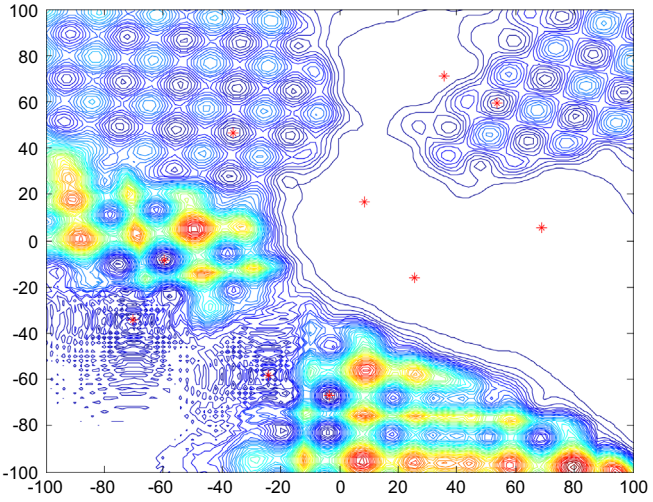

Fig. 11. Contour map for 2-D function $F_{11}(\boldsymbol{x})$.

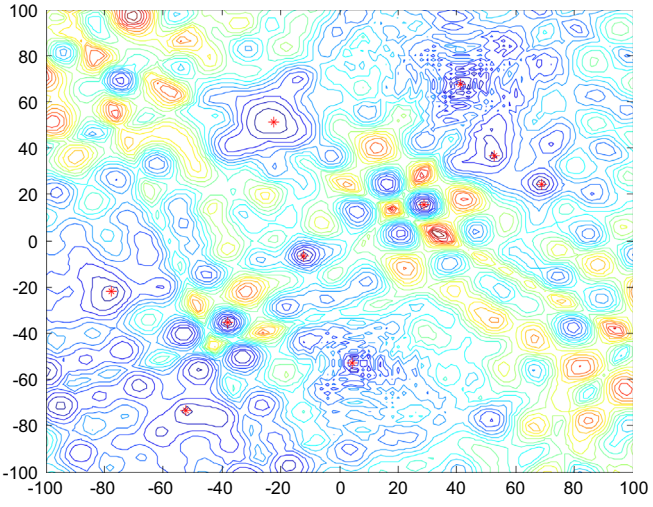

Fig. 12. Contour map for 2-D function $F_{12}(\boldsymbol{x})$.

the difference is below ε, the true solution is considered to have been found. In this way, it avoids one solution being counted more than once when there is no solution near an optimum. Note that the $dis_{threshold}$ is selected as $0.1*D$ for all test functions.

2. For the 7 composition functions, as it is difficult to find any good solutions, a different measure is used. For each problem, all the algorithms need to provide 5 best optima (list the parameters as well as the objective values) of median run that

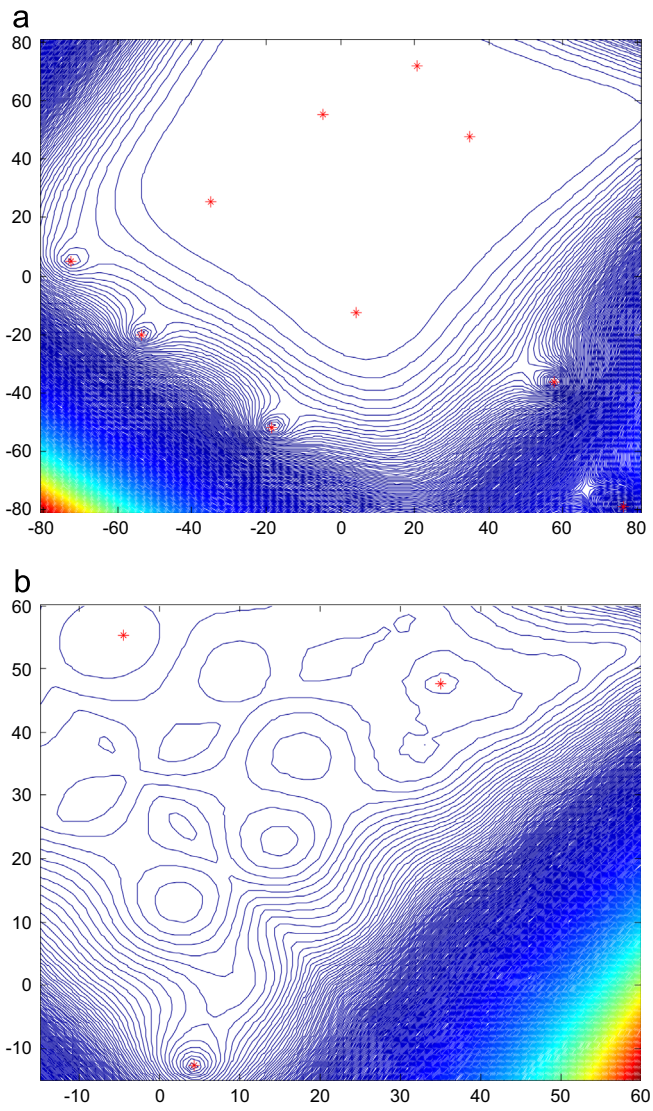

Fig. 13. Contour map for 2-D function $F_{13}(\boldsymbol{x})$

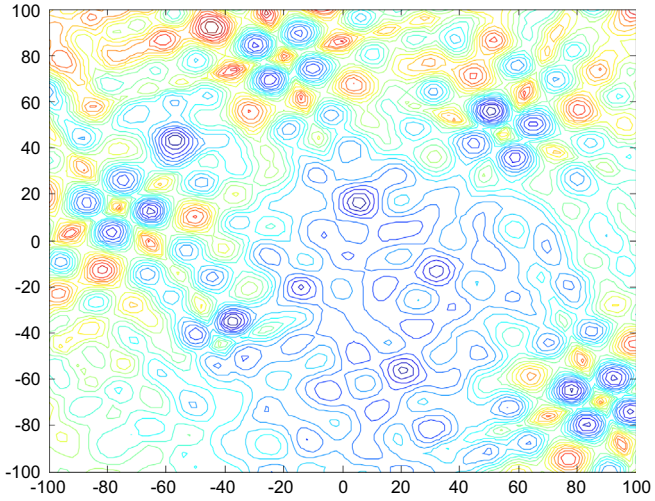

Fig. 14. Contour map for 2-D function $F_{14}(\boldsymbol{x})$.

mutually separated by at least 10 (that is $1*D$ in Euclidean distance. If D is increased 20, the distance will be selected as 20). The results are compared based on these criteria.

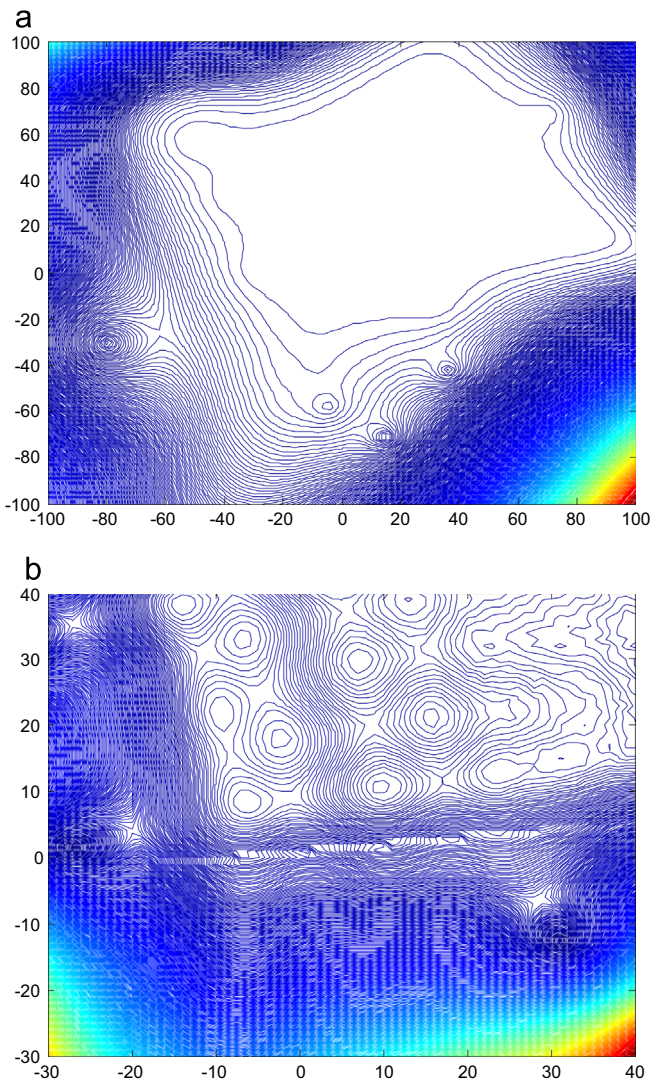

Fig. 15. Contour map for 2-D function $F_{15}(\boldsymbol{x})$.

4.3. Properties and settings of the test functions

The properties of the test functions as well as the settings such as level of accuracy, number of function evaluations are specified in Table 2. Although the dimensionalities of the problems are given in the table, users can scale up these problems according to their own testing requirements. Note that the number of function evaluations is defined by using the following formula $2000*D*\sqrt{q}$, where q is the number of optima and D is the dimensionality of the problem. For example, for function 1, $q=16$, MaxFES$=2000*5*4=40000$.

5. Experimental results

To compare the performance of the 4 algorithms on the proposed test problems, 25 independent runs are executed for each problem. The best run, worst run as well as the mean value (in boldface) of the simple functions are presented in Tables 3–6. In Tables 7 and 5 best optima (the objective values) of median run found by each algorithms are presented. The mean values of the 5 best optima are also calculated and highlighted in boldface in Table 7. Comparing the results in these tables, it is easy to find out that LIPS performs the best among the compared 4 algorithms. The

results suggest the same conclusion as in [1]. The superior performance is due to the better fine search generated by the locally informed particles. As LIPS selects several distance based nearest neighbors, the search can be easily localized with respect to each niche especially during the later search stages [1]. The neighborhood concept is also able to form stable subpopulations around different areas and these areas potentially contain the desired optimal points thereby solving the multimodal optimization efficiently. Table 8 presents the ranks of the four algorithms for Expanded Functions, Composition Functions and all the 15 test functions. For Expanded Functions, CDE performs the worst out of the 4 compared algorithms. This is because although CDE increases the diversity of the population, it slows down the convergence of the algorithm. Therefore, CDE is not able to provide highly accurate solutions with a small number of FEs. Moreover, the performances of all the compared algorithms are poor on test function F4. This is because there is only 1 global optimum for this function and the differences are quite large for the local optima which create extra difficulties for the optimization algorithms. For Composition Functions, R2PSO has the worst performance. It is because that the landscapes of Composition Functions are much more complex and the diversity of R2PSO is not enough to make the particles search the whole space. Among all the four algorithms considered in the comparison, LIPS provides the best performance on Expanded Functions and Composition Functions.

6. Conclusions

In this paper, 15 novel multimodal benchmark test functions have been presented. These functions offer an easy way to construct scalable functions and a challenging environment for comparing novel multimodal optimization algorithms with the state-of-the-art. These functions are divided into two categories, namely as the extended simple functions and composition functions, respectively. The extended simple functions were obtained through rotating and shifting some well known multimodal test functions like Two-Peak Trap functions, while the composition functions were obtained from modifying the functions presented in [20]. The complexity of the proposed test functions was demonstrated by employing four optimization approaches to solve them. The results demonstrate the difficulty of the problems for the recently proposed algorithms. Hence, we trust that these challenging scalable problems will encourage researchers to develop superior multimodal optimization algorithms in the future.

Acknowledgments

This research is partially supported by National Natural Science Foundation of China (61305080, 61473266, 61379113) and Postdoctoral Science Foundation of China (2014M552013) and the Scientific and Technological Project of Henan Province (132102210521).

References

[1] B.Y. Qu, P.N. Suganthan, S. Das, A distance based locally informed particle swarm optimization for multi-modal optimization, IEEE Trans. Evol. Comput. 17 (3) (2013) 387–402.
[2] J. Ronkkonen, X.D. Li, V. Kyrki, A framework for generating tunable test functions for multimodal optimization, Soft Comput. 15 (2011) 1689–1706.
[3] B.Y. Qu, J.J. Liang, P.N. Suganthan, Niching particle swarm optimization with local search for multi-modal optimization, Inf. Sci. 197 (2012) 131–143.
[4] S.W. Mahfoud, A comparison of parallel and sequential niching methods July, Proc. of Sixth Int. Conf. on Genetic Algorithms, USA (1995) 136–143.
[5] J.E. Vitela, O. Castanos, A real-coded niching memtic algorithm for continuous multimodal function optimization, IEEE Congr. Evol. Comput. (2008) 2170–2177.
[6] B.Y. Qu, P.N. Suganthan, J.J. Liang, Differential evolution with neighborhood mutation for multimodal optimization, IEEE Trans. Evol. Comput. 16 (5) (2012) 601–614.
[7] B. Sareni, L. krahenbuhl, Fitness sharing and niching methods revisited IEEE Trans. Evol. Comput., (3) (Sept. 1998) 97–106.
[8] R. Thomsen, Multimodal optimization using Crowding-based differential evolution, Proc. of the IEEE 2004 Congress on Evolutionary Computation (2004) 1382–1389.
[9] D.E. Goldberg, J. Richardson, Genetic algorithms with sharing for multimodal function optimization, Proc. of Second Int. Conf. on Genetic Algorithms (1987) 41–49.
[10] G. R. Harik, Finding multimodal solutions using restricted tournament selection, in: Proc. of Sixth International Conf. on Genetic Algorithms, pp. 24–31, San Francisco, 1995.
[11] A. Petrowski, A clearing procedure as a niching method for genetic algorithms, in: Proc. of Third IEEE Congress on Evolutionary Computation (1996) 798–803.
[12] S. Biswas, S. Kundu, S. Das., Inducing niching behavior in differential evolution through local information sharing, IEEE Trans. Evol. Comput. 19 (2) (2015) 246–253. http://dx.doi.org/10.1109/TEVC.2014.2313659.
[13] S. Hui, P. N. Suganthan "Ensemble and arithmetic recombination-based speciation differential evolution for multimodal optimization," accepted by IEEE Trans. Cybern., http://dx.doi.org/10.1109/TCYB.2015.2394466.
[14] S. Biswas, S. Kundu, S. Das., An improved parent-centric mutation with normalized neighborhoods for inducing niching behavior in differential evolution, IEEE Trans. Cybern. 44 (10) (2014) 1726–1737.
[15] K. Wong, C. Wu, K.P. Mok, C. Peng, Z. Zhang, Evolutionary multimodal optimization using the principle of locality, Inf. Sci. 194 (2012) 138–170.
[16] J.A. Vrugt, B.A. Robinson, J.M. Hyman, Self-adaptive multimethod search for global optimization in real-parameter spaces, IEEE Trans. Evol. Comput. 13 (2) (2009) 243–259, April.
[17] B. Y. Qu, P. N. Suganthan, Novel multimodal problems and differential evolution with ensemble of restricted tournament selection, 2010 IEEE Congr. Evol. Comput., 2010, 1–7.
[18] J.J. Liang, P.N. Suganthan, K. Deb, Novel composition test functions for numerical global optimization, Proc. of IEEE Swarm Intelligence Symposium (2005) 68–75.
[19] 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, 2013, January.
[20] J.J. Liang, B.Y. Qu, P.N. Suganthan, An efficient biogeography based optimization algorithm for solving reliability optimization problems, "Problem Definitions and Evaluation Criteria for the CEC 2014 Special Session and Competition on Single Objective Real-Parameter Numerical Optimization" Technical Report 201311, Computational Intelligence Laboratory, Zhengzhou University, Zhengzhou China and Technical Report, Nanyang Technological University, Singapore, 2013, December.
[21] X. Li, Niching without niching parameters: particle swarm optimization using a ring topology, IEEE Trans. Evol. Comput. 14 (1) (2010) 150–169.
[22] P.N. Suganthan, Particle swarm optimizer with neighbourhood operator, Proc. CEC (1999) 1958–1962. http://dx.doi.org/10.1109/CEC.1999.785514, July.
[23] S. Das, S. Maity, B-Y Qu, P.N. Suganthan, Real-parameter evolutionary multimodal optimization—a survey of the state-of-the-art, Swarm Evol. Comput. 1 (2) (2011) 71–78.