An Adaptive Differential Evolution Algorithm to Solve Constrained Optimization Problems in Engineering Design

Youyun AO¹, Hongqin CHI²

¹School of Computer and Information, Anqing Teachers College, Anqing, China
²Department of Computer, Shanghai Normal University, Shanghai, China
Email: youyun ao@gmail.com, chihq@shnu.edu.cn
Received July 28, 2009; revised August 23, 2009; accepted August 28, 2009

Abstract

Differential evolution (DE) algorithm has been shown to be a simple and efficient evolutionary algorithm for global optimization over continuous spaces, and has been widely used in both benchmark test functions and real-world applications. This paper introduces a novel mutation operator, without using the scaling factor $F$, a conventional control parameter, and this mutation can generate multiple trial vectors by incorporating different weighted values at each generation, which can make the best of the selected multiple parents to improve the probability of generating a better offspring. In addition, in order to enhance the capacity of adaptation, a new and adaptive control parameter, i.e. the crossover rate $CR$, is presented and when one variable is beyond its boundary, a repair rule is also applied in this paper. The proposed algorithm ADE is validated on several constrained engineering design optimization problems reported in the specialized literature. Compared with respect to algorithms representative of the state-of-the-art in the area, the experimental results show that ADE can obtain good solutions on a test set of constrained optimization problems in engineering design.

Keywords: Differential Evolution, Constrained Optimization, Engineering Design, Evolutionary Algorithm, Constraint Handling

1. Introduction

Many real-world optimization problems involve multiple constraints which the optimal solution must satisfy. Usually, these problems are also called constrained optimization problems or nonlinear programming problems. Engineering design optimization problems are constrained optimization problems in engineering design. Like a constrained optimization problem, an engineering design optimization problem can be generally defined as follows [1–4]:

Minimize $f(\vec{x})$, $\vec{x} = [x_1, x_2, ..., x_n] \in \mathbb{R}^n$

Subject to $g_j(\vec{x}) \leq 0$, $j = 1, 2, ..., q$
$h_j(\vec{x}) = 0$, $j = q + 1, q + 2, ..., m$

where $L_i \leq x_i \leq U_i, i = 1, 2, ..., D$

Here, $n$ is the number of the decision or parameter variables (that is, $\vec{x}$ is a vector of size $D$), the $i$th variable $x_i$ varies in the range $[L_i, U_i]$. The function $f(\vec{x})$ is the objective function, $g_j(\vec{x})$ is the $j$th inequality constraint and $h_j(\vec{x})$ is the $j$th equality constraint. The decision or search space $S$ is written as $S = \prod_{i=1}^{n} [L_i, U_i]$ , the feasible space expressed as $F = \{\vec{x} \in S | g_j(\vec{x}) \leq 0, j = 1, 2, ..., q; h_j(\vec{x}) = 0, j = q + 1, q + 2, ..., m\}$ is one subset of the decision space $S$ (obviously, $F \subseteq S$) which satisfies the equality and inequality constraints.

Population-based evolutionary algorithm, mainly due to its ease to implement and use, and its less susceptibility to the characteristics of the function to be optimized, has been very popular and successfully applied to constrained optimization problems [5]. And many successful applications of evolutionary algorithms to solve engineering design optimization problems in the specialized literature have been reported. Ray and Liew [6] used a swarm-like based approach to solve engineering optimization problems. He et al. [7] proposed an improved particle swarm optimization to solve mechanical design...
optimization problems. Zhang et al. [8] proposed a differential evolution with dynamic stochastic selection to constrained optimization problems and constrained engineering design optimization problems. Akhtar et al. [9] proposed a socio-behavioural simulation model for engineering design optimization. He and Wang [10] proposed an effective co-evolutionary particle swarm optimization for constrained engineering design problems. Wang and Yin [11] proposed a ranking selection-based particle swarm optimizer for engineering optimization design problems. Differential evolution (DE) [12,13], a relatively new evolutionary technique, has been demonstrated to be simple and powerful and has been widely applied to both benchmark test functions and real-world applications [14]. This paper introduces an adaptive differential evolution (ADE) algorithm to solve engineering design optimization problems efficiently.

The remainder of this paper is organized as follows. Section 2 briefly introduces the basic idea of DE. Section 3 describes in detail the proposed algorithm ADE. Section 4 presents the experimental setup adopted and provides an analysis of the results obtained from our empirical study. Finally, our conclusions and some possible paths for future research are provided in Section 5.

2. The Basic DE Algorithm

Let’s suppose that \( \vec{x}_t = [x_{t,1}, x_{t,2}, \ldots, x_{t,D}] \) are solutions at generation \( t \), \( P^t = \{\vec{x}_t^1, \vec{x}_t^2, \ldots, \vec{x}_t^N\} \) is the population, where \( D \) denotes the dimension of solution space, \( N \) is the population size. In DE, the child population \( P^{t+1} \) is generated through the following operators [12,15]:

1) Mutation Operator: For each \( \vec{x}_i^t \) in parent population, the mutant vector \( \vec{u}_i^{t+1} \) is generated according to the following equation:

\[
\vec{u}_i^{t+1} = \vec{x}_i^t + F \times (\vec{x}_{i}^r - \vec{x}_{i}^j) \tag{2}
\]

where \( r_1, r_2, r_3 \in \{1,2,\ldots,N\} \setminus i \) are randomly chosen and mutually different, the scaling factor \( F \) controls amplification of the differential variation \( (\vec{x}_{r_2}^t - \vec{x}_{r_3}^t) \).

2) Crossover Operator: For each individual \( \vec{x}_i^t \), a trial vector \( \vec{u}_{i,j}^{t+1} \) is generated by the following equation:

\[
\vec{u}_{i,j}^{t+1} = \begin{cases} 
\vec{v}_{i,j}^{t+1}, & \text{if } (\text{rand} \leq CR \| j = \text{rand}[1,D]) \\
\vec{x}_{i,j}^t, & \text{otherwise}
\end{cases} \tag{3}
\]

where \( \text{rand} \) is a uniform random number distributed between 0 and 1, \( \text{rand}[1,D] \) is a randomly selected index from the set \( \{1,2,\ldots,D\} \), the crossover rate \( CR \in [0,1] \) controls the diversity of the population.

3) Selection Operator: The child individual \( \vec{x}_{i}^{t+1} \) is selected from each pair of \( \vec{x}_i^t \) and \( \vec{u}_{i}^{t+1} \) by using greedy selection criterion:

\[
\vec{x}_{i}^{t+1} = \begin{cases} 
\vec{u}_{i}^{t+1}, & \text{if } (f(\vec{u}_{i}^{t+1}) < f(\vec{x}_{i}^t)) \\
\vec{x}_{i}^t, & \text{otherwise}
\end{cases} \tag{4}
\]

where the function \( f \) is the objective function and the condition \( f(\vec{u}_{i}^{t+1}) < f(\vec{x}_{i}^t) \) means the individual \( \vec{u}_{i}^{t+1} \) is better than \( \vec{x}_{i}^t \).

Therefore, the conventional DE algorithm based on scheme DE/rand/1/bin is described in Figure 1 [15].

3. The Proposed Algorithm ADE

3.1. Generating Initial Population Using Orthogonal Design Method

Usually, the initial population \( P^0 = \{\vec{x}_0^{1}, \vec{x}_0^{2}, \ldots, \vec{x}_0^{N}\} \) of evolutionary algorithms is randomly generated as follows:

\[
\forall i \leq N, \forall j \leq D : \vec{x}_{i,j}^0 = L_j + r_j \times (U_j - L_j) \tag{5}
\]

where \( N \) is the population size, \( D \) is the number of variables, \( r_j \) is a random number between 0 and 1, the \( j \)th variable of \( \vec{x}_i^0 \) is written as \( x_{i,j}^0 \), which is initialized in the range \([L_j, U_j]\). In order to improve the search efficiency, this paper employs orthogonal design method to generate the initial population, which can make some points closer to the global optimal point and improve the diversity of solutions. The orthogonal design method is described as follows [16]:

For any given individual \( \vec{x} = [x_1, x_2, \ldots, x_D] \), the \( i \)th

<table>
<thead>
<tr>
<th>Step</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Generate initial population ( P^0 = {\vec{x}_0^{1}, \vec{x}_0^{2}, \ldots, \vec{x}_0^{N}} )</td>
</tr>
<tr>
<td>2</td>
<td>Let ( t = 0 )</td>
</tr>
<tr>
<td>3</td>
<td>repeat</td>
</tr>
<tr>
<td>4</td>
<td>for each individual ( \vec{x}_i^t ) in the population ( P^t ) do</td>
</tr>
<tr>
<td>5</td>
<td>Generate three random integers ( r_1, r_2 ) and ( r_3 ) \in {1,2,\ldots,N} \setminus i</td>
</tr>
<tr>
<td>6</td>
<td>Generate a random integer ( j_{\text{rand}} \in [1,2,\ldots,D] )</td>
</tr>
<tr>
<td>8</td>
<td>for each parameter ( j ) do</td>
</tr>
</tbody>
</table>
| 9 | \( u_{i,j}^{t+1} = \begin{cases} 
\vec{x}_{i,j}^t + F \times (\vec{x}_{r_1,j}^t - \vec{x}_{r_2,j}^t), & \text{if } (\text{rand} \leq CR \| j = \text{rand}[1,D]) \\
\vec{x}_{i,j}^t, & \text{otherwise}
\end{cases} \) |
| 11 | Replace \( \vec{x}_i^t \) with the child \( \vec{u}_i^{t+1} \) in the population \( P^{t+1} \), |
| 12 | if \( \vec{u}_i^{t+1} \) is better, otherwise \( \vec{x}_i^t \) is retained |
| 13 | end for |
| 14 | \( t = t + 1 \) |
| 15 | until the termination condition is achieved |

Figure 1. Pseudocode of differential evolution based on scheme DE/rand/1/bin.
decision variable \( x_i \) varies in the range \( [L_i, U_i] \). Here, each \( x_i \) is regarded as one factor of orthogonal design. Suppose that each factor holds \( Q \) levels, namely, quantize the domain \( [L_i, U_i] \) into \( Q \) levels \( a_1, a_2, \ldots, a_Q \). The \( j \)th level of the \( i \)th factor is written as \( a_{i,j} \), which is defined as follows:

\[
a_{i,j} = \begin{cases} L_i, & j = 1 \\ L_i + (j-1)\frac{U_i-L_i}{Q-1}, & 2 \leq j \leq Q-1 \\ U_i, & j = Q 
\end{cases}
\]

(6)

And then, we create the orthogonal array \( M = (b_{i,j})_{N \times D} \) with \( D \) factors and \( Q \) levels, where \( N \) is the number of level combinations. The procedure of constructing one orthogonal array \( M = (b_{i,j})_{N \times D} \) is described in Figure 2.

Therefore, the initial population \( P^0 = (x_{i,0})_{N \times D} \) is generated by using the orthogonal array \( M = (b_{i,j})_{N \times D} \), where the \( j \)th variable of individual \( x_i^0 \) is \( x_{i,j}^0 = a_{j,b_{i,j}} \).

### 3.2. Multi-Parent Mutation Scheme

According to the different variants of mutation, there are several different DE schemes often used, which are formulated as follows [12]:

- **DE/rand/1/bin**:
  \[
  \tilde{v}_{i}^{t+1} = x_i^t + F \times (x_{r_2}^t - x_{r_1}^t) 
  \]
  (7)

- **DE/rand/2/bin**:
  \[
  \tilde{v}_{i}^{t+1} = x_i^t + F \times (x_{r_3}^t - x_{r_1}^t) 
  \]
  (8)

- **DE/current to best/2/bin**:
  \[
  \tilde{v}_{i}^{t+1} = x_i^t + F \times (x_{r_2}^t - x_{b_{t+1}}^t) + F \times (x_{r_1}^t - x_{b_{t+1}}^t) 
  \]
  (9)

- **DE/best/1/bin**:
  \[
  \tilde{v}_{i}^{t+1} = x_i^t + F \times (x_{b_{t+1}}^t - x_{r_1}^t) + F \times (x_{r_1}^t - x_{r_2}^t) 
  \]
  (10)

- **DE/rand/2/bin**:
  \[
  \tilde{v}_{i}^{t+1} = x_i^t + F \times (x_{r_2}^t - x_{r_1}^t) + F \times (x_{r_1}^t - x_{r_3}^t) 
  \]
  (11)

where \( \tilde{x}_{best} \) is the best individual of the current population. Usually, based on both the control parameter \( F \) and the selected multiple parents, using these DE schemes can only generate a vector after a single mutation. Tsutsui et al. [17] proposed a multi-parent recombination with simplex crossover in real coded genetic algorithms to utilize the selected multiple parents and improve the diversity of offspring. Inspired by multi-parent recombination with simplex crossover, this paper proposes a novel multi-parent mutation in differential evolution. The multi-parent mutation is described in the following.

For each individual \( x_i^t \) from the population \( P' \) with population size \( N \), \( i = 1,2,\ldots, N \) a perturbed vector \( \tilde{v}_{i}^{t+1} \) is generated according to the following formula:

\[
\tilde{v}_{i}^{t+1} = x_i^t + \sum_{k=1}^{K} w_k \times (x_{r_k}^t - x_{i,k}^t) 
\]

(12)

where \( r_1, r_2, \ldots, r_K \in \{1,2,\ldots,N\} \setminus i, K \) randomly chosen integers are mutually different, and \( x_{i,k}^t = \tilde{x}_{i,k}^t \). The weighted value \( w_k \) is defined as follows:

\[
\tilde{\xi} = \text{randn}(1, K), \tilde{w} = \tilde{\xi} / \text{sum}(\tilde{\xi}) 
\]

(13)

where \( \text{randn}(1, K) \) is a 1-by-\( K \) matrix with normally distributed random numbers, \( \text{sum}(\tilde{\xi}) \) is used for calculating the sum of all components of the vector \( \tilde{\xi} \), and \( \tilde{w} = [w_1, w_2, \ldots, w_K] \).

According to the varying \( \tilde{w} \), repeat Formulas (13) and (12) for \( K \) times, \( K \) new vectors \( \tilde{v}_{i}^{t+1}\{1\}, \tilde{v}_{i}^{t+1}\{2\}, \ldots, \tilde{v}_{i}^{t+1}\{K\} \) are generated from these \( K \) selected parents. And then \( K \) vectors \( \tilde{x}_{i}^{t+1}\{1\}, \tilde{x}_{i}^{t+1}\{2\}, \ldots, \tilde{x}_{i}^{t+1}\{K\} \) are created by crossover, repair and constraint handling described in Subsections 3.3-3.5 respectively. Finally, an offspring individual \( \tilde{x}_{i}^{t+1} \) of the \( (t+1) \)th generation population \( P^{t+1} \) is obtained by selecting the best individual from these \( K \) offspring and their common parent \( x_i^t \).

### 3.3. Adaptive Crossover Rate CR

In conventional DE, the crossover rate \( CR \) is a constant value between 0 and 1. This paper proposes an adaptive crossover rate \( CR \), which is defined as follows:

\[
CR = CR_0 \times \exp(-a(t/t)^b) 
\]

(14)

where the initial crossover rate \( CR_0 \) is a constant value and usually is set to 0.8 or 0.85, \( t \) is the current genera-
tion number and \( T \) is the maximal generation number, \( b \) is a shape parameter determining the degree of dependency on the generation number, \( a \) and \( b \) are positive constants, usually \( a \) is set to 2, \( b \) is set to 2 or 3. At the early stage, \( \text{DE} \) uses a bigger crossover rate \( CR \) to preserve the diversity of solutions and prevent premature; at the later stage, \( \text{DE} \) employs a smaller crossover rate \( CR \) to enhance the local search and prevent the better solutions found from being destroyed.

### 3.4. Repair Method

After crossover, if one or more of the variables in the new vector \( \bar{u}_{i+1} \) are beyond their boundaries, the violated variable value \( \bar{u}_{i+1} \) is either reflected back from the violated boundary or set to the corresponding boundary value using the repair rule as follows [18,19]:

\[
\bar{u}_{i+1} = \begin{cases} 
\frac{L_j + u_{i+1}^{(j)} - b}{2}, & \text{if} \ (p \leq 1/3) \wedge (u_{i+1}^{(j)} < L_j) \\
L_j, & \text{if} \ (1/3 < p \leq 2/3) \wedge (u_{i+1}^{(j)} < L_j) \\
\frac{2L_j - u_{i+1}^{(j)} + b}{2}, & \text{if} \ (p > 2/3) \wedge (u_{i+1}^{(j)} < L_j) \\
\frac{U_j + u_{i+1}^{(j)} - b}{2}, & \text{if} \ (p \leq 1/3) \wedge (u_{i+1}^{(j)} > U_j) \\
U_j, & \text{if} \ (1/3 < p \leq 2/3) \wedge (u_{i+1}^{(j)} > U_j) \\
\frac{2U_j - u_{i+1}^{(j)} + b}{2}, & \text{if} \ (p > 2/3) \wedge (u_{i+1}^{(j)} > U_j)
\end{cases}
\]

where \( p \) is a probability and uniformly distributed random number in the range \([0,1]\).

### 3.5. Constraint Handling Technique of Feasibility-Based Rule

In evolutionary algorithms for solving constrained optimization problems, the most common method to handle constraints is to use penalty functions. In general, the constraint violation function of one individual \( \bar{x} \) is transformed by \( m \) equality and inequality constraints as follows [4]:

\[
G(\bar{x}) = \sum_{j=1}^{q} w_j \max(0, g_j(\bar{x}))^\beta + \sum_{j=q+1}^{m} w_j \max(0, h_j(\bar{x}) - \varepsilon)^\beta
\]

where the exponent \( \beta \) is usually set to 1 or 2, \( \varepsilon \) is a tolerance allowed (a very small value) for the equality constraints and the coefficient \( w_j \) is greater than zero.

If \( \bar{x} \) is a feasible solution, \( G(\bar{x}) = 0 \), otherwise \( G(\bar{x}) > 0 \). The function value \( G(\bar{x}) \) shows that the degree of constraints violation of individual \( \bar{x} \). \( \beta \) is set to 2 and \( w_j \) is set to 1 in this study.

In this study, a simple and efficient constraint handling technique of feasibility-based rule is introduced, which is also a constraint handling technique without using parameters. When two solutions are compared at a time, the following criteria are always applied [1]:

1) If one solution is feasible, and the other is infeasible, the feasible solution is preferred;
2) If both solutions are feasible, the one with the better objective function value is preferred;
3) If both solutions are infeasible, the one with smaller constraint violation function value is preferred.

### 3.6. Algorithm Framework

The general framework of the proposed algorithm \( \text{ADE} \) is described in Figure 3.

### 4. Experimental Study

### 4.1. Constrained Optimization Problems in Engineering Design

In order to validate the proposed algorithm \( \text{ADE} \), we use six benchmark test problems, which are commonly used

```plaintext
1: Generate initial population \( P_0 = \{x_0^0, x_1^0, \ldots, x_N^0\} \) using
2: orthogonal design method, set \( CR_0 \) and let \( t = 0 \)
3: repeat
4: for each individual \( x_t^i \) in the population \( P_t \) do
5: Generate \( K \) random integers \( n_1, n_2, \ldots, n_K \)
6: \( e \in \{1, 2, \ldots, N\} \setminus i \), they are also mutually different
7: for each \( k \in \{1, 2, \ldots, K\} \) do
8: Apply multi-parent mutation to generate new
9: vector \( \bar{x}_t^{i+1} \{k\} \)
10: for each parameter \( j \) do
11: \( \bar{x}_t^{i+1} \{k\} = \begin{cases} 
\bar{x}_t^{i+1} \{k\}, & \text{if (rand} \leq CR \| j = \text{rand}[1], D) \\
\bar{x}_t^{i+1} \{k\}, & \text{otherwise}
\end{cases}
\)
12: If \( \bar{x}_t^{i+1} \{k\} \) is beyond its lower or upper boundaries,
13: repair rule is enforced
14: end for
15: end for
16: Find out the best one \( \bar{x}_t^{i+1} \) of the children
17: \( \{\bar{x}_t^{i+1}, \bar{x}_t^{i+1}(2), \ldots, \bar{x}_t^{i+1}(K)\} \) *apply the feasibility–based rule */
18: Replace \( x_t^i \) with \( \bar{x}_t^{i+1} \) in the population \( P_{t+1} \),
19: end for
20: if \( \bar{x}_t^{i+1} \) is better, otherwise \( x_t^i \) is retained
21: end for
22: \( t = t + 1 \)
23: until the termination condition is achieved
```

Figure 3. The general framework of the \( \text{ADE} \) algorithm.
in the specialized literature, and which are described in the following.

1) Three-bar truss design [8]:
Minimize $f(\bar{x}) = (2\sqrt{2}x_1 + x_2) \times l$
Subject to $g_1(\bar{x}) = \sqrt{2}x_1 + x_2 - P - \sigma \leq 0$,
$g_2(\bar{x}) = \sqrt{2}x_1^2 + 2x_1x_2 - P - \sigma \leq 0$,
$g_3(\bar{x}) = \frac{1}{x_1 + \sqrt{2}x_2} - \sigma \leq 0$,
where $0 \leq x_1 \leq 1$ and $0 \leq x_2 \leq 1$; $l = 100$ cm,
$P = 2$ Kn/cm$^2$, and $\sigma = 2$ Kn/cm$^2$.

2) Spring design [8]:
Minimize $f(\bar{x}) = (x_3 + 2)x_1x_2^2$
Subject to $g_1(\bar{x}) = 1 - \frac{x_1^3x_3}{71785x_2^2} \leq 0$,
$g_2(\bar{x}) = \frac{4x_1^2}{12566(x_1^2x_2^2 - x_2^4)} + \frac{1}{5108x_2^2} - 1 \leq 0$,
$g_3(\bar{x}) = 1 - \frac{140.45x_2}{x_1^2x_3} \leq 0$,
$g_4(\bar{x}) = \frac{x_1 + x_2}{1.5} - 1 \leq 0$,
where $0.25 \leq x_1 \leq 1.3$, $0.05 \leq x_2 \leq 0.20$, and $2 \leq x_3 \leq 15$.

3) Pressure vessel design [9,20]:
Minimize $f(\bar{x}) = 0.6224x_1x_2x_3 + 1.7781x_2x_3^2 + 3.166x_2^3x_4 + 19.84x_1^2x_3$
Subject to $g_1(\bar{x}) = -x_1 + 0.0193x_3 \leq 0$,
$g_2(\bar{x}) = -x_2 + 0.00954x_3 \leq 0$,
$g_3(\bar{x}) = -x_3x_4 - \frac{4}{3}x_3^3 + 1.296,000 \leq 0$,
$g_4(\bar{x}) = x_4 - 240 \leq 0$,
where $x_1 = 0.0625n_1$, $x_2 = 0.0625n_2$, $1 \leq n_1 \leq 99$, $1 \leq n_2 \leq 99$, $10 \leq x_3 \leq 200$, $10 \leq x_4 \leq 200$.

4) Welded beam design [9]:
Minimize $f(\bar{x}) = 1.1047lx_1x_2 + 0.0481lx_1x_3x_4(14.0 + x_2)$
Subject to $g_1(\bar{x}) = (\tau(x) - \tau_{\max}) \leq 0$,
$g_2(\bar{x}) = (\sigma(x) - \sigma_{\max}) \leq 0$,
$g_3(\bar{x}) = x_1 - x_4 \leq 0$,
$g_4(\bar{x}) = 0.1047lx_1^2 + 0.0481lx_1x_3x_4(14.0 + x_2) - 5.0 \leq 0$,
$g_5(\bar{x}) = 0.125 - x_1 \leq 0$,
$g_6(\bar{x}) = \delta(x) - \delta_{\max} \leq 0$,

\[ g_7(\bar{x}) = P - P_c(x) \leq 0 \]

The other parameters are defined as follows:
$\tau(\bar{x}) = \sqrt{(\tau'x_2)^2 + 2\tau'x_2 + (\tau')^2}$, \quad $\tau' = \frac{P}{\sqrt{2}x_1x_2}$, \quad $\tau'' = \frac{MR}{J}$,
$M = P(L + \frac{x_2}{2})$, \quad $R = \frac{x_1^2}{4} + \left(\frac{x_1 + x_2}{2}\right)^2$,
$J = 2\left[\frac{x_1x_2}{12} + \left(\frac{x_1 + x_2}{2}\right)^2\right]$, \quad $\sigma(\bar{x}) = \frac{6PL}{x_4x_3^2}$,
$\delta(\bar{x}) = \frac{4PL^3}{Ex_4x_3^2}$,
$P_c(\bar{x}) = \frac{4.013\sqrt{EGx_4^2/36}}{L^2} \left(1 - \frac{x_3}{2L} \frac{E}{4G}\right)$,
where $P = 6000$ lb, $L = 14$ in, $\delta_{\max} = 0.25$ in.,
$E = 30 \times 10^6$ psi, $G = 12 \times 10^6$ psi, $\tau_{\max} = 13,600$ psi,
$\sigma_{\max} = 30,000$ psi, $0.1 \leq x_1 \leq 2.0$, $0.1 \leq x_2 \leq 10.0$, $0.1 \leq x_3 \leq 10.0$, and $0.1 \leq x_4 \leq 2.0$.

5) Speed reducer design [8]:
Minimize $f(\bar{x}) = 0.7854x_1x_2^2(3.3333x_3^2 + 14.9334x_3 - 43.0934)$
$-1.508(x_2^3 + x_2^2) + 7.4777(x_3^2 + x_7^2) + 0.7854(x_4x_6^2 + x_5x_7^2)$
Subject to $g_1(\bar{x}) = \frac{27}{x_1x_2^2x_3} - 1 \leq 0$,
$g_2(\bar{x}) = \frac{397.5}{x_1^2x_2^3} - 1 \leq 0$,
$g_3(\bar{x}) = \frac{1.93x_4^3}{x_2x_3x_6} - 1 \leq 0$,
$g_4(\bar{x}) = \frac{1.93x_5^3}{x_2x_3x_4^2} - 1 \leq 0$,
$g_5(\bar{x}) = \frac{[(745x_2/(x_2x_3))^2 + 16.9 \times 10^6]^{1/2}}{110x_3^6} - 1 \leq 0$,
$g_6(\bar{x}) = \frac{[(745x_2/(x_2x_3))^2 + 157.5 \times 10^6]^{1/2}}{850x_7^3} - 1 \leq 0$,
$g_7(\bar{x}) = \frac{x_2x_3}{40} - 1 \leq 0$,
$g_8(\bar{x}) = \frac{5x_2}{x_1} - 1 \leq 0$,
$g_9(\bar{x}) = \frac{x_1}{12x_2} - 1 \leq 0$,
$g_{10}(\bar{x}) = 1.5x_4 + 1.9 - 1 \leq 0$,
$g_{11}(\bar{x}) = \frac{1.1lx_7 + 1.9}{x_5} - 1 \leq 0$.
where \( 2.6 \leq x_1 \leq 3.6, \) \( 0.7 \leq x_2 \leq 0.8, \)
\( 17 \leq x_3 \leq 28, \) \( 7.3 \leq x_4 \leq 8.3, \)
\( 7.3 \leq x_5 \leq 8.3, \)
\( 2.9 \leq x_6 \leq 3.9, \) \( 5.0 \leq x_7 \leq 5.5. \)

6) Himmelblau’s Nonlinear Optimization Problem
[21]:
This problem was proposed by Himmelblau and similar to problem g04 [22] of the benchmark except for the second coefficient of the first constraint. There are five design variables. The problem can be stated as follows:

Minimize \( f(\vec{x}) = 5.3578547x_1^2 + 0.8356891x_1x_5 + 37.293239x_1 - 40792.141 \)
Subject to \( g_1(\vec{x}) = 85.334407 + 0.0056858x_2x_5 + 0.00026x_1x_4 - 0.0022053x_3x_5, \)
\ \( -92 \leq 0 \)
\( g_2(\vec{x}) = -85.334407 - 0.0056858x_2x_5 - 0.00026x_1x_4 + 0.0022053x_3x_5 \leq 0, \)
\( g_3(\vec{x}) = 80.51249 + 0.0071317x_2x_5 + 0.0029955x_1x_2 + 0.0021813x_1^2, \)
\ \( -110 \leq 0 \)
\( g_4(\vec{x}) = -80.51249 - 0.0071317x_2x_5 - 0.0029955x_1x_2 - 0.0021813x_1^2, \)
\ \( +90 \leq 0 \)
\( g_5(\vec{x}) = 9.300961 + 0.0047026x_3x_5 + 0.0012547x_1x_3 + 0.0019085x_3x_4, \)
\ \( -25 \leq 0 \)
\( g_6(\vec{x}) = -9.300961 - 0.0047026x_3x_5 - 0.0012547x_1x_3 - 0.0019085x_3x_4, \)
\ \( +20 \leq 0 \)

where \( 78 \leq x_1 \leq 102, \) \( 33 \leq x_2 \leq 45, \) and \( 27 \leq x_i \leq 45 \) (\( i = 3,4,5 \)).

4.2. Convergence of ADE

In this section, Figures 4-9 depict the convergence graphs for 6 engineering optimization problems described above respectively. From Figures 4-6, we know that ADE and DE all can be quickly convergent. In the figures, FFES is the number of fitness function evaluations.

4.3. Comparing ADE with Respect to Some State-of-the-Art Algorithms

In this experimental study, the parameter values used in ADE are set as follows: the population size \( N = 50 \), the maximal generation number \( T = 300 \), the level number \( Q = \left \lfloor \sqrt{N} \right \rfloor \), the mutation parent number \( K = D+1 \), the
Figure 7. Convergence graph for welded beam design.

Figure 8. Convergence graph for speed reducer design.

Figure 9. Convergence graph for Himmelblau’s nonlinear optimization problem.

\[ \begin{align*}
    initial \; crossover \; rate \; \CR_0 &= 0.8, \; \text{the \; coefficient} \; a &= 2, \\
    \text{the \; shape \; parameter} \; b &= 3, \; \text{the \; exponent} \; \beta &= 2. \; \\
    \text{The \; number \; of \; fitness \; function \; evaluations} \; (FFES) \; \text{is \; equal} \; \text{to} \; N \times T \times K. \\
    \text{The \; achieved \; solution \; at \; the \; end \; of} \; N \times T \times K \; FFES \; \text{is \; used \; to \; measure \; the \; performance \; of} \; \text{ADE, ADE \; is \; independently \; run \; 30 \; times \; on \; each \; test} \; \text{problem \; above.} \\
    \text{The \; optimized \; objective \; function \; values} \; (\text{of} \; 30 \; \text{runs}) \; \text{arranged \; in \; ascending \; order} \; \text{and \; the} \; 15\text{th} \; \text{value \; in \; the \; list} \; \text{is \; called \; the \; median \; optimized \; function} \; \text{value.} \\
    \text{Experimental \; results \; are \; presented \; in \; Tables} \; 1-12. \\
    \text{And \; NA \; is \; the \; abbreviation \; for} \; \text{“Not \; Available”}. \\
    \text{For \; three-bar \; truss \; design \; problem, the \; experimental \; results} \; \text{are \; given \; in \; Tables} \; 1-2. \\
    \text{According \; to} \; \text{Table} \; 1, \; \text{ADE and} \; \text{DSS-MDE} \; [8] \; \text{can \; obtain \; the \; approximate \; best \; and \; median} \; \text{values, \; which \; are \; slightly \; better \; than \; those \; obtained \; by \; Ray and Liew} \; [6].
\end{align*} \]

Table 1. Comparison of statistical results for three-bar truss design over 30 runs.

<table>
<thead>
<tr>
<th>Algorithms</th>
<th>Best</th>
<th>Median</th>
<th>Mean</th>
<th>Worst</th>
<th>Std</th>
<th>FFES</th>
</tr>
</thead>
<tbody>
<tr>
<td>ADE</td>
<td>263.89584338</td>
<td>263.89584338</td>
<td>263.89584338</td>
<td>263.89584338</td>
<td>4.72e-014</td>
<td>45,000</td>
</tr>
<tr>
<td>DSS-MDE [8]</td>
<td>263.8958434</td>
<td>263.8958434</td>
<td>263.8958436</td>
<td>263.8958498</td>
<td>9.72e-07</td>
<td>15,000</td>
</tr>
<tr>
<td>Ray and Liew [6]</td>
<td>263.8958466</td>
<td>263.8989</td>
<td>263.9033</td>
<td>263.96975</td>
<td>1.26e-02</td>
<td>17,610</td>
</tr>
</tbody>
</table>

Table 2. Comparison of best solutions found for three-bar truss design.

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>( x_1 )</td>
<td>0.7886751376014</td>
<td>0.7886751359</td>
<td>0.7886210370</td>
<td>0.78976441</td>
<td>0.795</td>
</tr>
<tr>
<td>( x_2 )</td>
<td>0.4082482819599</td>
<td>0.4082482868</td>
<td>0.4084013340</td>
<td>0.40517605</td>
<td>0.395</td>
</tr>
<tr>
<td>( f(x) )</td>
<td>263.895843376</td>
<td>263.8958434</td>
<td>263.8958466</td>
<td>263.896710000</td>
<td>264.300</td>
</tr>
<tr>
<td>FFES</td>
<td>45,000</td>
<td>15,000</td>
<td>17,610</td>
<td>55,000</td>
<td>2712</td>
</tr>
</tbody>
</table>
Table 3. Comparison of statistical results for spring design over 30 runs.

<table>
<thead>
<tr>
<th>Algorithms</th>
<th>Best</th>
<th>Median</th>
<th>Mean</th>
<th>Worst</th>
<th>Std</th>
<th>FFES</th>
</tr>
</thead>
<tbody>
<tr>
<td>ADE</td>
<td>0.0126652328</td>
<td>0.0126652458</td>
<td>0.0129336018</td>
<td>0.02064372078</td>
<td>1.46e-03</td>
<td>60,000</td>
</tr>
<tr>
<td>SiC-PSO [20]</td>
<td>0.012665</td>
<td>NA</td>
<td>0.0131</td>
<td>NA</td>
<td>4.1e-04</td>
<td>24,000</td>
</tr>
<tr>
<td>FSA [25]</td>
<td>0.012665285</td>
<td>NA</td>
<td>0.012665299</td>
<td>0.012665338</td>
<td>2.2e-08</td>
<td>49,531</td>
</tr>
<tr>
<td>DSS-MDE [8]</td>
<td>0.012665233</td>
<td>0.012665304</td>
<td>0.012669366</td>
<td>0.012738262</td>
<td>1.25e-05</td>
<td>24,000</td>
</tr>
<tr>
<td>Ray and Liew [6]</td>
<td>0.01266924934</td>
<td>0.012922669</td>
<td>0.012922669</td>
<td>0.0167177272</td>
<td>5.92e-04</td>
<td>25,167</td>
</tr>
<tr>
<td>Coello [26]</td>
<td>0.01270478</td>
<td>0.01275576</td>
<td>0.01276920</td>
<td>0.01282208</td>
<td>NA</td>
<td>900,000</td>
</tr>
</tbody>
</table>

Table 4. Comparison of best solutions found for spring design.

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>$x_1$</td>
<td>0.35674653865</td>
<td>0.354190</td>
<td>0.3567177469</td>
<td>0.35800478345599</td>
<td>0.356750</td>
</tr>
<tr>
<td>$x_2$</td>
<td>0.05169025814</td>
<td>0.051583</td>
<td>0.0516890614</td>
<td>0.05174250340926</td>
<td>0.051690</td>
</tr>
<tr>
<td>$x_3$</td>
<td>11.28727756428</td>
<td>11.438675</td>
<td>11.2880653382</td>
<td>11.21390736278739</td>
<td>11.287126</td>
</tr>
<tr>
<td>$f(x)$</td>
<td>0.0126652328</td>
<td>0.012665</td>
<td>0.01265233</td>
<td>0.01266525814</td>
<td>0.012665</td>
</tr>
<tr>
<td>FFES</td>
<td>60,000</td>
<td>24,000</td>
<td>24,000</td>
<td>49,531</td>
<td>49,531</td>
</tr>
</tbody>
</table>

Table 5. Comparison of statistical results for pressure vessel design over 30 runs.

<table>
<thead>
<tr>
<th>Algorithms</th>
<th>Best</th>
<th>Median</th>
<th>Mean</th>
<th>Worst</th>
<th>Std</th>
<th>FFES</th>
</tr>
</thead>
<tbody>
<tr>
<td>ADE</td>
<td>5885.3327736</td>
<td>5885.3327785</td>
<td>5885.3349564</td>
<td>5885.376942</td>
<td>8.66e-03</td>
<td>75,000</td>
</tr>
<tr>
<td>SiC-PSO [20]</td>
<td>6509.714335</td>
<td>NA</td>
<td>6902.0498</td>
<td>NA</td>
<td>12.1725</td>
<td>24,000</td>
</tr>
<tr>
<td>Ray and Liew [6]</td>
<td>6171.00</td>
<td>NA</td>
<td>6335.05</td>
<td>NA</td>
<td>NA</td>
<td>20,000</td>
</tr>
<tr>
<td>He et al. [7]</td>
<td>6509.714</td>
<td>NA</td>
<td>6289.929</td>
<td>NA</td>
<td>3.1e+2</td>
<td>30,000</td>
</tr>
<tr>
<td>Montes et al. [3]</td>
<td>6059.702</td>
<td>6059.702</td>
<td>6059.702</td>
<td>6059.702</td>
<td>1.0e-12</td>
<td>24,000</td>
</tr>
</tbody>
</table>

and Liew [6] respectively. The mean and worst values obtained by ADE are the best among three algorithms, while the FFES (45,000) of ADE is also the highest. And we also find that these algorithms can find the near-optimal solutions. From Table 2, we can see that ADE can find the best value when compared with respect to DSS-MDE [8], Ray and Liew [6], ECT [22] and Ray and Saini [23]. The best result obtained by ADE is

$$f(\bar{x}) = 263.8958433764684,$$

and constraints

$$[g_1(\bar{x}), g_2(\bar{x}), g_3(\bar{x})] = [0, -1.46410162480516, -0.53589837519484].$$

For spring design problem, the experimental results are given in Tables 3-4. According to Table 3, ADE, SiC-PSO [20], FSA [24], DSS-MDE [8] can find out the best value when compared with respect to Ray and Liew [6] and Coello [25]. The median value obtained by ADE is better than obtained by other methods, but the mean and worst values are worse, this is because that ADE can only find 29 near-optimal solutions in 30 runs and the other is an exception solution (i.e., the worst value is 0.02064372078). Table 4 presents the detail of each best value obtained by ADE, SiC-PSO [20], DSS-MDE [8], FSA [24] or He et al. [7] respectively. The best result obtained by ADE is:

$$f(\bar{x}) = 0.01266523278832,$$

and constraints

$$[g_1(\bar{x}), g_2(\bar{x}), g_3(\bar{x})] = [0.35671785021031, 0.05168906567225, 11.28895927857073].$$

Copyright © 2010 SciRes.
Y. Y. AO ET AL.

\[ g_1(\bar{x}), g_2(\bar{x}), g_3(\bar{x}), g_4(\bar{x}) \]
\[-2.204464049250313e-016, -4.440892098500626e-016, 4.0537858439796, -0.7277287277244966. \]

For pressure vessel design problem, the experimental results are given in Tables 5-6. According to Table 5, the best, median, mean, worst and standard deviation of values obtained by ADE are the best when compared with respect to Sic-PSO [20], Ray and Liew [6], He et al. [7], and Montes et al. [3], while the FFES (75,000) of ADE is also the highest. Table 6 presents the detail of each best value obtained by ADE, Sic-PSO [20], Ray and Liew [6], He et al. [7] or Montes et al. [3] respectively. The best result obtained by ADE is

\[ f(\bar{x})=5885.33273616458, \]
corresponding to

\[ \bar{x} = [x_1, x_2, x_3, x_4] \]
\[= [0.7781686414, 0.3846491626, 40.319618724, 200] \]
and constraints

\[ [g_1(\bar{x}), g_2(\bar{x}), g_3(\bar{x}), g_4(\bar{x})] \]
\[= [-1.110223024625157e-016, 0, 0, -40]. \]

For welded beam design problem, the experimental results are provided with Tables 7-8. According to Table 7, the best, median, mean, worst and standard deviation of values obtained by ADE are slightly worse than those obtained by DSS-MDE [8] and are better than those obtained by Ray and Liew [6], FSA [25] and Deb [1]. However, the FFES (75,000) of ADE is the highest. Table 8 presents the detail of each best value obtained by DSS-MDE [8], He et al. [7], FSA [25], Ray and Liew [6], and Akhtar et al. [9] respectively. The best result obtained by ADE is

\[ f(\bar{x})=2.38095658032252, \]
corresponding to

\[ \bar{x}=[x_1, x_2, x_3, x_4] \]
\[= [0.24436897580173, 6.2175197151746, 8.929147139048, 0.24436897580173] \]
and constraints

\[ [g_1(\bar{x}), g_2(\bar{x}), g_3(\bar{x}), g_4(\bar{x})] \]
\[= [-1.091393642127514e-011, -3.310560714453459e-010, -1.7878144-016, -3.02295760400, -0.23424083488769, -0.24436897580173]. \]

### Table 6. Comparison of best solutions found for pressure vessel design.

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>(x_1)</td>
<td>0.7781686414</td>
<td>0.812500</td>
<td>0.8125</td>
<td>0.8125</td>
<td>0.8125</td>
</tr>
<tr>
<td>(x_2)</td>
<td>0.3846491626</td>
<td>0.437500</td>
<td>0.4375</td>
<td>0.4375</td>
<td>0.4375</td>
</tr>
<tr>
<td>(x_3)</td>
<td>40.3196187244</td>
<td>42.098445</td>
<td>41.9768</td>
<td>42.098446</td>
<td>42.098446</td>
</tr>
<tr>
<td>(x_4)</td>
<td>200</td>
<td>176.636595</td>
<td>182.9768</td>
<td>176.636052</td>
<td>176.636047</td>
</tr>
<tr>
<td>(f(x))</td>
<td>5885.33277363</td>
<td>6059.714335</td>
<td>6171.0</td>
<td>6059.7143</td>
<td>6059.701660</td>
</tr>
<tr>
<td>FFES</td>
<td>75,000</td>
<td>24,000</td>
<td>20,000</td>
<td>30,000</td>
<td>24,000</td>
</tr>
</tbody>
</table>

### Table 7. Comparison of statistical results for welded beam design over 30 runs.

<table>
<thead>
<tr>
<th>Algorithms</th>
<th>Best</th>
<th>Median</th>
<th>Mean</th>
<th>Worst</th>
<th>Std</th>
<th>FFES</th>
</tr>
</thead>
<tbody>
<tr>
<td>ADE</td>
<td>2.380956580</td>
<td>2.380956580</td>
<td>2.380956580</td>
<td>2.380956570</td>
<td>2.35e-08</td>
<td>75,000</td>
</tr>
<tr>
<td>DSS-MDE [8]</td>
<td>2.38095658</td>
<td>2.38095658</td>
<td>2.38095658</td>
<td>2.38095658</td>
<td>3.19e-10</td>
<td>24,000</td>
</tr>
<tr>
<td>FSA [25]</td>
<td>2.381065</td>
<td>NA</td>
<td>2.404166</td>
<td>2.488967</td>
<td>NA</td>
<td>56.243</td>
</tr>
<tr>
<td>Deb [1]</td>
<td>2.38119</td>
<td>2.39289</td>
<td>NA</td>
<td>2.64583</td>
<td>NA</td>
<td>40,080</td>
</tr>
</tbody>
</table>
Table 8. Comparison of best solutions found for welded beam design.

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>$x_1$</td>
<td>0.2443689758</td>
<td>0.2443689758</td>
<td>0.244369</td>
<td>0.24435257</td>
<td>0.24438276</td>
<td>0.2407</td>
</tr>
<tr>
<td>$x_3$</td>
<td>8.2914713904</td>
<td>8.2914713904</td>
<td>8.291471</td>
<td>8.28930464</td>
<td>8.288576143</td>
<td>8.2399</td>
</tr>
<tr>
<td>$x_4$</td>
<td>0.2443689758</td>
<td>0.2443689758</td>
<td>0.244369</td>
<td>0.24435258</td>
<td>0.244566182</td>
<td>0.2497</td>
</tr>
<tr>
<td>$f(x)$</td>
<td>2.38095658032</td>
<td>2.38095658032</td>
<td>2.380957</td>
<td>2.381065</td>
<td>2.3854347</td>
<td>2.4426</td>
</tr>
<tr>
<td>FFES</td>
<td>75,000</td>
<td>24,000</td>
<td>30,000</td>
<td>56,243</td>
<td>33,095</td>
<td>19,259</td>
</tr>
</tbody>
</table>

Table 9. Comparison of statistical results for speed reducer design over 30 runs.

<table>
<thead>
<tr>
<th>Algorithms</th>
<th>Best</th>
<th>Median</th>
<th>Mean</th>
<th>Worst</th>
<th>Std</th>
<th>FFES</th>
</tr>
</thead>
<tbody>
<tr>
<td>ADE</td>
<td>2994.471066</td>
<td>2994.471066</td>
<td>2994.471066</td>
<td>2994.471066</td>
<td>1.85e-012</td>
<td>120,000</td>
</tr>
<tr>
<td>DSS-MDE [8]</td>
<td>2994.471066</td>
<td>2994.471066</td>
<td>2994.471066</td>
<td>2994.471066</td>
<td>3.58e-012</td>
<td>30,000</td>
</tr>
<tr>
<td>Montes et al. [27]</td>
<td>2996.367220</td>
<td>NA</td>
<td>2996.367220</td>
<td>NA</td>
<td>8.2e-03</td>
<td>24,000</td>
</tr>
<tr>
<td>Akhtar et al. [9]</td>
<td>3008.08</td>
<td>NA</td>
<td>3012.12</td>
<td>3028</td>
<td>NA</td>
<td>19,154</td>
</tr>
</tbody>
</table>

Table 10. Comparison of best solutions found for speed reducer design.

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>$x_1$</td>
<td>3.5</td>
<td>3.5</td>
<td>3.50000681</td>
<td>3.500010</td>
<td>3.506122</td>
</tr>
<tr>
<td>$x_2$</td>
<td>0.7</td>
<td>0.7</td>
<td>0.70000001</td>
<td>0.7</td>
<td>0.700006</td>
</tr>
<tr>
<td>$x_3$</td>
<td>17</td>
<td>17</td>
<td>17</td>
<td>17</td>
<td>17</td>
</tr>
<tr>
<td>$x_4$</td>
<td>7.3</td>
<td>7.3</td>
<td>7.32760205</td>
<td>7.300156</td>
<td>7.549126</td>
</tr>
<tr>
<td>$x_5$</td>
<td>7.715319911478</td>
<td>7.7153199115</td>
<td>7.71532175</td>
<td>7.800027</td>
<td>7.859330</td>
</tr>
<tr>
<td>$x_6$</td>
<td>3.350214666906</td>
<td>3.3502146661</td>
<td>3.35026702</td>
<td>3.350221</td>
<td>3.365576</td>
</tr>
<tr>
<td>$x_7$</td>
<td>5.286654464980</td>
<td>5.2866544650</td>
<td>5.28665450</td>
<td>5.286685</td>
<td>5.289773</td>
</tr>
<tr>
<td>$f(x)$</td>
<td>2994.471066</td>
<td>2994.471066</td>
<td>2994.744241</td>
<td>2996.356689</td>
<td>3008.08</td>
</tr>
<tr>
<td>FFES</td>
<td>120,000</td>
<td>30,000</td>
<td>54,456</td>
<td>24,000</td>
<td>18,154</td>
</tr>
</tbody>
</table>

Table 11. Comparison of statistical results for himmelblau’s nonlinear optimization problem.

<table>
<thead>
<tr>
<th>Algorithms</th>
<th>Best</th>
<th>Median</th>
<th>Mean</th>
<th>Worst</th>
<th>Std</th>
<th>FFES</th>
</tr>
</thead>
<tbody>
<tr>
<td>ADE</td>
<td>-31025.56024</td>
<td>-31025.56024</td>
<td>-31025.56024</td>
<td>-31025.56024</td>
<td>5.91e-010</td>
<td>90,000</td>
</tr>
<tr>
<td>COPSO [28]</td>
<td>-31025.56024</td>
<td>NA</td>
<td>-31025.56024</td>
<td>NA</td>
<td>0</td>
<td>200,000</td>
</tr>
<tr>
<td>HU-PSO [29]</td>
<td>-31025.56142</td>
<td>NA</td>
<td>-31025.56142</td>
<td>NA</td>
<td>0</td>
<td>200,000</td>
</tr>
</tbody>
</table>
Table 12. Comparison of best solutions found for himmelblau’s nonlinear optimization problem.

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>$x_1$</td>
<td>78</td>
<td>78.0</td>
<td>78.0495</td>
<td>78.0000</td>
<td></td>
</tr>
<tr>
<td>$x_2$</td>
<td>33.0</td>
<td>33.0</td>
<td>33.0070</td>
<td>33.0000</td>
<td></td>
</tr>
<tr>
<td>$x_3$</td>
<td>27.07099710517604</td>
<td>27.070997</td>
<td>27.0810</td>
<td>29.9950</td>
<td></td>
</tr>
<tr>
<td>$x_4$</td>
<td>45.0000000000000000</td>
<td>45.0</td>
<td>45.0000</td>
<td>45.0000</td>
<td></td>
</tr>
<tr>
<td>$x_5$</td>
<td>44.96924255010549</td>
<td>44.969242</td>
<td>44.9400</td>
<td>36.7760</td>
<td></td>
</tr>
<tr>
<td>$f(x)$</td>
<td>-31025.56024249794</td>
<td>-31025.56024249794</td>
<td>-31025.56142</td>
<td>-31020.859</td>
<td>-30665.609</td>
</tr>
<tr>
<td>FFES</td>
<td>90,000</td>
<td>200,000</td>
<td>200,000</td>
<td>NA</td>
<td>NA</td>
</tr>
</tbody>
</table>

For speed reducer design problem, the experimental results are given in Tables 9-10. According to Table 9, the best, median, mean, worst and standard derivation of values obtained by ADE and DSS-MDE [8] are superior to those obtained by Ray and Liew [6], Montes et al. [27] and Akhtar et al. [9] respectively, while the FFES (120,000) of ADE is the highest. Table 10 shows the detail of each best value obtained by ADE, DSS-MDE [8], Ray and Liew [6], Montes et al. [27] and Akhtar et al. [9] respectively. The best result obtained by ADE is

$$f(\bar{x}) = 2994.47106614682020,$$

corresponding to

$$\bar{x} = [x_1, x_2, x_3, x_4, x_5, x_6, x_7]$$

$$= [3.5, 0.7, 17, 7.3, 7.71531991147825, 3.35021466609645, 5.28665446980222]$$

and constraints

$$[g_1(\bar{x}), g_2(\bar{x}), g_3(\bar{x}), g_4(\bar{x}), g_5(\bar{x}), g_6(\bar{x}), g_7(\bar{x})]$$

$$= [-0.07391528039787, -0.19799852714195, -0.49917224810242, -0.90464390455607, -6.661338147750939e-016, 0, -0.70250000000000, -2.22044604925031e-016, -0.58333333333333, -0.05132575354183, -8.881784197001252e-016].$$

For Himmelblau’s nonlinear optimization problem, the best, median, mean, worst and standard derivation of values is shown in Tables 11-12, it is clearly seen that ADE, COPSO [28], and HU-PSO [29] all can find one near-optimal solution after a single run. Additionally, ADE only requires 90,000 FFES, which is superior to other several algorithms, such as COPSO [28] 200,000 FFES and HU-PSO [29] 200,000 FFES. The best result obtained by ADE is

$$f(\bar{x}) = -3.1025.56024249794,$$

corresponding to

$$\bar{x} = [x_1, x_2, x_3, x_4, x_5]$$

$$= [78, 33, 27.07099710517604, 45, 44.96924255010549]$$

and constraints

$$[g_1(\bar{x}), g_2(\bar{x}), g_3(\bar{x}), g_4(\bar{x}), g_5(\bar{x}), g_6(\bar{x})]$$

$$= [0, -92, -9.59476568762383, -10.4052341237617, -5, 0].$$

In sum, compared with respect to several state-of-the-art algorithms, ADE can perform better on six benchmark test problems. It is clearly shown that ADE is feasible and effective to solve constrained optimization problems in engineering design. The reason is that ADE uses multi-parent mutation to generate a better offspring, and applies self-adaptive control parameter and effective repair rule etc.

5. Conclusions and Future Work

This paper proposes an adaptive differential evolution (ADE) algorithm for constrained optimization in Engineering Design. Firstly, ADE employs the orthogonal design method to generate the initial population to improve the diversity of solutions. Secondly, a multi-parent mutation scheme is developed to improve the capacity of exploration and the convergence speed of ADE. Thirdly, in order to improve the adaptive capacity of crossover operator, a new approach to adjusting the crossover rate is presented. In addition, ADE introduces a new repair rule and a constraint handling technique of the feasible-based rule is also applied when comparing two solutions at a time. Finally, ADE is tested on six constrained engineering design optimization problems taken from the specialized literature. Compared with respect to several state-of-the-art algorithms, the experimental results show that ADE is highly competitive and can obtain good results in terms of a test set of constrained optimization problems.
problems in engineering design. However, there are still some things to do in the future. Firstly, we will further validate ADE in the case of higher dimensions. Secondly, we also will take some measures to improve the convergence speed during the evolutionary process. Additionally, testing some initial parameters of ADE is another future work.

6. References


