J Glob Optim (2011) 49:69–90
DOI 10.1007/s10898-010-9535-7

A new differential mutation base generator for differential evolution

Chuan Lin · Anyong Qing · Quanyuan Feng

Received: 22 December 2008 / Accepted: 11 February 2010 / Published online: 27 February 2010

Abstract A new differential mutation base strategy for differential evolution (DE), namely best of random, is proposed. The best individual among several randomly chosen individuals is chosen as the differential mutation base while the other worse individuals are donors for vector differences. Hence both good diversity and fast evolution speed can be obtained in DE using the new differential mutation base. A comprehensive comparative study is carried out over a set of benchmark functions. Numerical results show that a better balance of reliability and efficiency can be obtained in differential evolution implementing the new generator of differential mutation base, especially in functions with high dimension.

Keywords Differential evolution · Differential mutation · Best of random · Diversity

1 Introduction

Differential evolution (DE) [1–6] is a simple, efficient and robust evolutionary algorithm (EA). It has found increasing applications in a number of scientific and engineering fields. Many researches have reported that DE outperformed some other EAs when optimizing benchmark functions or real-world problems [4–12].

In the community of differential evolution [2,13], differential evolution algorithms are marked as DE/x/y/z where x indicates how the differential mutation base is chosen, $y \geq 1$ is the number of vector differences added to the differential mutation base vector, and z indicates which kind of crossover operation is used. It is well known that differential mutation

C. Lin (✉) · Q. Feng
School of Information Science and Technology, Southwest Jiaotong University, Chengdu 610031, China
e-mail: lin_langai@163.com

C. Lin · A. Qing
Temasek Laboratories, National University of Singapore, 5A Engineering Dr 1#06-09,
Singapore 117508, Singapore
e-mail: tslqay@nus.edu.sg

is the crucial idea behind the success of differential evolution. Best and random are the two most often implemented differential mutation bases.

Through a comprehensive parametric study on differential evolution [5,14,15], it has been observed that differential evolution algorithms with best differential mutation base are in general more efficient while those with random differential mutation base are in general more reliable. Neither of them achieves harmony between reliability and efficiency, a challenge facing all evolutionary algorithms.

Generally speaking, differential mutation can be regarded as local search around differential mutation base. Guidance provided by the best differential mutation base leads to higher efficiency at the cost of diversity while blind search by using random differential mutation bases gains diversity and sacrifices efficiency. Hence, effectively balancing exploration and exploitation by using differential mutation base providing both guidance and diversity simultaneously might be a more promising solution. The new generator for differential mutation base, namely the best of random differential mutation base, is accordingly proposed.

The new generator for differential mutation base is very simple and introduces no additional intrinsic control parameter. Like random differential mutation, best of random differential mutation also randomly chooses several mutually different individuals. However, unlike random differential mutation, best of random differential mutation uses the best individual among these individuals as differential mutation base while the other worse individuals are donors for vector differences.

The new differential mutation base has been examined by comprehensive simulations over a set of benchmark functions. Numerical results show that a better balance of reliability and efficiency can be obtained in differential evolution implementing the new generator of differential mutation base, especially in functions with high dimension.

The rest of this paper is arranged as follows. Section 2 briefly describes differential evolution. Details of the new generator of differential mutation base are given in Sect. 3. A platform for evaluating differential evolution is constructed in Sect. 4. Section 5 presents numerical results of the conducted numerical evaluation. Conclusion is given in Sect. 6.

2 Differential evolution

2.1 Convention

Minimizing a single objective function $f(\mathbf{x})$ with N-dimensional real optimization parameters $\mathbf{x}$ without any constraint is considered in this paper. Natural real code is adopted.

Differential evolution operates on a population containing N_p individuals. It involves two stages, namely, initialization and evolution. Initialization generates initial population $\mathbf{P}^0$. Then $\mathbf{P}^0$ evolves to $\mathbf{P}^1$, $\mathbf{P}^1$ evolves to $\mathbf{P}^2$, …, and so on, until the termination conditions are fulfilled. While evolving from $\mathbf{P}^n$ to $\mathbf{P}^{n+1}$, the three evolutionary operations, namely, differential mutation, crossover and selection are executed in sequence.

2.2 Classic differential evolution

The Fortran-style pseudo-code of classic differential evolution (CDE) is shown in Fig. 1 where the vector of optimization parameters $\mathbf{x}_{i,n}$ is the ith individual in population $\mathbf{P}^n$, $\mathbf{v}_{i,n+1}$ is the mutant corresponding to $\mathbf{x}_{i,n}$, $\mathbf{b}_{i,n}$ is the differential mutation base, p_{1y} and p_{2y} are integer numbers, usually random, F_y, a constant usually in [0,1], is the mutation intensity for the y^{th} vector difference $\mathbf{x}_{p_{1y},n} - \mathbf{x}_{p_{2y},n}$, $\mathbf{x}_{p_{1y},n}$ and $\mathbf{x}_{p_{2y},n}$ are the donors for vector difference,

Initialization

$n = 0$

do $i = 1, N_p$

do $j = 1, N$

$$x_{i,j,0} = b_j^L + \alpha_{i,j}(b_j^U - b_j^L)$$

end do

evaluate $f(\mathbf{x}_{i,0})$

end do

Evolution

do while termination conditions are not satisfied

do $i = 1, N_p$

differential mutation to obtain mutant $\mathbf{v}_{i,n+1}$

choose differential mutation base $\mathbf{b}_{i,n}$, donors $\mathbf{x}_{p_{1y},n}$ and $\mathbf{x}_{p_{2y},n}$

$$\mathbf{v}_{i,n+1} = \mathbf{b}_{i,n} + \sum_{y \geq 1} F_y(\mathbf{x}_{p_{1y},n} - \mathbf{x}_{p_{2y},n}),\ 1 \leq i \neq p_{1y} \neq p_{2y} \leq N_p$$

regularize infeasible mutant $\mathbf{v}_{i,n+1}$

do $j = 1, N$

do while ($v_{i,j,n+1} < b_j^L$)

$$v_{i,j,n+1} = v_{i,j,n+1} + b_j^U - b_j^L$$

end do

do while ($v_{i,j,n+1} > b_j^U$)

$$v_{i,j,n+1} = v_{i,j,n+1} - b_j^U + b_j^L$$

end do

end do

crossover mutant $\mathbf{v}_{i,n+1}$with target individual $\mathbf{x}_{i,n}$ to deliver child $\mathbf{c}_{i,n+1}$

evaluate child $f(\mathbf{c}_{i,n+1})$

selection to get individual $\mathbf{x}_{i,n+1}$

if $f(\mathbf{c}_{i,n+1}) < f(\mathbf{x}_{i,n})$

$\mathbf{x}_{i,n+1} = \mathbf{c}_{i,n+1}$

else

$\mathbf{x}_{i,n+1} = \mathbf{x}_{i,n}$

end if

end do

$n = n + 1$

end do

Fig. 1 Fortran-style pseudo-code of classic differential evolution

$\mathbf{c}_{i,n+1}$.is the child or trial individual generated by crossover. $\alpha_{i,j}$ is a real random number in [0,1], $[b_i^L, b_i^U]$ is the search space of the jth optimization parameter, $x_{i,j}$.

2.2.1 Differential mutation base

There are various differential mutation base generators among which best and random differential mutation bases are the most often implemented.

A. Best differential mutation base

In this scheme, the best or the most dominant (in terms of objective function value) individual $\mathbf{x}_{best,n}$ in $\mathbf{P}^n$ is chosen as the differential mutation base.

B. Random differential mutation base

In this scheme, the differential mutation base is randomly chosen from $\mathbf{P}^n$.

2.2.2 Crossover

Likewise, different crossover schemes are applicable, among which binomial crossover and exponential crossover are the most often implemented.

A. Binomial crossover

In this scheme, $\mathbf{c}_{i,n+1}$ is generated as follows

$$c_{i,j,n+1} = \begin{cases} v_{i,j,n+1} & \beta_{i,j,n+1} \le C_r \\ x_{i,j,n} & \text{otherwise} \end{cases} \tag{1}$$

where $\beta_{i,j,n+1}$ is a real random number uniform in the range [0,1] and C_r, a constant in [0,1], is the crossover probability.

There is an extreme case. The child duplicates $\mathbf{v}_{i,n+1}$. In this case, a randomly chosen parameter of $\mathbf{x}_{i,n}$, $x_{i,j,n}$, will replace the corresponding parameter of the child $\mathbf{c}_{i,n+1}$, $c_{i,j,n+1}$.

There is another extreme case. $\mathbf{c}_{i,n+1}$inherits no parameter from $\mathbf{v}_{i,n+1}$ and hence no evolution happens. In this case, a randomly chosen parameter of the child $\mathbf{c}_{i,n+1}$, $c_{i,j,n+1}$, will be replaced by the corresponding parameter of the mutant $\mathbf{v}_{i,n+1}$, $v_{i,j,n+1}$.

B. Exponential crossover

The Fortran-style pseudo-code is given in Fig. 2.

```
do k = 1, N
   c_{i,j,n+1} = x_{i,j,n}
end do
r = N * rand(0, 1) + 1
k = r
c_{i,k,n+1} = v_{i,k,n+1}
M = 1
β = rand(0, 1)
do while (β ≤ C_r and M < N -1)
   M = M + 1
   β = rand(0, 1)
   k = 1 + mod(k, N)
   c_{i,k,n+1} = v_{i,k,n+1}
end do
```

Fig. 2 Fortran-style pseudo-code for exponential crossover for differential evolution

2.2.3 Termination conditions

In this paper, the following two termination conditions are applied simultaneously.

A. Objective met

This termination condition is only applicable to optimization problems whose optima are known. Such optimization problems include multi-dimensional benchmark functions and benchmark application problems for unconstrained single-objective optimization. Usually, they are involved in algorithm evaluation.

In this study, it can be mathematically expressed as

$$f\left(\mathbf{x}_{p,n}\right) - f\left(\mathbf{x}^*\right) \leq \varepsilon_f \quad \exists 1 \leq p \leq N_p \tag{2}$$

where $f(\mathbf{x}^*)$ is the known optima, ε_f is the predefined error tolerance.

B. Limit of number of objective function evaluations

Unlimited time to search optimal solution is unaffordable in practice. The search process has to be terminated when the consumed time exceeds certain limit. Although limit on search time is straightforward, it does not capture the essence of optimization and is dependent on hardware. Instead, limit of number of objective function evaluations is a hardware-free marker for consumed time.

2.3 Dynamic differential evolution

Classic differential evolution is static in nature from the point of view of population updating and is therefore inherently inefficient. Inspired by the advantage of Gauss-Seidel method against Jacobi method for linear equations, the dynamic differential evolution (DDE) [16] was developed. The Fortran-style pseudo-code for dynamic differential evolution is shown in Fig. 3, where the generation indexes n and $n + 1$ do not show up any more. This indicates that all evolutionary operations are executed over individuals in the current population. There is only one population whose individuals are continuously updated by their competitive children. For more details of dynamic differential evolution, please refer to [16].

Extensive performance evaluation has been carried out [5]. As expected, in general, dynamic differential evolution outperforms classic differential evolution. But comparing with CDE, DDE has a disadvantage. Unlike CDE, it cannot be parallelized. For the problems where the function evaluation requires much computation time, parallelization is important hence CDE is more suitable than DDE to solve them.

3 Best of random differential mutation base

For most evolutionary algorithms including DE, how to balance two contradictory aspects: exploration and exploitation, is a crucial problem. In fact, this problem can also be understood from another point of view: the balance of diversity and evolution of the population. Diversity here means the dispersing of the individuals in the population, while evolution means constant improvement of the (average) performance of the population as they evolve from one generation to the next. For a good balance of them, in principle we should assign more search resource to the promising regions for fast evolution speed while at the same time keep the diversity of the population from decreasing rapidly for reliable convergence.

Generally speaking, differential mutation operation in DE could be regarded as a local search. Differential mutation base vector acts as the local search center and the scaled vector difference determines the search range or search step around the center. The diversity of the

Initialization
  $n = 0$
  do $i = 1, N_p$
    do $j = 1, N$
      $x_{i,j} = b_j^L + \alpha_{i,j}(b_j^U - b_j^L)$
    end do
    evaluate $f(\mathbf{x}_i)$
  end do
Evolution
  $n = 1$
  $i = 1$
  do while termination conditions are not satisfied
    differential mutation to obtain mutant $\mathbf{v}_i$
      choose differential mutation base $\mathbf{b}_i$, donors $\mathbf{x}_{p_{1y}}$ and $\mathbf{x}_{p_{2y}}$
      $\mathbf{v}_i = \mathbf{b}_i + \sum_{y \geq 1} F_y(\mathbf{x}_{p_{1y}} - \mathbf{x}_{p_{2y}}),\ \ 1 \leq i \neq p_{1y} \neq p_{2y} \leq N_p$
      regularize infeasible mutant $\mathbf{v}_i$
        do $j = 1, N$
          do while ($v_{i,j} < b_j^L$)
            $v_{i,j} = v_{i,j} + b_j^U - b_j^L$
          end do
          do while ($v_{i,j} > b_j^U$)
            $v_{i,j} = v_{i,j} - b_j^U + b_j^L$
          end do
        end do
    crossover mutant $\mathbf{v}_i$ with target individual $\mathbf{x}_i$ to deliver child $\mathbf{c}_i$
    evaluate child $f(\mathbf{c}_i)$
    selection to get individual $\mathbf{x}_i$
      if $f(\mathbf{c}_i) < f(\mathbf{x}_i)$
        $\mathbf{x}_i = \mathbf{c}_i$
        additional selection
          if $f(\mathbf{c}_i) < f(\mathbf{x}_{\text{best}})$ $\mathbf{x}_{\text{best}} = \mathbf{c}_i$
      end if
    $i = 1 + \text{mod}(i, N_p)$
    if($i = 1$) $n = n + 1$
  end do

Fig. 3 Fortran-style pseudo-code of dynamic differential evolution

population at the next generation depends on the diversity of the local search centers (base vectors). In DE/rand/*/*, the base vectors are randomly chosen from the population, hence a good diversity of base vectors (local search centers) is gained. But as the base vectors have no preference for promising regions or solutions, the evolution speed of DE/rand/*/* is relatively slow. On the other hand, all the individuals in DE/best/*/* use the same base vector: the best vector so far. This kind of concentrated search around the best individual often leads to fast evolution speed but is more inclined to stagnation due to the lack of enough diversity of base vectors.

From the above analysis, it is noticed that using base vectors with high quality in DE often brings fast evolution speed, and increasing the diversity of base vectors also promotes the diversity of the population. Differential mutation base providing both guidance and diversity should be a more promising choice. Based on this idea, a new generator for differential mutation base, namely best of random, is proposed. Best of random differential mutation base uses the best individual among several randomly chosen individuals as differential mutation

$p_b = N_p$ * rand(0, 1) + 1
$p_1 = N_p$ * rand(0, 1) + 1
do while ($p_1 = p_b$) $p_1 = N_p$ * rand(0, 1) + 1
if $f\left(\mathbf{x}_{p_1,n}\right) < f\left(\mathbf{x}_{p_b,n}\right)$ swap p_1 and p_b
$p_2 = N_p$ * rand(0, 1) + 1
do while ($p_2 = p_b$ or $p_2 = p_1$) $p_2 = N_p$ * rand(0, 1) + 1
if $f\left(\mathbf{x}_{p_2,n}\right) < f\left(\mathbf{x}_{p_b,n}\right)$ swap p_2 and p_b

Fig. 4 Fortran-style pseudo-code for best of random differential mutation base

base and the other worse individuals as donors for vector differences. Thus both good quality and diversity of the base vectors can be obtained. The formula for best of random differential mutation strategy can be expressed as:

$$\mathbf{v}_{i,n+1} = \mathbf{x}_{p_b,n} + \sum_{y\geq 1} F_y(\mathbf{x}_{p_{1y},n} - \mathbf{x}_{p_{2y},n}),\ 1 \leq i \neq p_{1y} \neq p_{2y} \leq N_p \tag{3}$$

where, the individuals $\mathbf{x}_{p_{1y},n}, \mathbf{x}_{p_{2y},n}$ and $\mathbf{x}_{p_b,n}$ are randomly chosen from the population, and $\mathbf{x}_{p_b,n}$ is the best one of them, i.e. $f(\mathbf{x}_{p_b,n}) \leq \min(f(\mathbf{x}_{p_{1y},n}), f(\mathbf{x}_{p_{2y},n}))$. The Fortran-style pseudo-code of best of random differential mutation base generator is shown in Fig. 4. DE implementing best of random differential mutation base is denoted as DE/BoR/*/*. In this paper, DE/BoR/1/* is used. It should be pointed out that there is no need to modify any other evolutionary operations in differential evolution. What we need is to write a separate subroutine for the new differential mutation base and replace existing subroutines for differential mutation base, as highlighted by shadow in Figs. 1 and 3. It can be found easily that DE/BoR/1/* is almost as simple as DE/rand/1/* and introduce no extra intrinsic control parameters.

4 Evaluation platform

4.1 Test bed

For a fair comparison of different DE algorithms, a test bed consisting of 13 benchmark functions with different representative mathematical features, such as modality, separability, differentiability, symmetry, is used. The definition and mathematical features of the benchmark functions are given in Appendix.

All the benchmark functions are scalable for the sake of investigating the effect of dimensionality. Two different dimensions, $N = 8$ and 50 representing low and high dimensional problems respectively, are used in the simulations. For dimension 50, as the simulation of some functions are computationally very expensive, only nine functions are simulated. Function Griewank of dimension 50 is only simulated with CDE schemes as it is also computationally very expensive and it is observed that the simulation results of multimodal functions of dimension 50 in CDE and DDE are similar.

4.2 Differential evolution algorithms

Twelve differential evolution algorithms are involved in this study

1. CDE/best/1/bin (CBB)

2. CDE/rand/1/bin (CRB)
3. CDE/BoR/1/bin (CBoB)
4. CDE/best/1/exp (CBE)
5. CDE/rand/1/exp (CRE)
6. CDE/BoR/1/exp (CBoE)
7. DDE/best/1/bin (DBB)
8. DDE/rand/1/bin (DRB)
9. DDE/BoR/1/bin (DBoB)
10. DDE/best/1/exp (DBE)
11. DDE/rand/1/exp (DRE)
12. DDE/BoR/1/exp (DBoE)

4.3 Intrinsic control parameters

Here, the finite set of intrinsic control parameters is $\mathbf{S}_{\text{icp}} = \mathbf{S}_{\text{Np}} \cup \mathbf{S}_{\text{F}} \cup \mathbf{S}_{\text{Cr}}$. All combinations of 3 intrinsic control parameters from their sets $\mathbf{S}_{\text{Np}}$, $\mathbf{S}_{\text{F}}$ and $\mathbf{S}_{\text{Cr}}$ are simulated in our experiments.

4.3.1 Population size

Two different sets of population sizes are used

$$\mathbf{S}_{\text{Np}} = \{16, 24, 32, 40, 48, 56, 64, 72, 80, 120\}$$
$$\mathbf{S}_{\text{Np}} = \{20, 30, 40, 50, 60, 70, 80\}$$

The first set of population size is for simulating 8-dimensional benchmark functions while the second set for 50-dimensional benchmark functions. Initially, the sets of population sizes, $\mathbf{S}_{\text{Np}} = \{8, 16, 24, 32, 40, 48, 56, 64, 72, 80, 120, 160, 200, 400\}$ and $\mathbf{S}_{\text{Np}} = \{10, 20, 30, 40, 50, 60, 70, 80, 90, 100, 150, 200, 250, 500\}$, are devised for 8 and 50-dimensional functions in our previous parametric study of DE/best/1/* and DE/rand/1/* [5]. It has been observed from the simulation results that DE does not benefit from increasing of population size if it has been above the optimal population size. Thus in this paper we choose smaller sets of population sizes, but if necessary, other population sizes will be added for some benchmark functions in order to include the optimal population size.

4.3.2 Mutation intensity

The set of tested mutation intensity takes the form as $\mathbf{S}_{\text{F}} = \{F | F = j \times \Delta_F, 1 \leq j \leq m/\Delta_{\text{F}}\}$, $m = 1$ unless specified otherwise. $\Delta_{\text{F}} = 0.025$ is used in this paper.

4.3.3 Crossover probability

The set of tested crossover probability $\mathbf{S}_{\text{Cr}} = \{C_r | C_r = j \times \Delta_{Cr}, 1 \leq j \leq 1/\Delta_{Cr}\}$, Δ_{Cr} is usually equal to Δ_{F}.

4.4 Termination conditions

The error tolerance ε_f for each benchmark function is set to be 1×10^{-2}. The default limit of number of objective function evaluation is 2000 times the number of optimization parameters.

But it is increased to 64,000, 128,000 and 128,000 for 8-dimensional functions Griewank, Whitley and Weierstrass, respectively, and 200,000 for 50-dimensional Griewank function.

4.5 Performance indicators

Performances of evolutionary algorithms include reliability, efficiency, and robustness, in the order of importance, or priority. Reliability shows how confident the desired optima can be found by the evolutionary algorithm, efficiency shows how fast the evolutionary algorithm finds the desired optima, while robustness shows how sensitive the algorithm's efficiency is to other factors, such as randomness, intrinsic and non-intrinsic control parameters.

Evaluation of evolutionary algorithms can be done at two levels, i.e., parameter level and system level. Parameter level is the low level while system level is the high level.

4.5.1 Search and trial

A search is an execution of an evolutionary algorithm with fixed intrinsic control parameters. It is successful if it finds the desired optima. The number of objective function evaluations (NOFE) of a successful search is recorded.

To reduce randomness, each search with fixed intrinsic control parameters is repeated 100 times in this paper, such a repetition is a trial. The number of successful searches (NSS) in a trial is recorded. A trial with at least one successful search is a partially successful trial. In accordance, the average number of objective function evaluations (ANOFE) of all successful searches in a partially successful trial is computed. If all of the searches in a trial are successful, it is a successful trial.

4.5.2 Parameter level evaluation

So far, attention has been focused on evaluation of evolutionary algorithms at parameter level. At this level, evolutionary algorithms are evaluated with fixed intrinsic and non-intrinsic control parameters. The number of successful searches among a trial indicates reliability, the average number of objective function evaluations of all successful searches among the trial indicates efficiency, while the distribution of number of objective function evaluations of all successful searches among the trial indicates robustness.

The interaction between evolutionary algorithm, intrinsic and non-intrinsic control parameters has not been touched at parameter level evaluation. Evolutionary crimes are often observable within such evaluations[5]. From the point of view of practical application, such evaluation is not enough.

4.5.3 System level evaluation

At system level, instead, evolutionary algorithms are evaluated with many different combinations of control parameters. Evaluation of evolutionary algorithms at system level regards evolutionary algorithm, its intrinsic and non-intrinsic control parameters, and even problem features as a system. The evolutionary algorithm is not isolated. On the contrary, the interaction between evolutionary algorithms, intrinsic and non-intrinsic control parameters is systematically addressed. The ultimate goal here is to find the optimal intrinsic control parameter values and the relationship between evolution algorithm, intrinsic and non-intrinsic control parameters, and problem features. It is exactly the demand of application engineers.

However, by now, few people have paid attention to evaluation of evolutionary algorithms at system level.

At system level, many trials corresponding to different combinations of intrinsic control parameters, instead of one trial, are conducted. Attention is focused on successful trials. Under each population size, $40 \times 40 = 1{,}600$ trials corresponding to different combinations of F and C_r are conducted in this paper. The number of successful trials (NST) among the 1600 trials at each population size is recorded. The highest number of successful trials (HNST) among the NSTs of different population sizes is defined as the reliability indicator. By normalizing HNST to the total number of trials (TNT) under a fixed population size (TNT = 1,600 in this paper), the reliability indicator can also be expressed as the highest successful trail rate R_{HST} = HNST/TNT. The population size at which the HNST or R_{HST} is achieved is the optimal population size in terms of reliability. The minimal average number of objective function evaluations (MANOFE) among the ANOFEs of all successful trials at the optimal population size in terms of reliability is defined as the efficiency indicator. In this paper, no robustness indicator is defined since it is the least important performance for differential evolution at system level evaluation.

5 Numerical results

5.1 8-Dimensional benchmark functions

The reliability and efficiency of differential evolution for solving 8-dimensional benchmark functions and the corresponding optimal intrinsic control parameters (N_p, F, C_r,) are given in Table 1. The 12 participating DE algorithms are organized in 4 groups. Members in each group differ by differential mutation base. The best result in each group is highlighted in boldface and the best result of all participating differential evolution algorithms is highlighted in bold italic.

With regard to the reliability, DE/BoR/1/* is the most reliable among the three different differential mutation strategies in function $f3$. In function $f6$, DE/best/1/* is the most reliable, DE/BoR/1/* ranks second and DE/rand/1/* is the least reliable. In functions $f10$ and $f11$, DE/rand/1/* is a little bit more reliable than DE/BoR/1/*, and DE/best/1/* ranks the worst in terms of reliability. In the rest functions, the reliability of DE/BoR/1/* is close to that of DE/rand/1/*.

As for the efficiency, in functions $f1 \sim f6$ and $f8$, DE/best/1/* is the most efficient among the three different differential mutation strategies, and DE/BoR/1/* ranks second. In functions $f7$, $f9$, $f12$ and $f13$, the efficiency of DE/BoR/1/* is the best or close to the best among the three different differential mutation strategies. In function $f10$, CDE/rand/1/* is a little bit more efficient than CDE/BoR/1/* while DDE/rand/1/* is almost as efficient as DDE/BoR/1/*, and DE/best/1/* is the least efficient. In function $f11$, DE/best/1/* is the most efficient while the efficiency of DE/BoR/1/* is close to that of DE/rand/1*.

It is observed from Table 1 that the overall performance of DE/BoR/1/* is never the worst in each function. In general, DE/BoR/1/* is almost as reliable as but more efficient than DE/rand/1/*. As a whole, a better balance of reliability and efficiency is obtained in DE/BoR/1/* than in DE/best/1/* and DE/rand/1/*.

In order to find the relationship between DE algorithms and the mathematical features, such as modality, of the benchmark functions, the performance of DE algorithms with different differential mutation strategies in unimodal and multimodal functions are compared.

Table 1 Reliability and efficiency of differential evolution for 8-dimensional functions

		CBB	CRB	CBoB	CBE	CRE	CBoE	DBB	DRB	DBoB	DBE	DRE	DBoE
$f1$	R_{HST} (%)	68.1	**78.1**	75.8	83.1	**86.6**	84.2	**86.0**	77.8	76.9	***92.1***	86.8	85.3
	MANOFE	1789	2765	**1759**	**1624**	3517	2255	**779**	2716	1506	***707***	2669	1343
	N_p	48	32	**32**	56	40	40	56	40	40	72	40	40
	F	0.500	0.250	0.275	0.350	0.250	0.325	0.250	0.275	0.250	0.175	0.325	0.300
	C_r	0.925	0.750	0.850	0.925	0.975	1.000	0.975	0.775	0.725	1.000	1.000	1.000
$f2$	R_{HST} (%)	76.1	**83.9**	82.0	84.8	90.1	**90.5**	**86.8**	83.6	82.5	***92.9***	90.0	90.3
	MANOFE	**1159**	2902	1651	**1296**	3100	1854	***575***	1669	1104	**586**	2710	1384
	N_p	72	56	48	72	56	56	120	40	48	120	64	64
	F	0.375	0.225	0.275	0.450	0.025	0.250	0.050	0.225	0.325	0.100	0.250	0.275
	C_r	0.875	0.750	0.800	1.000	0.150	0.950	0.925	0.700	0.925	1.000	1.000	1.000
$f3$	R_{HST} (%)	9.3	21.1	**26.2**	11.9	18.8	**23.3**	11.7	22.9	***29.1***	15.9	21.1	**26.1**
	MANOFE	**4730**	19195	13851	**8716**	15031	14634	***1572***	23493	11518	**3728**	14231	7746
	N_p	32	32	40	64	24	40	56	48	48	48	24	32
	F	0.575	0.650	0.575	0.550	0.625	0.600	0.450	0.600	0.525	0.525	0.675	0.550
	C_r	0.950	1.000	0.975	0.975	0.925	1.000	0.850	0.925	1.000	1.000	1.000	0.975
$f4$	R_{HST} (%)	65.6	**76.8**	75.1	81.0	83.9	**84.3**	**84.1**	74.5	75.0	***91.1***	83.9	83.9
	MANOFE	**1971**	3159	1984	**1791**	4107	2552	**936**	2788	1380	***818***	3442	1978
	N_p	40	32	32	48	40	40	56	32	32	64	40	40
	F	0.475	0.225	0.325	0.375	0.225	0.350	0.275	0.350	0.350	0.175	0.275	0.400
	C_r	0.725	0.675	0.850	0.925	0.900	1.000	1.000	0.850	0.975	1.000	0.975	1.000
$f5$	R_{HST} (%)	**62.9**	60.9	61.0	**78.2**	74.1	74.3	**79.9**	60.8	62.8	***87.2***	74.8	76.2
	MANOFE	1837	2050	**1620**	**1622**	2950	2188	**959**	1913	1278	***842***	2707	1667
	N_p	32	16	16	48	24	24	40	16	24	40	24	32
	F	0.500	0.175	0.450	0.350	0.100	0.325	0.375	0.300	0.375	0.325	0.075	0.350
	C_r	0.700	0.125	0.725	0.925	0.075	0.825	0.975	0.425	0.825	0.975	0.175	0.950

Table 1 continued

		CBB	CRB	CBoB	CBE	CRE	CBoE	DBB	DRB	DBoB	DBE	DRE	DBoE
$f6$	R_{HST} (%)	**27.4**	19.6	22.9	**30.1**	23.2	28.5	***35.6***	21.8	27.4	**32.7**	22.9	28.4
	MANOFE	**2087**	4074	3109	**1994**	4103	2304	***1272***	3825	2652	**1867**	2775	2464
	N_p	24	16	16	16	16	16	24	16	16	16	8	16
	F	0.525	0.575	0.550	0.650	0.550	0.375	0.425	0.550	0.600	0.650	0.700	0.475
	C_r	0.800	0.925	0.775	0.975	0.975	0.850	0.825	0.925	0.975	0.975	0.875	0.850
$f7$	R_{HST} (%)	64.9	**75.4**	72.5	78.1	***86.6***	84.6	**78.3**	74.1	74.3	**86.1**	85.4	85.1
	MANOFE	2047	2363	**1632**	**2796**	3826	2822	***1355***	2109	1507	**1749**	2793	1997
	N_p	32	24	24	40	40	40	56	24	32	48	32	40
	F	0.300	0.200	0.175	0.325	0.125	0.200	0.300	0.225	0.200	0.350	0.200	0.225
	C_r	0.400	0.550	0.650	0.775	0.850	0.925	0.975	0.650	0.750	0.950	0.925	0.950
$f8$	R_{HST} (%)	47.4	64.8	**65.6**	66.1	80.9	***82.4***	55.8	62.7	**63.8**	71.4	81.3	**81.7**
	MANOFE	3834	**2154**	2508	**3002**	3276	3265	***1336***	2045	1740	**2798**	3234	2887
	N_p	80	24	48	64	40	56	56	24	32	80	48	48
	F	0.350	0.100	0.050	0.100	0.100	0.075	0.200	0.200	0.125	0.300	0.025	0.100
	C_r	0.500	0.075	0.200	0.375	0.425	0.550	0.525	0.300	0.350	0.750	0.050	0.625
$f9$	R_{HST} (%)	18.4	**34.2**	29.4	40.5	***63.3***	61.3	17.4	**33.9**	33.4	39.6	62.2	***63.2***
	MANOFE	6098	**4951**	5165	5912	5024	**4800**	7116	4632	**4342**	6631	4311	***4178***
	N_p	56	32	40	56	32	40	64	32	40	64	32	40
	F	0.275	0.150	0.150	0.325	0.125	0.125	0.225	0.150	0.100	0.375	0.125	0.100
	C_r	0.175	0.125	0.025	0.450	0.350	0.300	0.050	0.125	0.100	0.475	0.125	0.300
$f10$	R_{HST} (%)	6.8	**44.4**	37.9	20.0	**74.8**	68.1	6.8	**41.8**	**41.9**	22.4	***75.7***	71.8
	MANOFE	15992	***6328***	7213	25852	**8062**	9770	15744	**7728**	7882	19930	6620	**6480**
	N_p	72	40	56	140	56	72	72	48	64	120	48	64
	F	0.500	0.125	0.075	0.325	0.075	0.075	0.5009	0.100	0.150	0.500	0.075	0.225
	C_r	0.025	0.200	0.100	0.225	0.300	0.050	0.025	0.025	0.350	0.400	0.050	0.875

Table 1 continued

		CBB	CRB	CBoB	CBE	CRE	CBoE	DBB	DRB	DBoB	DBE	DRE	DBoE
$f11$	R_{HST} (%)	1.1	**8.8**	7.3	1.9	***19.8***	16.5	1.1	**8.8**	7.3	2.3	***19.7***	17.1
	MANOFE	19041	12689	**12554**	**7766**	10148	11231	10522	**9850**	11734	***4262***	11006	11700
	N_p	40	72	48	56	56	72	56	48	64	24	56	72
	F	0.775	0.250	0.275	0.275	0.325	0.225	0.400	0.300	0.300	0.450	0.325	0.250
	C_r	0.050	0.200	0.625	0.350	0.325	0.525	0.100	0.350	0.200	0.325	0.625	0.475
$f12$	R_{HST} (%)	56.8	**77.9**	**77.8**	69.3	81.1	***85.1***	57.4	**78.3**	77.1	70.8	81.4	***85.6***
	MANOFE	2755	3250	**2068**	2499	3714	**2175**	2162	2663	***1464***	2129	3480	**1690**
	N_p	64	40	40	64	40	40	80	40	40	80	48	48
	F	0.425	0.250	0.250	0.525	0.250	0.300	0.425	0.225	0.300	0.475	0.325	0.250
	C_r	0.550	0.700	0.800	1.000	0.950	0.975	0.675	0.725	0.975	0.950	1.000	0.950
$f13$	R_{HST} (%)	37.9	40.2	**40.8**	76.7	***83.6***	82.3	38.6	40.0	**40.5**	78.6	***83.4***	82.8
	MANOFE	3365	2307	**2274**	3261	**2329**	2349	3160	***2206***	2231	3142	2287	**2276**
	N_p	40	24	24	40	24	24	40	24	24	40	24	24
	F	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000
	C_r	0.100	0.050	0.050	0.200	0.025	0.075	0.150	0.075	0.025	0.200	0.075	0.050

In unimodal functions ($f1 \sim f7$), DDE/best/1/* is both most reliable and efficient in all the unimodal functions except $f3$ (Rosenbrock), in which DDE/best/1/* has better efficiency but worse reliability compared with DDE/rand/1/* and DDE/BoR/1/*. CDE/best/1/* also has best efficiency but not best reliability in most unimodal functions. As a whole, DE/best/1/* performs best among 3 different differential mutation strategies. DE/BoR/1/* ranks second. And DE/rand/1/* performs worst.

In multimodal functions ($f8 \sim f13$), DE/best/1/* performs worst. It is much less reliable than DE/rand/1/* and DE/BoR/1/*, and has no better efficiency. DE/rand/1/* is the most reliable while DE/BoR/1/* is the most efficient in general.

From the results of the unimodal and multimodal functions, it is evident that the modality of the function greatly affects the performance of DE algorithms with different differential mutation bases. Generally, DE/best/1/* performs better than DE/BoR/1/* and DE/rand/1/* in unimodal functions. While in multimodal functions better diversity is needed, hence DE/rand/1/* and DE/BoR/1/* perform better.

5.2 50-Dimensional benchmark functions

The reliability and efficiency of DE when solving 50-dimensional functions and the corresponding optimal intrinsic control parameters (N_p, F, C_r) are given in Table 2.

It is observed from Table 2 that the reliability of DE/BoR/1/* is close to that of the best one of the three different differential mutation strategies in all the functions except $f5$, in which DE/best/1/* is the most reliable and DE/BoR/1/* ranks second. In functions $f2$, $f8$, $f9$, $f10$ and $f12$, the reliability of DE/BoR/1/* and DE/rand/1/* is much better than that of DE/best/1/*.

As for the efficiency, in functions $f1$, $f4$ and $f5$, DE/BoR/1/* is almost as efficient as DE/best/1/* and more efficient than DE/rand/1/*. In functions $f2$, $f7$, $f10$ and $f12$, DE/BoR/1/* is the most efficient among the three different differential mutation strategies. In function $f8$, the efficiency of DE/BoR/1/* is close to that of DE/rand/1/* and better than that of DE/best/1/*. In functions $f9$, DE/rand/1/* is the most efficient among the three different differential mutation strategies and DE/BoR/1/* ranks second.

These simulation results show that DE algorithms with best of random differential mutation base generally perform better than those with other two differential mutation bases in 50-dimensional functions. The advantage of DE/BoR/1/* becomes more obvious in high dimensional functions.

The effect of modality on the performance of DE algorithms with different mutation bases has been observed again. In the uni-modal functions, though DE/best/1/* still performs better than DE/rand/1/*, it is not better than DE/BoR/1/*. In the multimodal functions, the performance of DE algorithms with 3 different differential mutation bases is similar to the case of dimension 8. DE/best/1/* performs worst. DE/rand/1/* is the most robust while DE/BoR/1/* is the most efficient.

5.3 Effects of error tolerance

In order to study the effect of error tolerance on the performance of different DE algorithms, a smaller error tolerance, i.e. 1e-5, is used to re-simulate the 8-dimensional Sphere ($f1$), Rosenbrock ($f3$), Rastrigin ($f9$) functions and 50-dimensional Sphere, Rastrigin functions. The limit of number of objective function evaluations under the error tolerance 1e-5 is the same as that under the error tolerance 1e-2 in the simple function $f1$. While it is twice as

Table 2 Reliability and efficiency of differential evolution for 50 -dimensional functions

		CBB	CRB	CBoB	CBE	CRE	CBoE	DBB	DRB	DBoB	DBE	DRE	DBoE
$f1$	R_{HST} (%)	**35.1**	33.8	34.6	72.8	75.1	**75.8**	**43.1**	34.0	39.3	74.4	75.3	***76.3***
	MANOFE	12851	15652	***10845***	21871	26497	**21240**	**10934**	15065	11221	19780	23549	**19476**
	N_p	30	30	20	40	40	40	30	30	30	40	40	40
	F	0.525	0.350	0.450	0.525	0.400	0.400	0.525	0.350	0.400	0.575	0.350	0.425
	C_r	0.375	0.325	0.275	0.900	0.950	0.925	0.450	0.300	0.450	0.950	0.925	0.950
$f2$	R_{HST} (%)	18.2	**35.1**	33.7	67.6	**79.5**	78.7	17.1	**34.1**	32.9	65.5	***80.3***	78.6
	MANOFE	14443	15476	**14163**	22733	27878	**18656**	17151	12883	***10832***	26340	23763	**15299**
	N_p	50	60	80	70	70	60	60	50	60	70	70	50
	F	0.475	0.250	0.250	0.525	0.275	0.350	0.475	0.275	0.300	0.575	0.325	0.325
	C_r	0.200	0.500	0.450	0.875	0.925	0.925	0.175	0.400	0.450	0.900	0.950	0.900
$f4$	R_{HST} (%)	**29.2**	23.4	27.6	**62.5**	58.4	60.6	**35.3**	24.0	31.9	***62.9***	58.2	60.5
	MANOFE	**21353**	26460	21530	31995	38118	**31187**	***19092***	26938	19285	31149	36620	**30516**
	N_p	20	20	20	20	20	20	20	20	20	20	20	20
	F	0.575	0.450	0.450	0.550	0.475	0.450	0.575	0.450	0.450	0.575	0.500	0.500
	C_r	0.325	0.325	0.325	0850	0.825	0.850	0.400	0.200	0.325	0.875	0.900	0.900
$f5$	R_{HST} (%)	**35.2**	9.1	20.1	**67.4**	56.3	62.6	**41.6**	10.4	24.2	***68.0***	55.5	62.8
	MANOFE	**18571**	29673	20452	26239	25281	**24048**	***15734***	27534	17610	24608	26119	**22834**
	N_p	30	20	20	30	20	20	30	20	20	30	20	20
	F	0.550	0.350	0.450	0.525	0.200	0.425	0.550	0.300	0.475	0.575	0.225	0.475
	C_r	0.350	0.025	0.275	0.875	0.175	0.825	0.525	0.025	0.400	0.925	0.175	0.900
$f7$	R_{HST} (%)	25.3	**29.1**	28.6	68.8	**73.0**	72.1	29.9	29.9	**32.7**	69.8	***73.8***	73.2
	MANOFE	19377	**15255**	16451	27415	26343	**25066**	20305	18239	***14835***	32556	31432	**27900**
	N_p	30	20	30	30	30	30	40	30	30	40	40	40
	F	0.500	0.375	0.350	0.400	0.300	0.325	0.475	0.300	0.350	0.450	0.275	0.300
	C_r	0.275	0.250	0.350	0.750	0.850	0.850	0.375	0.250	0.375	0.850	0.875	0.875

Table 2 continued

		CBB	CRB	CBoB	CBE	CRE	CBoE	DBB	DRB	DBoB	DBE	DRE	DBoE
$f8$	R_{HST} (%)	5.6	**10.4**	10.3	46.6	***67.8***	63.0	5.5	**10.9**	10.3	46.3	**66.6**	63.6
	MANOFE	47775	29968	**29388**	46548	**36793**	40816	41674	29811	***29150***	46568	39576	**38253**
	N_p	70	40	40	70	60	70	60	40	40	70	70	70
	F	0.375	0.325	0.375	0.350	0.125	0.100	0.400	0.325	0.375	0.400	0.075	0.125
	C_r	0.025	0.050	0.050	0.425	0.400	0.100	0.025	0.075	0.050	0.550	0.150	0.525
$f9$	R_{HST} (%)	0.8	**2.5**	2.3	14.4	***37.2***	35.2	0.8	**2.5**	2.3	15.1	***37.4***	35.7
	MANOFE	69015	**56199**	65632	63926	**54091**	61770	68554	**56066**	56647	62946	***53663***	61564
	N_p	40	40	50	50	40	50	40	40	40	50	40	50
	F	0.950	0.350	0.325	0.475	0.325	0.250	0.950	0.350	0.475	0.325	0.325	0.250
	C_r	0.025	0.025	0.025	0.075	0.100	0.150	0.025	0.025	0.025	0.025	0.050	0.125
$f10$	R_{HST} (%)	14.5	**41.9**	39.6	47.6	***81.8***	81.1	–	–	–	–	–	–
	MANOFE	22996	21863	***17913***	68828	28372	**27756**	–	–	–	–	–	–
	N_p	60	70	80	120	60	80	–	–	–	–	–	–
	F	0.475	0.225	0.275	0.300	0.275	0.300	–	–	–	–	–	–
	C_r	0.175	0.450	0.525	0.500	0.900	0.925	–	–	–	–	–	–
$f12$	R_{HST} (%)	20.9	**32.6**	32.1	65.5	72.8	***73.7***	20.4	31.8	**33.9**	64.7	***73.8***	73.4
	MANOFE	14253	16416	**12607**	25670	26333	**21629**	15586	17395	***11454***	24564	26610	**17731**
	N_p	30	40	40	50	50	50	40	50	40	50	50	40
	F	0.525	0.300	0.350	0.550	0.350	0.400	0.550	0.300	0.325	0.525	0.375	0.375
	C_r	0.200	0.350	0.450	0.900	0.950	0.950	0.300	0.525	0.400	0.875	0.950	0.925

many as that under the error tolerance 1e-2 in the more difficult functions $f3$ and $f9$. The reliability and efficiency of DE algorithms in these functions and the corresponding optimal intrinsic control parameters under the error tolerance 1e-5 are shown in Table 3.

It is observed from Table 3 that in 8-dimensional sphere function, DE/best/1/* has best reliability and efficiency among the three different differential mutation strategies under the error tolerance 1e-5. The reliability of DE/BoR/1/* is close to that of DE/rand/1/* but it is more efficient than DE/rand/1/*. In 50-dimensional sphere function, DE/best /1* is a little bit more reliable and efficient than DE/BoR/1/*, but the difference is small. DE/rand/1/* performs worst in this function. In general, the comparison results among the three different differential mutation strategies are similar to those under the error tolerance 1e-2. In function $f1$, the advantage of DE/best/1/* becomes a little bit more obvious under the error tolerance 1e-5.

In 8-dimensional function $f3$, DE/BoR/1/* performs best among the three differential mutation strategies under the error tolerance 1e-5. This result is similar to that under the error tolerance 1e-2. The performance of DE/best/1/* in function $f3$ deteriorate much under the error tolerance 1e-5. It is the least reliable and efficient among the three differential mutation strategies.

In function $f9$ with dimension 8 and 50, both the reliability and efficiency of DE/BoR/1/* are similar to those of DE/rand/1/*. DE/best/1/* performs worst in this function. These comparison results are consistent with those under the error tolerance 1e-2.

From these simulation results, it is observed that using the error tolerance 1e-5 in these functions generally doesn't change the comparison results of the DE algorithms with different differential mutation strategies under the error tolerance 1e-2. It appears that 1e-2 is also sufficient as the error tolerance in these functions.

6 Conclusions

A new differential mutation base generator for DE, namely best of random, is presented in this paper. The proposed differential mutation base generator uses the best one among several randomly chosen individuals as the differential mutation base while the other worse individuals as donors for vector differences. Thus both good diversity and fast evolution speed can be obtained in DE using the new differential mutation base. An evaluation platform for evolutionary algorithms including DE is built. With full consideration of the demand from practical applications, differential evolution algorithms, intrinsic and non-intrinsic control parameters, and problem features are treated as a system. A comprehensive comparative study over a set of benchmark functions has been conducted at system level instead of parameter level. The results show that a better balance of reliability and efficiency can be obtained in DE implementing the new generator of differential mutation base, especially in high dimensional functions.

Acknowledgments This work was supported by China Scholarship Council, the Ph. D Innovation Foundation of Southwest Jiaotong University (No. 2008-3), the National Natural Science Foundation of China (No. 60990320; 60990323; 10876029) and the National 863 Project of China under Grant No. 2009AA01Z230.

Table 3 Performance of differential evolution under the error tolerance 1e-5

		CBB	CRB	CBoB	CBE	CRE	CBoE	DBB	DRB	DBoB	DBE	DRE	DBoE
$f1$ 8D	R_{HST} (%)	63.6	**63.7**	62.1	74.8	**75.5**	73.4	**80.6**	62.5	61.4	***85.6***	74.7	74.0
	MANOFE	**2319**	4454	3032	**2431**	5608	3638	***1110***	3865	2383	**1116**	4725	2836
	N_p	40	32	32	40	40	40	48	32	32	48	40	40
	F	0.450	0.325	0.375	0.525	0.250	0.350	0.300	0.350	0.425	0.300	0.375	0.400
	C_r	0.675	0.625	0.700	0.975	0.800	0.925	1.000	0.625	0.825	1.000	0.975	0.975
$f1$ 50D	R_{HST} (%)	**31.4**	27.4	27.7	**69.3**	67.5	67.4	**38.5**	27.5	29.0	***70.1***	67.6	67.8
	MANOFE	**15111**	24337	19244	**25580**	30339	30660	***15055***	18806	15265	24591	29539	**24581**
	N_p	20	30	30	30	30	40	30	20	20	30	30	30
	F	0.550	0.400	0.375	0.550	0.375	0.450	0.550	0.450	0.475	0.575	0.425	0.425
	C_r	0.300	0.500	0.225	0.900	0.825	0.950	0.475	0.200	0.275	0.925	0.900	0.900
$f3$ 8D	R_{HST} (%)	0.9	22.9	**23.9**	2.6	20.5	**20.8**	0.9	24.4	***25.9***	2.7	21.5	**22.6**
	MANOFE	72301	44634	**34107**	107141	31042	**26514**	135240	35228	**24882**	105116	27864	***22239***
	N_p	80	64	72	160	40	48	120	56	64	160	40	48
	F	0.950	0.575	0.575	0.850	0.625	0.600	0.950	0.550	0.550	0.850	0.625	0.625
	C_r	0.950	1.000	1.000	0.850	1.000	0.975	0.775	1.000	1.000	0.850	1.000	1.000
$f9$ 8D	R_{HST} (%)	16.1	**34.5**	33.0	31.5	**62.5**	61.9	15.9	**35.3**	32.8	30.8	***62.9***	61.3
	MANOFE	12042	**10289**	10505	11729	11691	**10443**	11284	10243	**10223**	11657	***9862***	9864
	N_p	80	48	56	80	56	56	80	48	56	80	48	56
	F	0.325	0.200	0.200	0.250	0.200	0.200	0.250	0.225	0.200	0.275	0.200	0.175
	C_r	0.100	0.075	0.025	0.050	0.025	0.150	0.050	0.050	0.025	0.050	0.175	0.100
$f9$ 50D	R_{HST} (%)	1.4	**5.1**	4.4	27.5	**57.3**	55.9	1.4	**5.1**	4.7	27.0	***57.9***	56.4
	MANOFE	80297	81368	**78644**	116079	***77756***	86532	94612	80945	**79734**	115279	86865	**86606**
	N_p	50	50	50	80	50	60	60	50	50	80	60	60
	F	0.475	0.300	0.300	0.325	0.250	0.225	0.475	0.300	0.325	0.325	0.200	0.225
	C_r	0.025	0.025	0.025	0.150	0.050	0.225	0.025	0.025	0.025	0.225	0.200	0.100

Appendix

Benchmark functions in test bed

*f*1 *Sphere function*

The sphere function is continuous, differentiable, separable, scalable, uni-modal, and symmetric. It is the benchmark for studying problem features.

The sphere function reads

$$f(\mathbf{x}) = \sum_{i=1}^{N} x_i^2 \tag{4}$$

Its minimum is $f(\mathbf{x}^*) = 0$ at $\mathbf{x}^* = \mathbf{0}$. Standard search space is $\mathbf{x} \in [-500, 500]^N$

*f*2 *Step function 2*

The step function 2 is discontinuous, non-differentiable, separable, scalable, uni-modal, and symmetric. It is designed for studying the effect of continuity.

The step function 2 reads

$$f(\mathbf{x}) = \sum_{i=1}^{N} (\lfloor x_i + 0.5 \rfloor)^2 \tag{5}$$

Its minimum is $f(\mathbf{x}^*) = 0$ at $\mathbf{x}^* = \mathbf{0}$. Standard search space is $\mathbf{x} \in [-500, 500]^N$

*f*3 *Rosenbrock function*

The Rosenbrock function is continuous, differentiable, non-separable, scalable, uni-modal, and asymmetric. It reads

$$f(\mathbf{x}) = \sum_{i=1}^{N-1} \left[100 \left(x_{i+1} - x_i^2 \right)^2 + (x_i - 1)^2 \right] \tag{6}$$

Its minimum is $f(\mathbf{x}^*) = 0$ at $\mathbf{x}^* = \mathbf{1}$. Recommended search space is $\mathbf{x} \in [-100, 100]^N$

*f*4 *Hyper-ellipsoid function*

The hyper-ellipsoid function is continuous, differentiable, separable, scalable, uni-modal, and asymmetric. It is usually implemented to study the effect of symmetry.

The hyper-ellipsoid function reads

$$f(\mathbf{x}) = \sum_{i=1}^{N} 2^{i-1} x_i^2 \tag{7}$$

Its minimum is $f(\mathbf{x}^*) = 0$ at $\mathbf{x}^* = \mathbf{0}$. Standard search space is $\mathbf{x} \in [-500, 500]^N$

*f*5 *Qing function*

The Qing function is originally designed to study the effect of non-uniqueness. It is continuous, differentiable, separable, scalable, uni-modal, and asymmetric.

The Qing function reads

$$f(\mathbf{x}) = \sum_{i=1}^{N} \left(x_i^2 - i\right)^2 \tag{8}$$

Its minima are $f(\mathbf{x}^*) = 0$ at $x_i^* = \pm\sqrt{i}$. Standard search space is $\mathbf{x} \in [-500, 500]^N$

*f*6 *Schwefel function 1.2*

The Schwefel function 1.2 is a rotated sphere function. It is continuous, differentiable, non-separable, scalable, uni-modal, and asymmetric. The effect of symmetry and rotation can be investigated through it.

The Schwefel function 1.2 reads

$$f(\mathbf{x}) = \sum_{i=1}^{N} \left(\sum_{j=1}^{i} x_j \right)^2 \tag{9}$$

Its minimum is $f(\mathbf{x}^* = \mathbf{0}) = 0$. Standard search space is $\mathbf{x} \in [-500, 500]^N$

*f*7 *Schwefel function 2.22*

The Schwefel function 2.22 is continuous, non-differentiable, non-separable, scalable, uni-modal, and symmetric. It can be implemented to examine the effect of convexity because its landscape is non-convex.

The Schwefel function 2.22 reads

$$f(\mathbf{x}) = \sum_{i=1}^{N} |x_i| + \prod_{i=1}^{N} |x_i| \tag{10}$$

Its minimum is $f(\mathbf{x}^*) = 0$ at $\mathbf{x}^* = \mathbf{0}$. Standard search space is $\mathbf{x} \in [-500, 500]^N$

*f*8 *Schwefel function 2.26*

The Schwefel function 2.26 is continuous, non-differentiable, separable, scalable, multi-modal, and symmetric. The effect of differentiability and modality can be examined through it.

The Schwefel function 2.26 reads

$$f(\mathbf{x}) = -\frac{1}{N} \sum_{i=1}^{N} x_i \sin \sqrt{|x_i|} \tag{11}$$

Its minimum is $f(\mathbf{x}^*) = -418.983$ at $\mathbf{x}^* = \mathbf{420.968746}$. Recommended search space is $\mathbf{x} \in [-500, 500]^N$

*f*9 *Rastrigin function*

The Rastrigin function is continuous, differentiable, separable, scalable, multi-modal, and symmetric. It is very suitable for investigating the effect of modality.

The Rastrigin function reads

$$f(\mathbf{x}) = \sum_{i=1}^{N} \left[x_i^2 - 10\cos(2\pi x_i) + 10 \right] \tag{12}$$

Its minimum is $f(\mathbf{x}^*) = 0$ at $\mathbf{x}^* = \mathbf{0}$. Standard search space is $\mathbf{x} \in [-100, 100]^N$

*f*10 *Griewank function*

The Griewank function is continuous, differentiable, non-separable, scalable, multi-modal, and asymmetric. It is also difficult to optimize. Accordingly, it is a good candidate for study the effect of modality, separability, and symmetry.

The Griewank function reads

$$f(\mathbf{x}) = \frac{1}{4000} \sum_{i=1}^{N} x_i^2 - \prod_{i=1}^{N} \cos\left(x_i / \sqrt{i} \right) + 1 \tag{13}$$

Its minimum is $f(\mathbf{x}^*) = 0$ at $\mathbf{x}^* = \mathbf{0}$. Recommended search space is $\mathbf{x} \in [-500, 500]^N$

*f*11 *Whitley function*

The Whitley function is a composite function. It is continuous, differentiable, non-separable, scalable, multi-modal, and symmetric. It is very difficult to optimize. It is also used to study the effect of modality and separability.

The Whitley function reads

$$f(\mathbf{x}) = \sum_{i=1}^{N} \sum_{j=1}^{N} \left[\frac{y_{ji}^2}{4000} - \cos y_{ji} + 1 \right] \tag{14}$$

where $y_{ji} = 100\left(x_i - x_j^2 \right)^2 + \left(x_j - 1 \right)^2$.

Its minimum is $f(\mathbf{x}^*) = 0$ at $\mathbf{x}^* = \mathbf{1}$. Recommended search space is $\mathbf{x} \in [-500, 500]^N$

*f*12 *Ackley function*

The Ackley function is continuous, differentiable, non-separable, scalable, multi-modal, and symmetric. It is therefore suitable for investigating the effect of modality and separability.

The Ackley function reads

$$f(\mathbf{x}) = 20 - 20e^{-0.2\sqrt{\frac{1}{N}\sum_{i=1}^{N} x_i^2}} + e - e^{\frac{1}{N}\sum_{i=1}^{N} \cos(2\pi x_i)} \tag{15}$$

Its minimum is $f(\mathbf{x}^*) = 0$ at $\mathbf{x}^* = \mathbf{0}$, standard search space is $\mathbf{x} \in [-30, 30]^N$

f 13 *Weierstrass function*

The Weierstrass function is continuous, differentiable, separable, scalable, multi-modal, and symmetric. It reads

$$f(\mathbf{x}) = \sum_{i=1}^{N} \sum_{k=0}^{K_{\max}} a^k \cos\left[2\pi b^k (x_i + 0.5)\right] - N \sum_{k=0}^{K_{\max}} a^k \cos\left(\pi b^k\right)$$
$$a = 1, b = 1, K_{\max} = 10 \tag{16}$$

Its minimum is $f(\mathbf{x}^* = \mathbf{0}) = 0$. Recommended search space is $\mathbf{x} \in [-500, 500]^N$

References

1. Storn, R., Price, K.: Differential evolution—a simple and efficient adaptive scheme for global optimization over continuous spaces. Technical Report TR-95-012, International Computer Science Institute, Berkley, CA (1995)
2. Storn, R., Price, K.V.: Differential evolution—a simple and efficient heuristic for global optimization over continuous spaces. J. Glob. Optim. **11**(4), 341–359 (1997)
3. Price, K.V.: An introduction to differential evolution. In: Corn, D., Dorigo, M., Glover, F. (eds.) New Ideas in Optimization, chap. 6., pp. 79–108. McGraw-Hill, London (1999)
4. Price, K.V., Storn, R.M., Lampinen, J.A.: Differential Evolution: A Practical Approach to Global Optimization. Springer, Berlin (2005)
5. Qing, A.: Differential Evolution: Fundamentals and Applications in Engineering. Wiley, New York (2009)
6. Feoktistov, V.: Differential Evolution: In Search of Solutions. Springer, Berlin (2006)
7. Joshi, R., Sanderson, A.C.: Multisensor fusion and model selection using a minimal representation size framework. In: IEEE/SICE/RSJ International Conference on Multisensor Fusion Integration Intelligent Systems, Washington, DC, December 8–11, 1996, pp. 25–32 (1996)
8. Storn, R.: On the usage of differential evolution for function optimization. In: North American Fuzzy Information Processing Society Conference, Berkeley, pp. 519–523 (1996)
9. Qing, A.: Electromagnetic inverse scattering of multiple two-dimensional perfectly conducting objects by the differential evolution strategy. IEEE Trans. Antennas Propag. **51**(6), 1251–1262 (2003)
10. Vesterstrøm, J., Thomsen, R.: A comparative study of differential evolution, particle swarm optimization, and evolutionary algorithms on numerical benchmark problems. In: IEEE Congress Evolutionary Computation, Portland, OR, June 19–23, 2004, vol. 2, pp. 1980–1987 (2004)
11. Paterlini, S., Krink, T.: Differential evolution and particle swarm optimisation in partitional clustering. Comput. Stat. Data Anal. **50**(5), 1220–1247 (2006)
12. Wong, K.P., Dong, Z.Y.: Differential evolution, an alternative approach to evolutionary algorithm. In: 13th International Conference Intelligent Systems Application Power Systems. Arlington, VA, Nov. 6–10, pp. 73–83 (2005)
13. http://www.icsi.berkeley.edu/~storn/code.html
14. Qing, A.: A parametric study on differential evolution based on benchmark electromagnetic inverse scattering problem. In: IEEE Congress Evolutionary Computation, Singapore, Sept. 25–28, 2007, pp. 1904–1909 (2007)
15. Qing, A.: A study on base vector for differential evolution. IEEE World Congress Computational Intelligence/2008 IEEE Congress Evolutionary Computation, Hong Kong, June 1–6, 2008, pp. 550–556 (2008)
16. Qing, A.: Dynamic differential evolution strategy and applications in electromagnetic inverse scattering problems. IEEE Trans. Geosci. Remote Sens. **44**(1), 116–125 (2006)