International Journal of Applied Science and Engineering
2011. 9, 2: 123-141

# An Investigation and Improvement of the Performance of the GA–DE Hybrid Evolutionary Algorithm

Wen-Yi Lin[*]

*Department of Mechanical Engineering, De Lin Institute of Technology, No. 1 Lane 380, Qingyan Road, Tucheng, New Taipei City, Taiwan (R.O.C.)*

**Abstract:** Optimum dimensional synthesis of the five-point double-toggle mould clamping mechanism for thrust saving performance has been successfully solved using a genetic algorithm–differential evolution (GA–DE) method. To further validate the performances of the GA–DE algorithm in terms of its search ability (accuracy), efficiency, reliability and robustness across the widest possible range of functions, a test-suite of 20 functions of 1–20 variables discussed in the literature is performed. The premature convergence, related to the reliability performance, for the optimum dimensional synthesis problem using the GA–DE algorithm has not been investigated and improved in the previous work. Thus, a scheme of combining certain excellent individuals as the disturbed vectors and the technique of a large initial population size is proposed to improve the premature convergence for the GA–DE algorithm. The results obtained by the GA–DE algorithm are compared with those obtained by the other four evolutionary algorithms. Findings show that the GA–DE algorithm generally has very good search ability, efficiency and robustness. Lastly, it can also be seen that the proposed scheme can improve the reliability of the GA–DE algorithm for the test functions and the optimum dimensional synthesis problem.

**Keywords:** genetic algorithm; differential evolution; evolutionary algorithm; optimum synthesis; toggle clamping mechanism.

## 1. Introduction

Optimization techniques are very important in many fields. The well-known evolutionary algorithms, such as genetic algorithm (GA) developed by Holland [1] and differential evolution (DE) developed by Storn and Price [2], or well-known swarm intelligence methods such as particle swarm optimization (PSO) developed by Kennedy and Eberhart [3], have become increasingly popular for solving optimization problems. Evolutionary algorithms (evolutionary computation) also include evolutionary programming (EP) as developed by Fogel [4] and evolution strategies (ES) as pioneered by Rechenberg and further explored by Schwefel [5]. Evolutionary algorithms are based on the concepts of fitness and survival of the fittest. It has been shown that DE outperforms adaptive simulated annealing and ES [2]. It also outperforms PSO and GA in terms of reliability and robustness across all the test functions except two noisy functions [6]. The main technique in DE is the perturbation vector; i.e., differential vector produced by two random individuals added to

[*] Corresponding author; e-mail: wylin@mail.dlit.edu.tw

*Accepted for Publication: May 11, 2011*

the base vector. The differential vectors are self-adaptive in the search space. Furthermore, because the individuals are scattered throughout the search space in the early stage of searching, the magnitudes of the differential vectors are generally large. This is necessary in order to be able to explore a wide range. However, the magnitudes of the differential vectors generally become smaller in the later stage, because individuals cluster in certain special regions after repeated selection operations based on the survival of the fittest. In contrast, PSO is one of the swarm intelligence methods that imitate swarm behavior of natural creatures in order to construct optimization methods. PSO may be classified as a generalized evolutionary algorithm, because it utilizes the concept of fitness. However, the selection operation of the survival of the fittest is not utilized in PSO. A comprehensive survey of various swarm intelligence optimization methods (bees algorithm, artificial bee colony algorithm, ant colony optimization, PSO and artificial fish swarm algorithm) and several evolutionary algorithms (GA, ES and EP) has recently been carried out by Pham and Castellani [7]. A comparative study among the improved bees algorithms, artificial bee colony algorithm, evolutionary algorithm and PSO was also presented in the same publication.

A parametric study based on exhaustive search has been proposed in references [8,9] to accomplish the optimization design of the dimensional synthesis of the five-point double-toggle mould clamping mechanism (as shown in Fig. 1) with the performance of thrust saving for the prescribed input and output strokes, on condition that the overall horizontal length cannot exceed a prescribed value. The parametric study has the advantage that the individual influence of the geometric parameters and a deeper understanding of the mechanism might be obtained. Besides, the obtained results may be used to estimate the solutions using the other optimization methods. However, this kind of search requires a tremendous number of evaluations of the objective function, which amounts to 2,151,513 for 4 design parameters. Thus, Lin and Wang [10] applied a more efficient optimization method, termed the GA–DE hybrid algorithm, to solve the optimum dimensional synthesis problem. The GA–DE hybrid algorithm is obtained by combining the real-valued genetic algorithm with the method of differential evolution. The only difference between the GA–DE hybrid algorithm and real-valued genetic algorithm is in the content of the crossover. The crossover operation in the genetic algorithm is replaced by differential vector perturbation, with the best individual or some excellent individuals as the disturbed vectors. Thus, both the main perturbation of differential vectors and the minor perturbation of mutation are used as genetic operators in the GA–DE hybrid algorithm. The number of the disturbed vector and the definition of disturbing vectors (differential vectors) are the two significant differences between the GA–DE and DE algorithms. The number of disturbed objects for the GA–DE hybrid algorithm can be one, two, or more, while the number of base vectors for DE is only one. This may be helpful in overcoming premature convergence for some problems that use the GA–DE algorithm. On the other hand, for the same number of differential vectors, the range (or the degree of uniformity) of the differential vector distribution for the GA－DE hybrid algorithm may be better than that of DE, because of the definition of disturbing differential vectors.

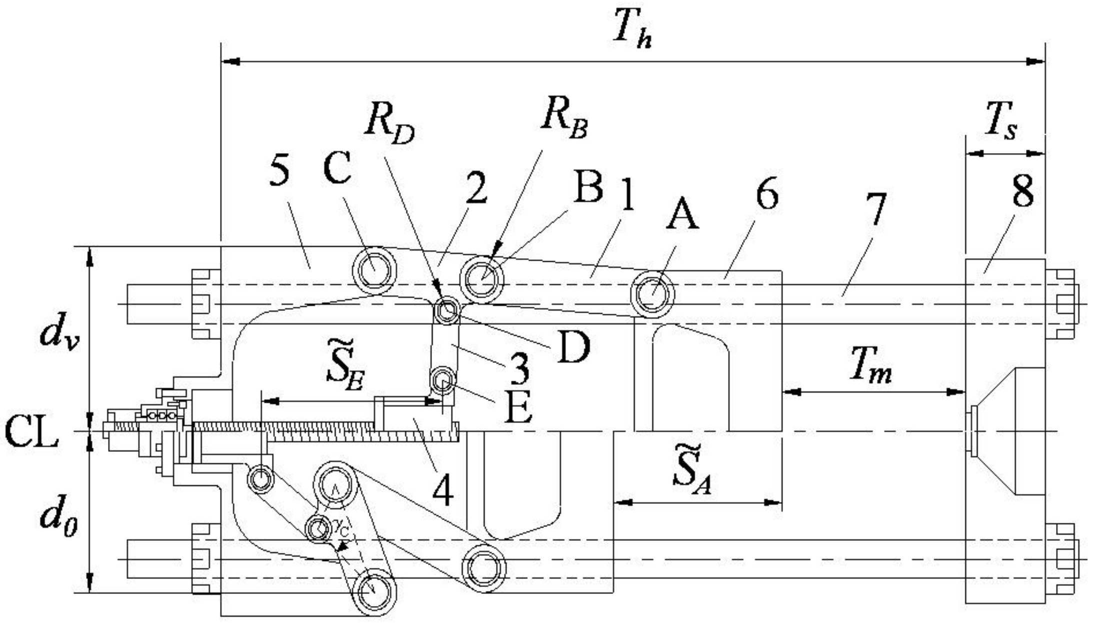

Figure 1. Conventional five-point double-toggle mould clamping mechanism

All optimization algorithms usually encounter the problem of premature convergence (falling into local extremes or undesired points) when used to solve difficult real-life or numerical problems such as high-dimensional and multimode objective functions (which have several local extremes). For example, the GA–DE hybrid algorithm employed by Lin and Wang [10] to solve the optimum dimensional synthesis of the five-point double-toggle mould clamping mechanism encountered the premature convergence problem. Therefore, 20 repeated runs were executed to find the best result. However, the premature convergence problem for the optimum dimensional synthesis has not been further investigated and improved. Andre *et al.* [11] proposed an enhanced binary-coded genetic algorithm (termed the EBGA) to fight premature convergence and to enhance the performance of the BGA. They also introduced criteria to evaluate the performances of an optimization algorithm. Intensive tests for the EBGA on more than 20 test functions of 1–20 variables each have been performed. Hrstka and Kučerová [12] used 20 test functions and the test methodology discussed by Andre *et al.* [11] to compare the performances of DE and SADE (a simplified adaptation of DE combing the features of DE with those of traditional genetic algorithms). Although it has been confirmed that real-coded algorithms (DE and SADE) generally exhibit better performance on real domains than binary-coded algorithms (BGA and EBGA), both DE and SADE also suffer the problem of premature convergence for some functions. To fight the premature convergence, an alternative SADE+CERAF (abbreviation of the French expression CEntre RAdioactiF) was presented in the same publication. The strategy produces areas of higher level of radioactivity in the neighborhood of all previously found local extremes by increasing the mutation probability (usually by 100%) in these areas many times. Findings have shown that the SADE+CERAF method can be completely free from premature convergence; that is, it is capable of solving all test functions over 100 repeated runs with a 100% success rate.

To further validate the performances of the GA–DE algorithm in terms of its search ability (accuracy), efficiency, reliability and robustness across the widest possible range of functions, a test-suite of 20 functions of 1–20 variables previously discussed in the literature is performed. The premature convergence, related to the reliability performance, for the optimum dimensional synthesis problem using the GA–DE algorithm has not been investigated and improved in the previous work. Thus, a scheme of combining certain excellent individuals as the disturbed vectors and the technique of a large initial population size is proposed to improve the premature convergence problem for the GA–DE algorithm. The results obtained by the GA–DE algorithm are compared with those obtained by the other four evolutionary algorithms. The criteria from Andre *et al.* [11] and Hrstka and Kučerová [12] are used to estimate the performance of the optimization algorithms, since this methodology minimizes the influence of random circumstances and the variable power of the computer used.

## 2. GA–DE hybrid evolutionary algorithm

A population with randomly generated chromosomes is initialized. Each chromosome (individual) is a candidate solution. The quality of the individual is estimated by the fitness value. Roulette-wheel selection [13] is employed to allow those individuals with higher fitness to have a higher chance of being selected. Thereafter, two individuals are paired randomly to be parents. Offspring are generated using genetic operators, differential vector perturbation and mutation, from either one or two individuals (parents). The technique of elitism [14] is employed.

Here, genes $x_i$ $(i = 1 - n)$ represent $n$ variables encoded in terms of real numbers that fall between their bounds. All genes are grouped in a vector **X** that represents a chromosome. That is

$$x_i \in [\min(x_i), \max(x_i)] \tag{1}$$

$$\mathbf{X} = [x_1, x_2, x_3, \ldots\ldots, x_n] \tag{2}$$

### 2.1. Initialization

An initial population with $N_p$ chromosomes is randomly generated. The gene in each chromosome is given by

$$x_i = \min(x_i) + r(\max(x_i) - \min(x_i)) \tag{3}$$

where $r$ is a random real number between 0 and 1.

### 2.2. Differential vector perturbations

The differential vector perturbation of DE, with the best individual or some excellent individuals as the disturbed vectors, is employed to replace the crossover operation in the real-coded genetic algorithm. Therefore, parents are used as differential vectors, not as crossover. All parents are randomly divided into $k$ groups corresponding to the top $k$ individuals, denoted by $\mathbf{X}_{top1}$, $\mathbf{X}_{top2} \cdots \mathbf{X}_{topk}$. These form the disturbed vectors. The value of $k$ is a user-defined integer. In the $i$-th group, the offspring may be generated by

$$\begin{aligned} \mathbf{X}_1 &= \mathbf{X}_{topi} + r_1(\mathbf{X}_{r1} - \mathbf{X}_{r2}) \\ \mathbf{X}_2 &= \mathbf{X}_{topi} - r_2(\mathbf{X}_{r1} - \mathbf{X}_{r2}) \end{aligned} \tag{4}$$

where $\mathbf{X}_{r1}$ and $\mathbf{X}_{r2}$ are the parents; $r_1$ and $r_2$ are random real numbers between 0 and 1 for each design variable. Random and different values for $r_1$ and $r_2$, in addition to some excellent individuals used as the disturbed vectors, may be considered another way of maintaining population diversity. A main perturbation rate $P_{m1}$ is defined as the ratio of the expected number of offspring generated by differential vector perturbations to the total population size.

### 2.3. Mutation

Here, mutation is performed to replace an entire, randomly selected chromosome with random real numbers (that are within the limits of variables). The minor perturbation rate $P_{m2}$, i.e., mutation rate, is defined as the ratio of the expected number of offspring introduced by mutation to the total population size.

## 3. Optimum synthesis problem

The conventional five-point double-toggle mould clamping mechanism shown in Fig. 1 is used for small- to middle-sized injection-moulding machines [15]. In Fig. 1, the upper portion above the centre-line (CL) illustrates the mould closing state, while the lower portion below the centre-line illustrates the initial state. Numerals 1 to 8 in Fig. 1 denote moving-platen side links (links 1), tailstock-platen side links (links 2), crosshead links (links 3), crosshead (4), tailstock platen (5), moving platen (6), tie bars (7) and stationary platen (8) respectively. Points A to E denote the centres of pin joints. Figure 2 depicts a kinematic stick diagram of the lower half of the conventional five-point double-toggle mould clamping mechanism. The solid lines denote the position during the mould closing operation and the dashed lines denote the initial position. In Fig. 2, $L_i$ ($i$ = 1 – 5) indicate the distances between points A and B, B and C, D and E, C and D and B and D respectively; $S_E$ and $S_A$ are the displacements of the crosshead and the moving platen respectively. The symbol $(\tilde{\ })$ denotes that the quantity in parentheses is in the final position state during the mould closing operation. The symbol $(\ )^0$ denotes that the quantity in parentheses is in the initial position state of the mould closing process. The superscript * is used only for the original mechanism [8].

### 3.1. Specified parameters

The following geometric and material-property parameters: $A_1$, $A_2$, $A_c$, $d_f$, $e_2$, $E_1$, $E_2$, $E_c$, $\tilde{h}_{bsE}$, $n_1$, $n_2$, $n_c$, $r_B$, $r_C$, $r_D$, $R_B$, $R_D$, $R_{bs}$, $R_o$, $T_m$, $T_s$, $\tilde{\phi} = 92^\circ$ and $L_1 + L_2 = 395$ are assumed to be fixed and their meanings and values can be found in the previous works of Lin and Hsiao [8] and Lin *et al.* [9]. The friction coefficient in the pin joints is assumed to be 0.1 and the final total deformational force of the tie bars $\tilde{F}_c = 539$ kN is considered. The input and output strokes are also prescribed: $\tilde{S}_A = \tilde{S}_A^*$= 180.67 mm and $\tilde{S}_E = \tilde{S}_E^*$=215.23 mm.

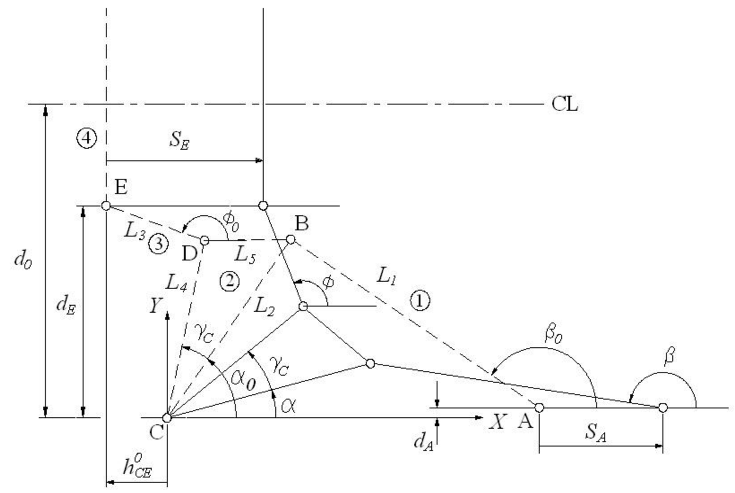

Figure 2. Geometry of the conventional toggle linkage during mould closing

## 3.2. Design variables

There are four design variables, $d_0$, $d_E$, $L_1/L_2$ and $L_4/L_2$, to be optimized. The offset $d_A$ is $d_0-(d_0^*-d_A^*)$ [9]. The other geometric parameters $\tilde{\alpha}$, $\tilde{\beta}$, $\alpha_0$, $\beta_0$, $\gamma_C$, $L_3$, $\phi_0$ and $L_5$ can be determined sequentially according to their kinematic relationships.

## 3.3. Design objective

The design objective is to minimize the necessary maximum value of the thrust applied to the crosshead during the mould clamping process, which may be expressed by

$$f_{obj}=F_{o,\max} \tag{5}$$

The elasto-static model proposed by Lin and Hsiao [8] is used to solve the thrust that must be applied to the crosshead of the clamping mechanism.

## 3.4. Constraints

Although there are only four design variables, this optimization problem is fairly difficult due to the considerable number of constraints (a total of 20) appearing at different stage of computation. These constraints arise from avoiding geometric interferences, unsound motion characteristic and initial transmission performance and the limitation of horizontal length of the mechanism. The detailed constraints and corresponding implications can be found in the previous works of Lin and Hsiao [8] and Lin *et al.* [9]. A proper and natural handling of these constraints has been presented in the previous works of Lin and Wang [10]. Two cases for the initial thrust applied the crosshead (i.e., for strict and loose initial transmission performance) are considered: (1) $F_o^0/F_{o,\max}^*\le 3.1\%$ and (2) $F_o^0/F_{o,\max}^*\le 6.0\%$

## 4. Performance indices and the scheme for improving the reliability performance

The algorithm should be adaptable the widest possible range of functions (robustness). Thus a test-suite of 20 functions with 1–20 variables, as previously discussed in the literature, is performed. For the sake of completeness, the test suite is listed in Appendix 1. A brief introduction including graphical illustrations with one or two variables of some of the test functions can be found in references [16,17]. For the low- and middle-dimensional functions (dimensions $n\le 10$), function F3 has 20 minima (17 local minima and 3 global minima); function Shubert has 760 minima (742 local minima and 18 global minima); the PShubert 1 and PShubert 2 functions are similar to the Shubert function; the Shekel 3 function has 10 minima (9 local minima and 1 global minimum); the number of minima of the other functions is not more than 7.

The criteria proposed by Andre *et al.* [11] and Hrstka and Kučerová [12] are used to estimate the performance of the optimization algorithms, since this methodology minimizes the influence of random circumstances and the varying power of different computers. Here, three performance indices are presented; that is, $f^g$ and $F_{o,\max}^g$ denoted the best function value at the end of evolution for test functions and the optimum dimensional synthesis, respectively, out of 100 repeated runs (accuracy), $N^c$ denoted the mean number of evaluations of the objective function needed to satisfy the convergence criterion (given below) for successful runs, out of 100 repeated runs (efficiency); and $R_s$ denoted the percentage of successful runs needed to satisfy the convergence criterion, out of 100 repeated runs (reliability).

The convergence criteria for test functions are as follows:

$$e_r=\left|(f-f_{std})/f_{std}\right|<1\% \quad \text{for} \quad f_{std}\ne 0 \tag{6}$$

$$e_a = |f| < 0.1 \quad \text{for} \quad f_{std} = 0 \qquad (7)$$

and for the optimum dimensional synthesis problem as follows:

$$e_a = |f| < 1.01 f_{std} \qquad (8)$$

where $e_r$ and $e_a$ are the relative and absolute errors, respectively; $f$ is the function value for the test function or the objective value ( $f_{obj}$) shown in Eq. (5); $f_{std}$ is the exact value of the global optimum for the test functions or the best objective value using the GA–DE algorithm with 20 population sizes as taken from Lin and Wang [10] for the optimum dimensional synthesis problem.

Unless otherwise specified, the user-supplied parameters of the GA–DE algorithm for the test functions are given as follows: population size is $N_p$= 10$n$ used by Hrstka and Kučerová [12] and 200 in Andre *et al.* [11]; maximum number of generations is 500; the major perturbation rate $P_{m1} = 0.6$; an the minor perturbation rate $P_{m2} = 0.01$.

The differences in the success rates using the GA–DE hybrid algorithm for some functions for different 100 repeated runs are fairly marked. This may be attributed the fact that an inappropriate or un-uniform distribution of the stochastic initial population and/or insufficient population size and/or the mechanisms of the algorithm does not prevail against the premature convergence for certain difficult cases arising from the stochastic process and the nature of the functions. Thus, a scheme of combining certain excellent individuals as the disturbed vectors and the techniques of a large initial population size $N_p^0 = 10 N_p$ and/or a population size of more than 10$n$ is proposed to account for such the phenomenon. For these functions, the success rate is estimated using 10 consecutive sets of 100 repeated runs each. Moreover, if the average number of evaluations of the function shown in Andre *et al.* [11] and Hrstka and Kučerová [12] is more than 5000$n$, the maximum number of generations does not exceed the quotient of the average number of evaluation of the function divided by the population size used for a fair comparison.

## 5. Results and discussion

### 5.1. Effect of the number of disturbed vectors

In order to show the effect that the number of disturbed vector (i.e., the value of $k$) has on premature convergence, a difficult problem (F10n) for the EBGA and SADE is tested, using the GA-DE hybrid algorithm with $k$ =1 – 10. Table 1 compares the mean number of evaluations of the function and the success rate versus the number of disturbed vector. Out of 100 repeated runs, the success rate is 41% for the best individual as the disturbed vector only and the algorithm spends very large iterations. In other words, premature convergence is encountered in 59 runs. In contrast, the success rate is equal to or greater than 98% using the GA–DE algorithm with $k \geq 4$. Therefore, premature convergence can be greatly improved for the GA–DE algorithm when certain excellent individuals are used as disturbed vectors. It also can be seen that the value of $N^c$ increases by about 50% between $k$ = 5 and $k$ = 10. This is due to the decrease in the number of disturbing differential vectors for $k$ = 10 for the same population size. Regardless, for this function, it should be adequate to have five excellent individuals as disturbed vector.

Table 1. Results for $k = 1 - 10$ for function F10n using the GA–DE hybrid algorithm

| $k$ | $N^c$ | $R_s$ (%) |
|---|---|---|
| 1 | 78327 | 41 |
| 2 | 23428 | 87 |
| 3 | 16893 | 93 |
| 4 | 19037 | 98 |
| 5 | 17855 | 99 |
| 6 | 17265 | 98 |
| 7 | 20095 | 99 |
| 8 | 23708 | 100 |
| 9 | 25986 | 99 |
| 10 | 26559 | 98 |

$N_p^0 = N_p$ =200

## 5.2. Techniques of a large initial population size and/or increasing population size

An added complication is that for some functions using the GA–DE algorithm with several excellent individuals as disturbed vectors, the obtained results remain unsatisfactory. For example, the results for Brown 1 using the GA-DE algorithm with $N_p^0 = N_p = 200$ and $k = 4$, repeated 10 times consecutively, is shown in the left panel of Table 2. The success rate varies from 91% to 72%. This is partly attributable to the fact that the GA–DE algorithm is sensitive to the distribution of the initial population when used with this function. Therefore, the quality of the solutions is dependent on the quality of the initial stochastic population; the mechanism of the algorithm does not prevail on its own. Thus, a technique designed to address a large initial population size ($N_p^0 = 10N_p = 2000$) is employed. The obtained results are shown in the right panel of Table 2. It can be seen that the success rate is improved significantly and stabilized, with only a 3% range of variation. On the other hand, this technique can also reduce the number of evaluations of the function.

Table 2. Results of 10 consecutive sets of 100 repeated runs each for function Brown 1 using the GA–DE hybrid algorithm with $N_p^0 = N_p$ and $N_p^0$ =10 $N_p$

| $N_p^0$ | $N^c$ | $R_s$ (%) | $N_p^0$ | $N^c$ | $R_s$ (%) |
|---|---|---|---|---|---|
| 200 | 12092 | 72 | 2000 | 14325 | 98 |
| 200 | 12352 | 91 | 2000 | 13512 | 98 |
| 200 | 12437 | 87 | 2000 | 14699 | 99 |
| 200 | 16441 | 87 | 2000 | 16029 | 97 |
| 200 | 16416 | 76 | 2000 | 13569 | 97 |
| 200 | 14055 | 87 | 2000 | 12905 | 97 |
| 200 | 15524 | 82 | 2000 | 15190 | 96 |
| 200 | 17063 | 86 | 2000 | 14325 | 96 |
| 200 | 12690 | 84 | 2000 | 13190 | 98 |
| 200 | 13750 | 84 | 2000 | 16622 | 99 |
| Average | 14282 | 84 | Average | 14437 | 98 |

$N_p$ =200, $k$=4

Unfortunately, this technique for addressing large initial population sizes for the GA–DE algorithm is not sufficient to stabilize the success rate of several other functions (i.e., Shekel 1, 2, and 3). It can be seen from the left panel of Table 3 that the success rate varies from 84% to 100% for function Shekel 2. This may be partly attributable to an insufficient population size. When the population size $N_p$ is increased from 40 to 80 (right panel of Table 3), the quality of the solutions can be improved. However, this also increases the required number of the evaluations of the function.

### 5.3. Performance comparisons for test functions

The success rate ($R_s$), the mean number of evaluations of the function ($N^c$) and the best function value ($f^g$) for all test functions using the GA–DE algorithm and PSO with CFA are shown in Tables 4, 5 and 6, respectively, together with the results discussed in the literature. The GA–DE algorithm has very good robustness and can adapt to all kinds of functions, as the success rates for all test functions exceed 90%. In addition, the GA–DE algorithm generally has very good efficiency, with the exception that a few functions (Shekel 1, 2 and 3) need more evaluations of the function using the GA–DE algorithm than they do when when using DE. The DE and SADE also have quite good robustness. CERAF has the best robustness; however, its efficiency performance is worst compared to the GA–DE and DE. The values of $N^c$ using CERAF for functions Hosc45, Brown 1, F10n and Hartman 2 are approximately 11, 11, 11 and 29 times those of using the GA–DE algorithm, respectively. The values of $N^c$ using the EBGA are about 9 to 222 times those of using the GA–DE algorithm for the 20 functions evaluated.

Table 3. Results of 10 consecutive sets of 100 repeated runs each for function Shekel 2 using the GA–DE hybrid algorithm with $N_p$=10*n* and $N_p$=20*n*

| $N_p$ | $N^c$ | $R_s$ (%) | $N_p$ | $N^c$ | $R_s$ (%) |
|---|---|---|---|---|---|
| 40 | 1858 | 93 | 80 | 3541 | 100 |
| 40 | 2187 | 90 | 80 | 3149 | 100 |
| 40 | 1987 | 84 | 80 | 3415 | 100 |
| 40 | 2164 | 100 | 80 | 5165 | 96 |
| 40 | 3972 | 92 | 80 | 2146 | 100 |
| 40 | 2193 | 86 | 80 | 3425 | 95 |
| 40 | 2732 | 99 | 80 | 4365 | 97 |
| 40 | 1348 | 100 | 80 | 4037 | 100 |
| 40 | 2776 | 93 | 80 | 3648 | 99 |
| 40 | 2252 | 100 | 80 | 2694 | 96 |
| Average | 2347 | 94 | Average | 3559 | 98 |

$N_p^0$=400, *k*=10

Except for Brown 1, the GA–DE algorithm can almost find the theoretical minima for all test functions. The relative error for Brown 1 using the GA–DE algorithm is about $5.8\times10^{-4}$. In contrast, the solution accuracy of the EBGA is relatively unsatisfactory, because the number of the functions for which the relative or absolute error is not less than $10^{-4}$ is 14. The solution accuracy for the other algorithms was not shown. The BGA is not the best choice for solving high-dimensional continuous optimization problems because the accuracy and the efficiency conflict. If a precision of five digits after the decimal point is required, the length of the chromosome for the BGA is 420 for functions F5n, F10n and F15n for 20 design variables with domains in the range [-10,10]. This generates a search space of about $10^{126}$. The BGA cannot simultaneously give accuracy and efficiency in such a situation. It can also be seen in the low-dimensional problems that the efficiency of the BGA is the worst, although the accuracy is fairly good. Therefore, it is not recommended that the BGA method is utilized to solve continuous optimization problems.

### 5.4. Improvement in reliability performance for the optimum synthesis problem

The scheme to improve the reliability performance of the GA–DE algorithm is also applied to the optimum dimensional synthesis problem. The success rate, efficiency and best objective value $F_{o,\max}^{g}$ using the GA–DE algorithm for $F_o^0 / F_{o,\max}^{*} \le 3.1\%$ (case 1) and $F_o^0 / F_{o,\max}^{*} \le 6.0\%$ (case 2) for the optimum synthesis problem are shown in Tables 7 and 8, respectively. It can be seen that the success rates for case 1 using $N_p^0 = 10\,N_p$ and $k$ = 4 can increase by about 103%, 156%, 76% and 53% for $N_p$ = 40, 80, 120 and 160, respectively, when compared to those using $N_p^0 = N_p$ and $k$ = 1. Similarly, the success rates for case 2 using $N_p^0 = 10\,N_p$ and $k$ = 4 can increase by about 189%, 126%, 77% and 37% for $N_p$ = 40, 80, 120 and 160, respectively, when compared to those using $N_p^0 = N_p$ and $k$ = 1. The combined effect of the two strategies not only increases the success rate, but also decreases the mean number of evaluations of the objective functions except for $N_p$ = 40 for case 1.

A total of 32 optimum synthesized results for each case are obtained using the GA–DE algorithm and a very high degree of repeatability for the 32 optimum synthesized results for each case is found. Representative optimum synthesized results obtained using the GA–DE algorithm for cases 1 and 2 are shown in Table 9.

## 6. Conclusions

The GA–DE algorithm has a very good mechanism of evolution designed to exploit and refine the given information, and gradually to find the optimal solution (or near optimal solution). However, the GA–DE algorithm sometimes suffers the premature convergence, arising from high population homogeneity, especially for the difficult problem. In the GA–DE algorithm, there are six ways to maintain the population diversity to explore all feasible regions: 1. a large initial population size; 2. increasing the population size; 3. using a mutation operator; 4. using random and different values for $r_1$ and $r_2$ in the disturbing vectors; 5. using certain excellent individuals as the disturbed vectors; 6. using disturbing vectors from randomly selected and non-repeated two individuals. The fourth to the sixth of these approaches also function as an evolutionary mechanism, which combines with the first to the sixth (provides the population diversity), in order to exploit and refine the given information and gradually to find the optimal solution.

Table 4. Comparison of the reliability between algorithms

| Function | $n$ | GA–DE | | | | EBGA | DE | SADE | CERAF |
|---|---|---|---|---|---|---|---|---|---|
| | | $N_p^0$ | $N_p$ | $k$ | $R_S$ (%) | $R_S$ (%) | $R_S$ (%) | $R_S$ (%) | $R_S$ (%) |
| F1 | 1 | 10 | 10 | 1 | 100 | 100 | 100 | 100 | 100 |
| F3 | 1 | 20 | 20 | 1 | 100 | 100 | 100 | 100 | 100 |
| Branin | 2 | 20 | 20 | 1 | 100 | 100 | 100 | 100 | 100 |
| Camelback | 2 | 20 | 20 | 1 | 100 | 100 | 100 | 100 | 100 |
| Goldprice | 2 | 20 | 20 | 1 | 100 | 100 | 100 | 100 | 100 |
| PShubert 1 | 2 | 60 | 60 | 4 | 95 | 100 | 83 | 100 | 100 |
| PShubert 2 | 2 | 60 | 60 | 4 | 98 | 100 | 90 | 100 | 100 |
| Quartic | 2 | 20 | 20 | 1 | 100 | 100 | 97 | 100 | 100 |
| Shubert | 2 | 20 | 20 | 1 | 100 | 100 | 94 | 100 | 100 |
| Hartman 1 | 3 | 30 | 30 | 1 | 100 | 100 | 100 | 100 | 100 |
| Shekel 1 | 4 | 400 | 80 | 10 | 95 | 97 | 72 | 99 | 100 |
| Shekel 2 | 4 | 400 | 80 | 10 | 98 | 98 | 91 | 100 | 100 |
| Shekel 3 | 4 | 400 | 80 | 10 | 98 | 100 | 89 | 99 | 100 |
| Hartman 2 | 6 | 60 | 60 | 1 | 100 | 92 | 16 | 67 | 100 |
| Hosc 45 | 10 | 50 | 50 | 1 | 100 | 2 | 100 | 100 | 100 |
| Brown 1 | 20 | 2000 | 200 | 4 | 98 | 0 | 100 | 95 | 100 |
| Brown 3 | 20 | 200 | 200 | 4 | 100 | 5 | 100 | 100 | 100 |
| F5n | 20 | 200 | 200 | 5 | 100 | 100 | 96 | 66 | 100 |
| F10n | 20 | 200 | 200 | 5 | 99 | 49 | 90 | 47 | 100 |
| F15n | 20 | 200 | 200 | 5 | 100 | 100 | 100 | 93 | 100 |

Table 5. Comparison of the efficiency between algorithms

| Function | $n$ | GA–DE | | | | EBGA | DE | SADE | CERAF |
|---|---|---|---|---|---|---|---|---|---|
| | | $N_p^0$ | $N_p$ | $k$ | $N^c$ | $N^c$ | $N^c$ | $N^c$ | $N^c$ |
| F1 | 1 | 10 | 10 | 1 | 20 | 784 | 52 | 72 | 72 |
| F3 | 1 | 20 | 20 | 1 | 40 | 744 | 98 | 88 | 88 |
| Branin | 2 | 20 | 20 | 1 | 160 | 2040 | 506 | 478 | 478 |
| Camelback | 2 | 20 | 20 | 1 | 60 | 1316 | 244 | 273 | 273 |
| Goldprice | 2 | 20 | 20 | 1 | 140 | 4632 | 350 | 452 | 452 |
| PShubert 1 | 2 | 60 | 60 | 4 | 1232 | 8853 | 1342 | 2738 | 2388 |
| PShubert 2 | 2 | 60 | 60 | 4 | 835 | 4116 | 908 | 1033 | 1014 |
| Quartic | 2 | 20 | 20 | 1 | 144 | 3168 | 313 | 425 | 425 |
| Shubert | 2 | 20 | 20 | 1 | 234 | 2364 | 10098 | 585 | 585 |
| Hartman 1 | 3 | 30 | 30 | 1 | 60 | 1680 | 284 | 464 | 464 |
| Shekel 1 | 4 | 400 | 80 | 10 | 3230 | 36388 | 1968 | 61243 | 3942 |
| Shekel 2 | 4 | 400 | 80 | 10 | 3559 | 36774 | 1851 | 17078 | 3746 |
| Shekel 3 | 4 | 400 | 80 | 10 | 4053 | 36772 | 1752 | 11960 | 3042 |
| Hartman 2 | 6 | 60 | 60 | 1 | 540 | 53792 | 4241 | 2297 | 15396 |
| Hosc 45 | 10 | 50 | 50 | 1 | 568 | 126139 | 1174 | 6438 | 6438 |
| Brown 1 | 20 | 2000 | 200 | 4 | 14437 | — | 65346 | 163919 | 137660 |
| Brown 3 | 20 | 200 | 200 | 4 | 6612 | 106859 | 41760 | 43426 | 43426 |
| F5n | 20 | 200 | 200 | 5 | 8772 | 99945 | 38045 | 17785 | 20332 |
| F10n | 20 | 200 | 200 | 5 | 17855 | 113929 | 71631 | 110593 | 200136 |
| F15n | 20 | 200 | 200 | 5 | 10870 | 102413 | 44248 | 28223 | 31574 |

Table 6. Comparison of the accuracy between algorithms

| Function | $n$ | Theoretical minimum | GA–DE | EBGA |
|---|---|---|---|---|
| | | | $f^g$ | $f^g$ |
| F1 | 1 | -1.12323 | -1.12323 | -1.12323 |
| F3 | 1 | -12.03125 | -12.03125 | -12.03120 |
| Branin | 2 | 0.39789 | 0.39789 | 0.39791 |
| Camelback | 2 | -1.03163 | -1.03163 | -1.03163 |
| Goldprice | 2 | 3 | 3 | 3.00028 |
| PShubert 1 | 2 | -186.73091 | -186.73091 | -186.68574 |
| PShubert 2 | 2 | -186.73091 | -186.73091 | -186.70469 |
| Quartic | 2 | -0.35239 | -0.35239 | -0.35238 |
| Shubert | 2 | -186.73091 | -186.73091 | -186.72802 |
| Hartman 1 | 3 | -3.86278 | -3.86278 | -3.86114 |
| Shekel 1 | 4 | -10.15320 | -10.15320 | -10.14866 |
| Shekel 2 | 4 | -10.40294 | -10.40294 | -10.38253 |
| Shekel 3 | 4 | -10.53641 | -10.53641 | -10.51404 |
| Hartman 2 | 6 | -3.32237 | -3.32237 | -3.31383 |
| Hosc 45 | 10 | 1 | 1 | 1.00943 |
| Brown 1 | 20 | 2 | 2.00115 | 8.55162 |
| Brown 3 | 20 | 0 | 0 | 0.67464 |
| F5n | 20 | 0 | $4.22892\times10^{-14}$ | 0.00221 |
| F10n | 20 | 0 | $6.10182\times10^{-13}$ | 0.04960 |
| F15n | 20 | 0 | $4.25771\times10^{-14}$ | 0.00342 |

Table 7. Improvement in the reliability performance of the GA–DE for $F_o^0 \big/ F_{o,\max}^* \le 3.1\%$ for the optimum synthesis problem

| GA–DE with $N_p^0 = N_p$ | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | $N_p^0 = N_p = 40$ | | | $N_p^0 = N_p = 80$ | | | $N_p^0 = N_p = 120$ | | | $N_p^0 = N_p = 160$ | | |
| | $N^c$ | $R_s$ (%) | $F_{o,\max}^g$ | $N^c$ | $R_s$ (%) | $F_{o,\max}^g$ | $N^c$ | $R_s$ (%) | $F_{o,\max}^g$ | $N^c$ | $R_s$ (%) | $F_{o,\max}^g$ |
| $k=1$ | 371 | 39 | 13.94 | 1431 | 36 | 13.93 | 3260 | 54 | 13.94 | 2280 | 64 | 13.93 |
| $k=2$ | 869 | 36 | 13.94 | 2130 | 48 | 13.94 | 1495 | 57 | 13.93 | 1783 | 76 | 13.93 |
| $k=3$ | 385 | 31 | 13.94 | 2105 | 44 | 13.94 | 1560 | 72 | 13.93 | 1655 | 82 | 13.93 |
| $k=4$ | 834 | 47 | 13.94 | 1209 | 52 | 13.94 | 2904 | 64 | 13.93 | 1683 | 71 | 13.93 |
| GA–DE with $N_p^0 = 10N_p$ | | | | | | | | | | | | |
| | $N_p^0 = 400, N_p = 40$ | | | $N_p^0 = 800, N_p = 80$ | | | $N_p^0 = 1200, N_p = 120$ | | | $N_p^0 = 1600, N_p = 160$ | | |
| | $N^c$ | $R_s$ (%) | $F_{o,\max}^g$ | $N^c$ | $R_s$ (%) | $F_{o,\max}^g$ | $N^c$ | $R_s$ (%) | $F_{o,\max}^g$ | $N^c$ | $R_s$ (%) | $F_{o,\max}^g$ |
| $k=1$ | 772 | 54 | 13.94 | 1582 | 76 | 13.93 | 1755 | 80 | 13.93 | 2698 | 87 | 13.93 |
| $k=2$ | 786 | 73 | 13.93 | 1206 | 82 | 13.93 | 1687 | 90 | 13.93 | 2882 | 93 | 13.93 |
| $k=3$ | 898 | 73 | 13.93 | 1220 | 86 | 13.93 | 2003 | 91 | 13.93 | 2169 | 90 | 13.93 |
| $k=4$ | 825 | 79 | 13.93 | 1202 | 92 | 13.93 | 1834 | 95 | 13.93 | 2121 | 98 | 13.93 |

$f_{std}$ =13.94 kN, $F_{o,\max}^g$ in kN.

**Table 8.** Improvement in the reliability performance of the GA–DE for $F_o^0 / F_{o,\max}^* \le 6.0\%$ for the optimum synthesis problem

| GA–DE with $N_p^0 = N_p$ | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | $N_p^0 = N_p = 40$ | | | $N_p^0 = N_p = 80$ | | | $N_p^0 = N_p = 120$ | | | $N_p^0 = N_p = 160$ | | |
| | $N^c$ | $R_s$ (%) | $F_{o,\max}^g$ | $N^c$ | $R_s$ (%) | $F_{o,\max}^g$ | $N^c$ | $R_s$ (%) | $F_{o,\max}^g$ | $N^c$ | $R_s$ (%) | $F_{o,\max}^g$ |
| $k$ =1 | 1053 | 19 | 11.76 | 3406 | 35 | 11.75 | 8649 | 53 | 11.75 | 6096 | 68 | 11.75 |
| $k$ =2 | 842 | 21 | 11.75 | 3532 | 45 | 11.75 | 3478 | 69 | 11.75 | 4739 | 79 | 11.75 |
| $k$ =3 | 787 | 33 | 11.76 | 2057 | 53 | 11.75 | 4742 | 80 | 11.75 | 4424 | 86 | 11.75 |
| $k$ =4 | 690 | 28 | 11.75 | 2708 | 61 | 11.75 | 3279 | 73 | 11.75 | 4800 | 87 | 11.75 |
| GA–DE with $N_p^0 = 10\,N_p$ | | | | | | | | | | | | |
| | $N_p^0 = 400, N_p = 40$ | | | $N_p^0 = 800, N_p = 80$ | | | $N_p^0 =$ 1200, $N_p =$ 120 | | | $N_p^0 =$ 1600, $N_p =$ 160 | | |
| | $N^c$ | $R_s$ (%) | $F_{o,\max}^g$ | $N^c$ | $R_s$ (%) | $F_{o,\max}^g$ | $N^c$ | $R_s$ (%) | $F_{o,\max}^g$ | $N^c$ | $R_s$ (%) | $F_{o,\max}^g$ |
| $k$ =1 | 1125 | 42 | 11.75 | 2495 | 61 | 11.75 | 3528 | 73 | 11.75 | 8249 | 81 | 11.75 |
| $k$ =2 | 620 | 51 | 11.75 | 2526 | 73 | 11.75 | 3612 | 88 | 11.75 | 4951 | 87 | 11.75 |
| $k$ =3 | 1033 | 54 | 11.75 | 1866 | 73 | 11.75 | 2400 | 85 | 11.75 | 3955 | 96 | 11.75 |
| $k$ =4 | 748 | 55 | 11.75 | 1961 | 79 | 11.75 | 2504 | 94 | 11.75 | 2557 | 93 | 11.75 |

$f_{std}$ =11.78 kN, $F_{o,\max}^g$ in kN.

**Table 9.** Synthesized results for the optimum synthesis problem

| $F_o^0 / F_{o,\max}^* \le 3.1\%$ ($\tilde{S}_A = \tilde{S}_A^*$ and $\tilde{S}_E = \tilde{S}_E^*$) | | | | | | | |
|---|---|---|---|---|---|---|---|
| $d_0$ | $d_A$ | $d_E$ | $L_1/L_2$ | $L_4/L_2$ | $\gamma_C\,(\circ)$ | $L_3$ | $F_{o,\max}^g$ (kN) |
| GA-DE hybrid algorithm with $N_p = 80$ | | | | | | | |
| 253.654 | 23.6535 | 151.272 | 1.0226 | 1.0 | 20.9467 | 70.7017 | 13.9340 |
| $F_o^0 / F_{o,\max}^* \le 6.0\%$ ($\tilde{S}_A = \tilde{S}_A^*$ and $\tilde{S}_E = \tilde{S}_E^*$) | | | | | | | |
| $d_0$ | $d_A$ | $d_E$ | $L_1/L_2$ | $L_4/L_2$ | $\gamma_C\,(\circ)$ | $L_3$ | $F_{o,\max}^g$ (kN) |
| GA-DE hybrid algorithm with $N_p = 80$ | | | | | | | |
| 273.393 | 43.393 | 178.393 | 0.729117 | 0.689197 | 0.0 | 161.195 | 11.7483 |

$L_1 + L_2 = 395$ mm

## Appendix 1: List of test functions

1. Function F1 with n = 1:

$$f(x) = 2(x-0.75)^2 + \sin(5\pi x - 0.4\pi) - 0.125,$$

where $0 \le x \le 1$.

2. Function F3 with n = 1:

$$f(x) = -\sum_{j=1}^{5}[j\sin[(j+1)x+j]],$$

where $-10 \le x \le 10$.

3. Function Branin with n = 2:

$$f(x,y) = a(y-bx^2+cx-d)^2 + h(1-f)\cos x + h,$$

where $a=1, b=5.1/4\pi^2, c=5/\pi, d=6, h=10, f=1/8\pi$, $-5 \le x \le 10, 0 \le y \le 15$.

4. Function Camelback with n = 2:

$$f(x,y) = (4-2.1x^2+\frac{x^4}{3})x^2 + xy + (-4+4y^2)y^2,$$

where $-3 \le x \le 3, -2 \le y \le 2$.

5. Function Goldprice with n = 2:

$$f(x,y) = [1+(x+y+1)^2(19-14x+3x^2-14y+6xy+3y^2)]$$
$$[30+(2x-3y)^2(18-32x+12x^2+48y-36xy+27y^2)],$$

where $-2 \le x \le 2, -2 \le y \le 2$.

6. Functions PShubert 1 and 2 with n = 2:

$$f(x,y) = \{\sum_{i=1}^{5} i\cos[(i+1)x+i]\}\{\sum_{i=1}^{5} i\cos[(i+1)y+i]\}$$
$$+\beta[(x+1.42513)^2+(y+0.80032)^2],$$

where $-10 \le x \le 10, -10 \le y \le 10$, for PShubert 1: $\beta = 0.5$ for PShubert 2: $\beta = 1.0$.

7. Function Quartic with n = 2:

$$f(x,y) = \frac{x^4}{4} - \frac{x^2}{2} + \frac{x}{10} + \frac{y^2}{2},$$

where $-10 \le x \le 10, -10 \le y \le 10$.

8. Function Shubert with n = 2:

$$f(x,y) = \{\sum_{i=1}^{5} i\cos[(i+1)x+i]\}\{\sum_{i=1}^{5} i\cos[(i+1)y+i]\},$$

where $-10 \le x \le 10, -10 \le y \le 10$.

9. Function Hartman1 with n = 3:

$$f(x_1, x_2, x_3) = -\sum_{i=1}^{4} c_i \exp(-\sum_{j=1}^{3} a_{ij}(x_j - p_{ij})^2),$$

where $0 \le x_j \le 1$, $j = 1, \ldots, 3$ $x = (x_1, \ldots, x_3)$, $p_i = (p_{i1}, \ldots, p_{i3})$, $a_i = (a_{i1}, \ldots, a_{i3})$.

| $i$ | $a_{ij}$ | | | $c_i$ | $p_{ij}$ | | |
|---|---|---|---|---|---|---|---|
| 1 | 3.0 | 10.0 | 30.0 | 1.0 | 0.36890 | 0.1170 | 0.2673 |
| 2 | 0.1 | 10.0 | 35.0 | 1.2 | 0.46990 | 0.4387 | 0.7470 |
| 3 | 3.0 | 10.0 | 30.0 | 3.0 | 0.10910 | 0.8732 | 0.5547 |
| 4 | 0.1 | 10.0 | 35.0 | 3.2 | 0.03815 | 0.5743 | 0.8828 |

10. Functions Shekel 1, 2 and 3 with n = 4:

$$f(\mathbf{x}) = -\sum_{i=1}^{m} \frac{1}{(\mathbf{x} - \mathbf{a}_i)^{\mathbf{T}}(\mathbf{x} - \mathbf{a}_i) + \mathbf{c}_i},$$

where $0 \le x_j \le 10$, for Shekel 1: $m = 5$, for Shekel 2: $m = 7$, for Shekel 3: $m = 10$.

$\mathbf{x} = (x_1, x_2, x_3, x_4)^{\mathbf{T}}$, $\mathbf{a}_i = (a_{i1}, a_{i2}, a_{i3}, a_{i4})^{\mathbf{T}}$.

| $i$ | $a_{ij}$ | | | | $c_i$ |
|---|---|---|---|---|---|
| 1 | 4.0 | 4.0 | 4.0 | 4.0 | 0.1 |
| 2 | 1.0 | 1.0 | 1.0 | 1.0 | 0.2 |
| 3 | 8.0 | 8.0 | 8.0 | 8.0 | 0.2 |
| 4 | 6.0 | 6.0 | 6.0 | 6.0 | 0.4 |
| 5 | 3.0 | 7.0 | 3.0 | 7.0 | 0.4 |
| 6 | 2.0 | 9.0 | 2.0 | 9.0 | 0.6 |
| 7 | 5.0 | 5.0 | 3.0 | 3.0 | 0.3 |
| 8 | 8.0 | 1.0 | 8.0 | 1.0 | 0.7 |
| 9 | 6.0 | 2.0 | 6.0 | 2.0 | 0.5 |
| 10 | 7.0 | 3.6 | 7.0 | 3.6 | 0.5 |

11. Function Hartman 2 with n = 6:

$$f(x_1, \ldots, x_6) = -\sum_{i=1}^{4} c_i \exp(-\sum_{j=1}^{6} a_{ij}(x_j - p_{ij})^2),$$

where $0 \le x_j \le 1$, $j = 1, \ldots, 6$.

$x = (x_1, \ldots, x_6)$, $p_i = (p_{i1}, \ldots, p_{i6})$, $a_i = (a_{i1}, \ldots, a_{i6})$.

| $i$ | $a_{ij}$ | | | | | | $c_i$ |
|---|---|---|---|---|---|---|---|
| 1 | 10.00 | 3.00 | 17.00 | 3.50 | 1.70 | 8.00 | 1.0 |
| 2 | 0.05 | 10.00 | 17.00 | 0.10 | 8.00 | 14.00 | 1.2 |
| 3 | 3.00 | 3.50 | 1.70 | 10.00 | 17.00 | 8.00 | 3.0 |
| 4 | 17.00 | 8.00 | 0.05 | 10.00 | 0.01 | 14.00 | 3.2 |

| $i$ | $p_{ij}$ | | | | | |
|---|---|---|---|---|---|---|
| 1 | 0.1312 | 0.1696 | 0.5569 | 0.0124 | 0.8283 | 0.5886 |

| | | | | | | |
|---|---|---|---|---|---|---|
| 2 | 0.2329 | 0.4135 | 0.8307 | 0.3736 | 0.1004 | 0.9991 |
| 3 | 0.2348 | 0.1451 | 0.3522 | 0.2883 | 0.3047 | 0.6650 |
| 4 | 0.4047 | 0.8828 | 0.8732 | 0.5743 | 0.1091 | 0.0381 |

12. Function Hosc 45 with n = 10:

$$f(x)=2-\frac{1}{n!}\prod_{i=1}^{n}x_i,$$

where $x=(x_1,\ldots,x_n)$, $0\le x_i\le i$, $n=10$.

13. Function Brown 1 with n = 20:

$$f(x)=[\sum_{i\in J}(x_i-3)]^2+\sum_{i\in J}[10^{-3}(x_i-3)^2-(x_i-x_{i+1})+e^{20(x_i-x_{i+1})}],$$

where $J=\{1,3,\ldots,19\}$, $-1\le x_i\le 4$, $1\le i\le 20$, $x=(x_1,\ldots,x_{20})^{\mathbf{T}}$.

14. Function Brown 3 with n = 20:

$$f(x)=\sum_{i=1}^{19}\left[(x_i^2)^{(x_{i+1}^2+1)}+(x_{i+1}^2)^{(x_i^2+1)}\right]$$

$x=(x_1,\ldots,x_{20})^{\mathbf{T}}$, $-1\le x_i\le 4$, $1\le i\le 20$.

15. Function F5n with n = 20:

$$f(x)=(\pi/20)\{10\sin^2(\pi y_1)+\sum_{i=1}^{19}[(y_i-1)^2(1+10\sin^2(\pi y_{i+1}))]+(y_{20}-1)^2\}$$

where $x=(x_1,\ldots,x_{20})^{\mathbf{T}}$, $-10\le x_i\le 10$, $y_i=1+0.25(x_i-1)$.

16. Function F10n with n = 20:

$$f(x)=(\pi/20)\{10\sin^2(\pi x_1)+\sum_{i=1}^{19}[(x_i-1)^2(1+10\sin^2(\pi x_{i+1}))]+(x_{20}-1)^2\}$$

where $x=(x_1,\ldots,x_{20})^{\mathbf{T}}$, $-10\le x_i\le 10$.

17. Function F15n with n = 20:

$$f(x)=(1/10)\{\sin^2(3\pi x_1)+\sum_{i=1}^{19}[(x_i-1)^2(1+\sin^2(3\pi x_{i+1}))]$$
$$+(1/10)(x_{20}-1)^2[1+\sin^2(2\pi x_{20})]\}$$

where $x=(x_1,\ldots,x_{20})^{\mathbf{T}}$, $-10\le x_i\le 10$.

## References

[ 1] Holland, J. H. 1975. *“Adaptation in Natural and Artificial System”*. The University of Michigan press, Michigan.

[ 2] Storn, R. and Price, K. 1997. Differential evolution. A simple and efficient heuristic scheme for global optimization over continuous spaces. *Journal of Global Optimization*, 11: 341-359.

[ 3] Kennedy, J. and Eberhart, R. 1995. Particle swarm optimization. *Proceedings of the 1995* IEEE *International Conference on Neural Networks*, Perth, Australia, 1942-1948.

[ 4] Fogel, L. J. Owens, A. J., and Walsh, M. J. 1966. *“Artificial intelligence through*

*simulated evolution"*. John Wiley, New York.

[ 5] Schwefel, H. P. 1995. *"Evolution and Optimum Seeking"*. John Wiley, New York.

[ 6] Vesterstrøm J. and Thomson R. 2004. A comparative study of differential evolution, particle swarm optimization, and evolutionary algorithms on numerical benchmark problems. Proceedings *of the 2004 Congress on Evolutionary Computation*, Portland, USA, 1980-1987.

[ 7] Pham, D. T. and Castellani, M. 2009. The bees algorithms: modeling foraging behaviour to solve continuous optimization problems. *Proceedings of the Institution of Mechanical Engineers, Part C : Journal of Mechanical Engineering Science*, 223: 2919-2938.

[ 8] Lin, W. Y., and Hsiao, K. M. 2004. Study on improvements of the five-point double-*toggle* mould clamping mechanism. *Proceedings of the Institution of Mechanical Engineers, Part C: Journal of Mechanical Engineering Science,* 218: 761–774.

[ 9] Lin, W. Y., Shen, C. L., and Hsiao, K. M. 2006. A case study of the five-point double-*toggle* mould clamping mechanism. *Proceedings of the Institution of Mechanical Engineers, Part C: Journal of Mechanical Engineering Science,* 220: 527-535.

[10] Lin, W. Y. and Wang, S. S. 2010. Dimensional synthesis of the five-point double-toggle mould clamping mechanism using a genetic algorithm-differential evolution hybrid algorithm, *Proceedings of the Institution of Mechanical Engineers, Part C: Journal of Mechanical Engineering Science,* 224:1305-1313.

[11] Andre, J., Siarry, P., and Dognon, T. 2001. An improvement of the standard genetic algorithm *fighting* premature convergence in continuous optimization. *Advances in Engineering Software,* 32: 49-60.

[12] Hrstka, O. and Kučerová, A. 2004. Improvement of real coded genetic algorithms based on differential operators preventing premature convergence. *Advances in Engineering Software*, 35: 237-246.

[13] Deb K. 2001. "*Multi-Objective Optimization using Evolutionary Algorithms"*. John Wiley & Sons, West Sussex.

[14] Goldberg, D. E. 1989. *"Genetic Algorithms in Search, Optimization and Machine Learning"*. Addison-Wesley, Massachusetts.

[15] Johannaber, F. 1994. *"Injection Molding Machines"*. Hanser, New York.

[16] Rönkkönen J. http://www.it.lut.fi/ ip /evo /functions/.

[17] Hedar A. R. http://www-optima.amp.i. Kyoto - u.ac.jp /memer /student/hedar/ Hedar_files/ TestGO_files/ Page364.htm.