An Improved Adaptive Differential Evolution Algorithm with Population Adaptation

Ming Yang  
School of Computer Science,  
China University of Geosciences  
430074, Wuhan, China  
yangming0702@gmail.com

Zhihua Cai  
School of Computer Science,  
China University of Geosciences  
430074, Wuhan, China  
zhcai@cug.edu.cn

Changhe Li  
School of Computer Science,  
China University of Geosciences  
430074, Wuhan, China  
changhe lw@gmail.com

Jing Guan  
China Ship Development and Design Center  
430064, Wuhan, China  
gjing0414@yahoo.com.cn

ABSTRACT

In differential evolution (DE), there are many adaptive algorithms proposed for parameters adaptation. However, they mainly aim at tuning the amplification factor $F$ and crossover probability $CR$. When the population diversity is at a low level or the population becomes stagnant, the population is not able to improve any more. To enhance the performance of DE algorithms, in this paper, we propose a method of population adaptation. The proposed method can identify the moment when the population diversity is poor or the population stagnates by measuring the Euclidean distances between individuals of a population. When the moment is identified, the population will be regenerated to increase diversity or to eliminate the stagnation issue. The population adaptation is incorporated into the jDE algorithm and is tested on a set of 25 scalable CEC05 benchmark functions. The results show that the population adaptation can significantly improve the performance of the jDE algorithm. Even if the population size of jDE is small, the jDE algorithm with population adaptation also has a superior performance in comparisons with several other peer algorithms for high-dimensions function optimization.

Categories and Subject Descriptors

G.1.6 [NUMERICAL ANALYSIS]: Optimization—global optimization, unconstrained optimization

Keywords

Differential evolution, self-adaptation, population adaptation

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

Copyright 2013 ACM 978-1-4503-1963-8/13/07 ...$15.00.

1. INTRODUCTION

Differential evolution (DE), introduced by Price and Storn [17], is a simple yet powerful evolutionary algorithm (EA) for global optimization. Nowadays DE has become one of the most frequently used EAs for solving global optimization problems [15], mainly because it has good convergence properties and is principally easy to understand. Its effectiveness and efficiency have been successfully demonstrated in many real-life application fields.

DE creates new candidate solutions by combining the parent individual and different information between several other individuals of the same population. There are three control parameters in DE: amplification factor of the difference vector—$F$, crossover control parameter—$CR$, and population size—$NP$. The control parameters involved in DE are highly dependent on the problems to be solved [7, 10, 12], and the original DE algorithm keeps all the three control parameters fixed during the optimization process [10]. For a specific task, it may have to spend a huge amount of time to try and fine-tune the corresponding parameters. For adapting control parameters $F$ and $CR$, some adaptive or self-adaptive DE algorithms were developed to solve general problems more efficiently [5, 14, 20]. Liu and Lampinen proposed a fuzzy adaptive differential evolution (FADE) [11], using fuzzy logic controllers to adapt the control parameters $F$ and $CR$ for the mutation and crossover operations. Qin and Suganthan proposed a self-adaptive differential evolution (SaDE), where the choice of learning strategy and the two control parameters $F$ and $CR$ do not require pre-defining [16]. During evolution, suitable learning strategy and parameter settings are gradually self-adapted, according to the learning experience. In [19], the algorithm DE-SAP makes $F$, $CR$ and $NP$ evolve with individuals in the process of evolution to adapt mutation, crossover parameters, and population size by normal random numbers. Brest et al. in [2] proposed a new adaptive DE, called jDE, using a self-adapting mechanism on the control parameters $F_i$ and $CR$, associated with each individual. jDE was further extended by adapting two mutation strategies and the new algorithm was named jDE-2 [1]. Another new adaptive DE, called JADE, was proposed by Zhang et al. [23], in which the parameter adaptation was implemented by evolving the
mutation factors and crossover probabilities based on their historical records of success.

Although several adaptive DE algorithms have been proposed, they mainly focus on tuning the mutation factors $F$ and crossover probabilities $CR$. Over all the possible moves given by a population, some moves are beneficial in the search for the optimum while some others are ineffective and result in a waste of computational effort [6]. Therefore, increasing the population size will increase the diversity of possible movements, promoting the exploration of the search space. However, the probability to find the correct search direction decreases considerably. In [4], Brest et al. proposed a novel population size reduction method based on jDE [2], called dynNP-jDE. The population size reduction mechanism has a good performance in terms of robustness [14]. In the beginning of evolutionary process, the population size is large, and later it decreases by half at each predefined control point. By this way, the algorithm can spend much more time to improve the best individual using the saved resources. This population reduction method was also adopted in [3, 8, 22]. If the population size $NP$ is small, the algorithm will converge fast; but the risk of the DE premature convergence and stagnation becomes higher as the problem’s dimension increases, especially for problems with large decision spaces. To solve this issue with small size of population, this paper proposes a population adaptation method, which is able to increase the population diversity automatically.

2. ADAPTIVE DE WITH POPULATION ADAPTATION

In this section, we will introduce the jDE algorithm [2] with population adaptation, called PA-jDE. The PA-jDE algorithm is able to enhance the population diversity when the population diversity is poor and make the population continue to evolve when it has been in a stagnation.

2.1 The Population Adaptation

Suppose that $x_{i,G} = \{x_{i,1,G},x_{i,2,G},\ldots,x_{i,D,G}\}$, a $D$-dimensional target vector, is an individual at the $G$-th generation, $i=1,2,\ldots,NP$, where $NP$ is the population size. The sum Euclidean distances between individuals of a population is calculated as follows:

$$d_G = \sum_{i_1=1}^{NP} \sum_{i_2=1}^{NP} \sum_{j=1}^{D} (x_{i_1,j,G} - x_{i_2,j,G})^2.$$ 

When the population has converged at an optimum, the population diversity is poor. $d_G$ will not change any more in this case. In the case of stagnation, the algorithm may occasionally stop proceeding toward the global optimum even though the population has not converged to a local optimum or any other point [9]. When stagnation occurs, $d_G$ will also not change. $z_G$ is a flag to denote whether the population diversity is poor or the population is in a stagnation at the $G$-th generation:

$$z_G = \begin{cases} 1 & \text{if } \lambda_{j,G} \geq UN \\ 0 & \text{otherwise} \end{cases}$$

where

$$\lambda_{j,G} = \begin{cases} \lambda_{j,G-1} + 1 & \text{if } d_G = d_{G-1} \\ 0 & \text{otherwise} \end{cases}$$

and $UN$ is an integer whose value is set to $NP$ (the larger population size, the longer time it will take to enter into a stable stagnation state for the population, therefore we use $UN=NP$). $\lambda_{j,G}$ denotes the number of generations that the value of $d_G$ consecutively maintain unchanged ($\lambda_{j,0}=0$). If $d_G$ maintains unchanged over $UN$ consecutive generations, it indicates that the algorithm can not generate better trial vectors. In this case $z_G=1$, otherwise $z_G=0$ (see Eq. (2)). If $z_G$ equals 1, the algorithm needs to regenerate the population.

When $z_G$ equals 1, the new population $x_{i,G+1}, i=1,2,\ldots,NP$, is generated as follows:

$$x_{i,j,G+1} = \text{low}_{j,G} + \left(\text{up}_{j,G} - \text{low}_{j,G}\right) \cdot \text{randN}_{j,G}, j = 1, 2, \ldots, D$$

where

$$\text{low}_{j,G} = \min(m_{j,G}, x_{\text{low},j})$$

and

$$\text{up}_{j,G} = \max(m_{j,G}, x_{\text{up},j}).$$

$x_{\text{low},j}$ and $x_{\text{up},j}$ are the predefined lower and upper bounds for the $j$-th dimension, respectively. $m_{j,G}$ is the value of the best individual at the $j$-th dimension. For functions which are without search bounds, $\text{low}_{j,G}$ and $\text{up}_{j,G}$ are allowed to be beyond of the predefined bounds. In this case the algorithm can search outside of the predefined bounds. This is beneficial for the optimization of functions whose global optima are outside of the predefined bound. $\text{randN}_{j,G}$ is a random number with normal distribution of mean $\mu_{j,G}$ and variance $\sigma_{j,G}^2$ and then truncated to $[0,1]$. The values of $\mu_{j,G}$ and $\sigma_{j,G}$ are as follows:

$$\mu_{j,G} = \frac{m_{j,G} - \text{low}_{j,G}}{\text{up}_{j,G} - \text{low}_{j,G}}$$

$$\sigma_{j,G} = \sqrt{\frac{k}{\text{MaxFEs} + 1} \cdot \bar{\sigma}_{j,G}}$$

where

$$\bar{\sigma}_{j,G} = \max(\mu_{j,G}, 1 - \mu_{j,G}).$$

In Eq. (8), MaxFEs is a predefined maximal number of fitness evaluations and $k$ is the number of the current function evaluations. $\sigma_{j,G}$ decreases with the evolutionary progress by Eq. (8), so the diversity of the $j$-th dimension also decreases with the evolutionary progress. From Eq. (7) and (8), the new value of $x_{i,j,G+1}$ is generated nearby the best individual $m_{j,G}$ with a large probability but far way from $m_{j,G}$ with a small probability. This scheme is able to enable the algorithm to further exploit the local area of a revisited location if the area is not sufficiently searched. And the algorithm is also able to explore other promising areas.

Algorithm 1 illustrates the framework of the population adaptation (PA). The fitness of new individuals is evaluated and these evaluations are counted. Note that, in order to prevent the current best solution from being destroyed, the PA approach does not re-diversify the best solution found so far by the whole population.

2.2 Adaptations of $F$ and $CR$

The adaptations of $F$ and $CR$ employ the approaches of jDE [2]. The encoding of each individual in the population is extended with parameter values of $F$ and $CR$ (see Fig. 1). The better values of these (encoded) control parameters
numbers, and where individuals are beyond the search range. For the
this method is used for all algorithms to handle the situa-
defined lower and upper bounds, respectively. In this paper,
and \( \tau \), respectively.
end if
12: end if
end for
11: Evaluate the fitness for each individual \( x_{i,G} \), \( i = 1, 2, \ldots, NP \);
end for
10: end for
9: end if
8: end if
7: Regenerate \( x_{i,j,G+1} \) of the next generation using
Eq. (4);
6: if \( x_{i,j,G} \) is not the best individual then
5: for each individual \( x_{i,j,G} \), \( i = 1, 2, \ldots, NP \) do
4: for each dimension \( j = 1, 2, \ldots, D \) do
3: if rand equals 1 then
2: Compute \( d_{G} \);
1: Compute \( z_{2G} \) using Eq. (2);
end if
end for
Algorithm 1 Population Adaptation (PA)

lead to better individuals which, in turn, are more likely
to survive and produce offspring. The control parameters
\( F_{i,G+1} \) and \( CR_{i,G+1} \) are adapted as
\[
F_{i,G+1} = \begin{cases} 
F_{1,G} + rand_{1} \cdot F_{u} & \text{if } rand_{2} < \tau_{1} \\
F_{1,G} & \text{otherwise}
\end{cases}
\]
(10)
\[
CR_{i,G+1} = \begin{cases} 
rand_{3} & \text{if } rand_{4} < \tau_{2} \\
CR_{1,G} & \text{otherwise}
\end{cases}
\]
(11)
where \( rand_{j} \in [0, 1], j \in \{1, 2, 3, 4\} \), are uniform random
numbers, and \( \tau_{1} \) and \( \tau_{2} \) represent probabilities to adjust \( F \)
and \( CR \), respectively. \( F_{1} = 0.1 \), \( F_{u} = 0.9 \) and \( \tau_{1} = \tau_{2} = 0.1 \),
which are recommended for any test function [2]. So the
new \( F \) takes a value from \([0.1,1.0]\) and the new \( CR \) takes a value from \([0,1]\).

| \( x_{1,G} \) | \( x_{1,2G} \) | ... | \( x_{1,D,G} \) | \( F_{1G} \) | \( CR_{1G} \) |
| \( x_{2,1G} \) | \( x_{2,2G} \) | ... | \( x_{2,D,G} \) | \( F_{2G} \) | \( CR_{2G} \) |
| ... | ... | ... | ... | ... | ... |
| \( x_{NP,1G} \) | \( x_{NP,2G} \) | ... | \( x_{NP,D,G} \) | \( F_{NPG} \) | \( CR_{NPG} \) |

Figure 1: Self-adapting \( F \) and \( CR \): Individual encoding.

2.3 The Improved Adaptive DE Algorithm

The jDE algorithm with PA, called PA-jDE, applies the
PA approach after jDE operators at each iteration. The
pseudo-code of PA-jDE is presented in Algorithm 2. Com-
pared with the original jDE algorithm, only the step 26 is
added to perform Algorithm 1 in \( O(D \times NP) \). When \( u_{i,j} \)
exceeds the search range after the mutation, we map \( u_{i,j} \) to
be legal as follows:
\[
u_{i,j} = \begin{cases} 
(F_{\text{max}} \cdot x_{\text{up},j} - x_{\text{low},j} + u_{i,j}) / F_{\text{max}} & \text{if } u_{i,j} < x_{\text{low},j} \\
(F_{\text{max}} \cdot x_{\text{low},j} - x_{\text{up},j} + u_{i,j}) / F_{\text{max}} & \text{if } u_{i,j} > x_{\text{up},j}
\end{cases}
\]
(12)
where \( F_{\text{max}} \) is a max value of \( F \); \( x_{\text{low},j} \) and \( x_{\text{up},j} \) are pre-
defined lower and upper bounds, respectively. In this paper,
this method is used for all algorithms to handle the situation
where individuals are beyond the search range. For the
PA-jDE algorithm, \( F \in [0.1,1.0] \), so we set \( F_{\text{max}} = 1.0 \).

Algorithm 2 PA-jDE Algorithm

1: Generate uniform randomly the initial population \( P_{0} \);
2: Evaluate the fitness for each individual in \( P_{0} \);
3: \( G = 1 \);
4: while the stop criterion is not satisfied do
5: for each individual \( x_{i} \in P_{G} \) do
6: Select uniform randomly \( r_{1} \neq r_{2} \neq r_{3} \neq i \);
7: \( j_{\text{rand}}=\text{randint}(1,D) \);
8: for \( j = 1 \) to \( D \) do
9: if \( \text{rand}(0,1) < CR_{j} \) or \( j = j_{\text{rand}} \) then
10: \( u_{i,j} = x_{r_{2,j}} + F_{i} \cdot (x_{r_{2,j}} - x_{r_{3,j}}) \);
else
11: \( u_{i,j} = x_{i,j} \);
end if
12: end for
13: end if
14: end for
15: if \( u_{i,j} \notin [x_{\text{low},j}, x_{\text{up},j}] \) then
16: Use Eq. (12) to map \( u_{i,j} \) to be in the search range
\( [x_{\text{low},j}, x_{\text{up},j}] \);
17: end if
18: Evaluate the offspring \( u_{i} \);
19: end for
20: for each individual \( x_{i} \in P_{G} \) do
21: if \( u_{i} \) is not worse than \( x_{i} \) then
22: \( x_{i} = u_{i} \);
23: end if
24: end for
25: Adapt \( F_{i} \) and \( CR_{i} \) using Eq. (10) and Eq. (11),
respectively;
26: Implement the population adaptation using Algorithm
1;
27: \( G = G + 1 \);
28: end while

3. EXPERIMENTAL STUDY

In this section, PA-jDE is applied to minimize a set of
25 scalable CEC05 benchmark functions (see Table 1) in
dimensions \( D=30 \) and \( D=100 \). In Table 1, these CEC05
functions \( f_{1} \sim f_{25} \) include shifted functions, rotated functions
and rotated shifted functions. A more detailed description
and parameter settings of these CEC05 functions can be
found in [18]. \( f_{1} \sim f_{5} \) are unimodal functions, \( f_{6} \sim f_{12} \) are
basic multimodal functions, \( f_{13} \sim f_{14} \) are expanded multimodal
functions, and \( f_{15} \sim f_{25} \) are hybrid composition functions.
Because \( f_{7} \) and \( f_{25} \) are the functions without bounds, when \( u_{i,j} \)
exceeds the initial bounds, we do not map it by Eq. (12)
for all the DE algorithms and remain it unchanged for these
two functions.

PA-jDE will be compared with the classic DE/rand/1/bin
[17] to study the effect of the adaptive \( F \), \( CR \) and population
on DE. To study the effect of the adaptive population,
PA-jDE will be also compared with jDE [2]. The maximal
number of fitness evaluations (MaxFEs = 10000 \( \times D \)) is
used as the stop criteria for all the algorithms on each
function. The settings of all the algorithms are as follows:

1) DE/rand/1/bin: \( F=0.5 \) and \( CR=0.9 \) as used or rec-
ommended in [13, 17, 21]. \( NP=100 \) when \( D=30 \) and
\( NP=400 \) when \( D=100 \), which are used in [2, 16, 23].

2) jDE: \( F_{i}=0.1 \), \( F_{u}=0.9 \) and \( \tau_{1}=\tau_{2}=0.1 \) as recommended in
[2]. \( NP=100 \) when \( D=30 \) and \( NP=400 \) when \( D=100 \),
as used in [2]. The mutation strategy of rand/1/bin is
Table 1: CEC05 benchmark functions \[18\]. \(D\) denotes the dimensionality of the test problem, \(S\) denotes the ranges of the variables, and \(f_{min}\) is the function value of global optimum.

<table>
<thead>
<tr>
<th>(F)</th>
<th>Name</th>
<th>(S)</th>
<th>(f_{min})</th>
</tr>
</thead>
<tbody>
<tr>
<td>(f_1)</td>
<td>Shifted Sphere Function ((F_1))</td>
<td>([-100, 100]^D)</td>
<td>-450</td>
</tr>
<tr>
<td>(f_2)</td>
<td>Shifted Schwefel’s Problem 1.2 ((F_2))</td>
<td>([-100, 100]^D)</td>
<td>-450</td>
</tr>
<tr>
<td>(f_3)</td>
<td>Shifted Rotated High Conditioned Elliptic Function ((F_3))</td>
<td>([-100, 100]^D)</td>
<td>-450</td>
</tr>
<tr>
<td>(f_4)</td>
<td>Shifted Schwefel’s Problem 1.2 with Noise in Fitness ((F_4))</td>
<td>([-100, 100]^D)</td>
<td>-450</td>
</tr>
<tr>
<td>(f_5)</td>
<td>Schwefel’s Problem 2.6 with Global Optimum on Bounds ((F_5))</td>
<td>([-100, 100]^D)</td>
<td>-310</td>
</tr>
<tr>
<td>(f_6)</td>
<td>Shifted Rosenbrock’s Function ((F_6))</td>
<td>([-100, 100]^D)</td>
<td>390</td>
</tr>
<tr>
<td>(f_7)</td>
<td>Shifted Rotated Griewank’s Function without Bounds ((F_7))</td>
<td>([0, 600]^D)</td>
<td>-180</td>
</tr>
<tr>
<td>(f_8)</td>
<td>Shifted Rotated Ackley’s Function with Global Optimum on Bounds ((F_8))</td>
<td>([-32, 32]^D)</td>
<td>-140</td>
</tr>
<tr>
<td>(f_9)</td>
<td>Shifted Rastrigin’s Function ((F_9))</td>
<td>([-5, 5]^D)</td>
<td>-330</td>
</tr>
<tr>
<td>(f_{10})</td>
<td>Shifted Rotated Rastrigin’s Function ((F_{10}))</td>
<td>([-5, 5]^D)</td>
<td>-330</td>
</tr>
<tr>
<td>(f_{11})</td>
<td>Shifted Rotated Weierstrass Function ((F_{11}))</td>
<td>([-0.5, 0.5]^D)</td>
<td>90</td>
</tr>
<tr>
<td>(f_{12})</td>
<td>Schwefel’s Problem 2.13 ((F_{12}))</td>
<td>([-\pi, \pi]^D)</td>
<td>-460</td>
</tr>
<tr>
<td>(f_{13})</td>
<td>Expanded Extended Griewank’s plus Rosenbrock’s Function ((F_{13}))</td>
<td>([-5, 5]^D)</td>
<td>-130</td>
</tr>
<tr>
<td>(f_{14})</td>
<td>Shifted Rotated Expanded Scaffer’s F6 ((F_{14}))</td>
<td>([-100, 100]^D)</td>
<td>-300</td>
</tr>
<tr>
<td>(f_{15})</td>
<td>Hybrid Composition Function ((F_{15}))</td>
<td>([-5, 5]^D)</td>
<td>120</td>
</tr>
<tr>
<td>(f_{16})</td>
<td>Rotated Hybrid Composition Function ((F_{16}))</td>
<td>([-5, 5]^D)</td>
<td>120</td>
</tr>
<tr>
<td>(f_{17})</td>
<td>Rotated Hybrid Composition Function with Noise in Fitness ((F_{17}))</td>
<td>([-5, 5]^D)</td>
<td>120</td>
</tr>
<tr>
<td>(f_{18})</td>
<td>Rotated Hybrid Composition Function ((F_{18}))</td>
<td>([-5, 5]^D)</td>
<td>10</td>
</tr>
<tr>
<td>(f_{19})</td>
<td>Rotated Hybrid Composition Function with a Narrow Basin for the Global Optimum ((F_{19}))</td>
<td>([-5, 5]^D)</td>
<td>10</td>
</tr>
<tr>
<td>(f_{20})</td>
<td>Rotated Hybrid Composition Function with the Global Optimum on the Bounds ((F_{20}))</td>
<td>([-5, 5]^D)</td>
<td>10</td>
</tr>
<tr>
<td>(f_{21})</td>
<td>Rotated Hybrid Composition Function ((F_{21}))</td>
<td>([-5, 5]^D)</td>
<td>360</td>
</tr>
<tr>
<td>(f_{22})</td>
<td>Rotated Hybrid Composition Function with High Condition Number Matrix ((F_{22}))</td>
<td>([-5, 5]^D)</td>
<td>360</td>
</tr>
<tr>
<td>(f_{23})</td>
<td>Non-Continuous Rotated Hybrid Composition Function ((F_{23}))</td>
<td>([-5, 5]^D)</td>
<td>360</td>
</tr>
<tr>
<td>(f_{24})</td>
<td>Rotated Hybrid Composition Function ((F_{24}))</td>
<td>([-5, 5]^D)</td>
<td>260</td>
</tr>
<tr>
<td>(f_{25})</td>
<td>Rotated Hybrid Composition Function without Bounds ((F_{25}))</td>
<td>([2, 5]^D)</td>
<td>260</td>
</tr>
</tbody>
</table>

employed and the parameters use the same settings as DE/rand/1/bin.

3) PA-jDE: The parameters use the same settings as jDE, except that its population size is set to be \(NP=20\) for all the experiments.

Table 2 and Table 3 summarize the average error results of 30 independent runs for each algorithm on each function with \(D=30\) and \(D=100\), respectively. For each function, the best value of the results got by all the algorithms is shown in bold font. \(b/n/w\) summarizes the statistical results: \(b\), \(n\) and \(w\) denote the number of functions for which PA-jDE performs significantly better, no significantly different and significantly worse than an algorithm, respectively. For 30-dimensions problems, from Table 2, it can be seen that there are 12 functions for which DE/rand/1/bin performs best; while there are 9 functions for which PA-jDE performs best. Compared with DE/rand/1/bin, PA-jDE performs worse than DE/rand/1/bin. But PA-jDE performs better than the two jDE algorithms with \(NP=20\) and \(NP=100\). Especially, compared with jDE with \(NP=20\), although PA-jDE has the same initial populations and the same adaptation of \(F\) and \(CR\) as jDE, PA-jDE performs significantly better than jDE, because that the PA approach does help improve the performance of jDE. In Table 3, for the 100-dimensions problems, there are 19 functions for which PA-jDE can get the best results, when \(D=100\). And PA-jDE performs much significantly better than DE/rand/1/bin and jDE with \(NP=400\). This is because an enlargement in population size causes an increase in the set of potential ineffective moves. For these high-dimension functions, compared with jDE with \(NP=20\), PA-jDE performs significantly better. From the above results, we can see that although the population size of PA-jDE is small, its performance is good, and we can conclude that the population adaptation can improve the performance of jDE algorithm, especially for high-dimension problems.

We select two typical functions to show the convergence graphs in Fig. 2: the unimodal function \(f_1\) and the multimodal and hybrid composition function \(f_{15}\) with \(D=100\). In Fig. 2, the curves record the mean error of the best results over 30 independent runs. From Fig. 2a, it can be seen that PA-jDE and jDE with \(NP=20\) converge much faster than other algorithms for the unimodal function \(f_1\), because they have small population size. In Fig. 2b, for the multimodal and hybrid composition function \(f_{15}\), in the beginning of the evolutionary process, the convergence of PA-jDE and jDE with \(NP=20\) is faster than jDE and DE/rand/1/bin with \(NP=400\), but the convergence of PA-jDE and jDE with \(NP=20\) becomes slow as the evolutionary progresses. Compared with jDE with \(NP=20\), when the population is trapped in a local optimum, PA-jDE can jump out of the
Table 2: Average error values archived for 25 30-dimensions CEC05 benchmark functions over 30 independent runs: the mean best result and corresponding standard deviation value.

<table>
<thead>
<tr>
<th>Function</th>
<th>PA-jDE</th>
<th>jDE (NP=20)</th>
<th>jDE (NP=100)</th>
<th>DE/rand/1/bin</th>
</tr>
</thead>
<tbody>
<tr>
<td>$f_1$</td>
<td>0±0</td>
<td>0±0</td>
<td>0±0</td>
<td>0±0</td>
</tr>
<tr>
<td>$f_2$</td>
<td>4.02e-013±4.57e-013</td>
<td>4.02e-013±4.57e-013</td>
<td>6.80e±000±4.03e±000</td>
<td>1.90e±004±1.96e±004</td>
</tr>
<tr>
<td>$f_3$</td>
<td>0±0</td>
<td>0±0</td>
<td>0±0</td>
<td>0±0</td>
</tr>
<tr>
<td>$f_4$</td>
<td>2.89e-001±3.51e-001</td>
<td>8.99e-001±2.65e-002</td>
<td>1.13e±002±7.62e±001</td>
<td>4.10e±002±3.14e±002</td>
</tr>
<tr>
<td>$f_5$</td>
<td>6.42e-001±1.37e-000</td>
<td>6.42e-001±1.37e-000</td>
<td>6.68e+001±2.19e+001</td>
<td>3.84e+001±4.28e+001</td>
</tr>
<tr>
<td>$f_6$</td>
<td>3.92e+000±3.36e+000</td>
<td>3.92e+000±3.36e+000</td>
<td>8.67e+000±5.30e+001</td>
<td>3.47e+001±3.32e+001</td>
</tr>
<tr>
<td>$f_7$</td>
<td>8.29e+003±7.63e+003</td>
<td>8.29e+003±7.63e+003</td>
<td>3.64e+012±5.66e+012</td>
<td>0±0</td>
</tr>
<tr>
<td>$f_8$</td>
<td>2.08e+001±5.04e+002</td>
<td>2.09e+001±5.18e+002</td>
<td>2.10e+001±4.43e+002</td>
<td>2.09e+001±4.16e+002</td>
</tr>
<tr>
<td>$f_9$</td>
<td>4.84e+000±3.48e+000</td>
<td>4.48e+000±3.19e+000</td>
<td>1.13e+004±3.82e+004</td>
<td>1.27e+002±2.04e+001</td>
</tr>
<tr>
<td>$f_{10}$</td>
<td>4.99e+001±1.46e+001</td>
<td>5.62e+001±1.09e+001</td>
<td>1.37e+002±8.87e+000</td>
<td>1.83e+002±1.00e+001</td>
</tr>
<tr>
<td>$f_{11}$</td>
<td>1.25e+001±2.54e+000</td>
<td>2.02e+001±4.29e+000</td>
<td>3.36e+001±1.49e+000</td>
<td>3.91e+001±1.30e+000</td>
</tr>
<tr>
<td>$f_{12}$</td>
<td>3.65e+003±5.53e+003</td>
<td>3.82e+003±5.81e+003</td>
<td>3.96e+004±9.66e+003</td>
<td>4.96e+003±4.75e+003</td>
</tr>
<tr>
<td>$f_{13}$</td>
<td>3.29e+000±2.52e+000</td>
<td>3.55e+000±2.50e+000</td>
<td>2.07e+000±5.24e+001</td>
<td>1.26e+000±2.94e+001</td>
</tr>
<tr>
<td>$f_{14}$</td>
<td>1.25e+001±3.58e+001</td>
<td>1.30e+001±2.01e+001</td>
<td>1.32e+001±2.02e+001</td>
<td>1.35e+001±1.36e+001</td>
</tr>
<tr>
<td>$f_{15}$</td>
<td>2.25e+002±2.35e+002</td>
<td>7.08e+002±1.24e+002</td>
<td>1.00e+002±2.33e+001</td>
<td>1.20e+002±1.09e+002</td>
</tr>
<tr>
<td>$f_{16}$</td>
<td>3.13e+002±2.85e+002</td>
<td>5.76e+002±2.43e+002</td>
<td>1.00e+002±2.33e+001</td>
<td>1.00e+002±2.38e+012</td>
</tr>
<tr>
<td>$f_{17}$</td>
<td>1.50e+002±1.57e+002</td>
<td>6.95e+002±1.16e+002</td>
<td>4.34e+002±2.53e+002</td>
<td>1.12e+002±6.30e+000</td>
</tr>
<tr>
<td>$f_{18}$</td>
<td>6.01e+002±3.51e+002</td>
<td>8.52e+002±3.12e+002</td>
<td>3.00e+002±4.07e+013</td>
<td>3.00e+002±2.46e+012</td>
</tr>
<tr>
<td>$f_{19}$</td>
<td>5.60e+002±3.48e+002</td>
<td>8.30e+002±3.01e+002</td>
<td>3.00e+002±1.73e+013</td>
<td>3.00e+002±1.51e+012</td>
</tr>
<tr>
<td>$f_{20}$</td>
<td>1.01e+003±2.29e+001</td>
<td>1.02e+003±2.38e+001</td>
<td>3.00e+002±4.53e+010</td>
<td>3.00e+002±2.79e+011</td>
</tr>
<tr>
<td>$f_{21}$</td>
<td>1.18e+003±1.35e+001</td>
<td>1.18e+003±1.57e+001</td>
<td>1.15e+003±1.16e+001</td>
<td>1.14e+003±6.35e+000</td>
</tr>
<tr>
<td>$f_{22}$</td>
<td>8.70e+002±4.33e+002</td>
<td>1.16e+003±2.52e+001</td>
<td>1.18e+003±1.45e+001</td>
<td>9.18e+002±3.49e+002</td>
</tr>
<tr>
<td>$f_{23}$</td>
<td>1.22e+003±1.41e+001</td>
<td>1.22e+003±1.50e+001</td>
<td>1.19e+003±9.53e+000</td>
<td>1.19e+003±6.21e+000</td>
</tr>
<tr>
<td>$f_{24}$</td>
<td>1.10e+003±8.42e+000</td>
<td>1.10e+003±1.11e+001</td>
<td>1.11e+003±4.12e+000</td>
<td>1.12e+003±4.08e+000</td>
</tr>
<tr>
<td>$f_{25}$</td>
<td>1.08e+003±2.39e+002</td>
<td>1.16e+003±3.24e+001</td>
<td>1.24e+003±3.39e+001</td>
<td>1.05e+003±3.94e+002</td>
</tr>
</tbody>
</table>

b/n/w — 11/13/1 12/3/10 6/8/11

1 PA-jDE performs significantly better than the algorithm at a 0.05 level of significance by the paired samples Wilcoxon signed rank test.

2 PA-jDE performs significantly worse than the algorithm at a 0.05 level of significance by the paired samples Wilcoxon signed rank test.

Figure 2: Convergence graph for test functions $f_1$ and $f_{15}$ when $D=100$. The horizontal axis is the number of function evaluations, and the vertical axis is the mean error of best results over 30 independent runs.
Table 3: Average error values archived for 25 100-dimensions CEC05 benchmark functions over 30 independent runs: the mean best result and corresponding standard deviation value.

<table>
<thead>
<tr>
<th>$F$</th>
<th>PA-jDE</th>
<th>jDE ($NP=20$)</th>
<th>jDE ($NP=400$)</th>
<th>DE/rand/1/bin</th>
</tr>
</thead>
<tbody>
<tr>
<td>$f_1$</td>
<td>5.49e-014±1.04e-014</td>
<td>5.49e-014±1.04e-014</td>
<td>2.49e-005±5.55e-006†</td>
<td>5.77e+001±1.53e+001†</td>
</tr>
<tr>
<td>$f_2$</td>
<td>1.39e+001±2.30e+001</td>
<td>1.01e+001±1.56e+001</td>
<td>3.13e+005±2.88e+004†</td>
<td>4.31e+005±3.29e+004†</td>
</tr>
<tr>
<td>$f_3$</td>
<td>5.68e-014±0</td>
<td>5.68e-014±0</td>
<td>2.36e-002±5.47e-003†</td>
<td>2.48e-004±6.10e-003†</td>
</tr>
<tr>
<td>$f_4$</td>
<td>2.72e+004±5.74e+003</td>
<td>6.74e+004±1.51e+004†</td>
<td>3.99e+005±2.65e+004†</td>
<td>5.13e+005±4.41e+004†</td>
</tr>
<tr>
<td>$f_5$</td>
<td>2.93e+001±1.29e+001</td>
<td>2.93e+001±1.29e+001</td>
<td>6.23e+004±3.09e+003†</td>
<td>4.59e+005±4.33e+004†</td>
</tr>
<tr>
<td>$f_6$</td>
<td>8.25e+001±3.97e+000</td>
<td>8.25e+001±3.97e+000</td>
<td>9.20e+001±1.80e+001†</td>
<td>9.75e+001±3.61e+001†</td>
</tr>
<tr>
<td>$f_7$</td>
<td>8.29e-003±9.52e-003</td>
<td>8.29e-003±9.52e-003</td>
<td>5.23e+001±4.88e-002†</td>
<td>5.11e+000±1.08e+000†</td>
</tr>
<tr>
<td>$f_8$</td>
<td>2.12e+001±2.19e-002</td>
<td>2.13e+001±1.59e-002</td>
<td>2.13e+001±2.26e-002†</td>
<td>2.13e+001±2.08e-002†</td>
</tr>
<tr>
<td>$f_9$</td>
<td>9.31e+001±1.66e+001</td>
<td>6.32e+001±2.18e+001†</td>
<td>4.06e+002±2.32e+001†</td>
<td>9.83e+002±2.76e+001†</td>
</tr>
<tr>
<td>$f_{10}$</td>
<td>3.14e+002±4.90e+001</td>
<td>3.56e+002±5.26e+001†</td>
<td>8.61e+002±3.23e+001†</td>
<td>9.87e+002±3.17e+001†</td>
</tr>
<tr>
<td>$f_{11}$</td>
<td>8.22e+001±7.29e+000</td>
<td>1.34e+002±1.12e+001†</td>
<td>1.15e+002±2.48e+000†</td>
<td>1.61e+002±1.66e+000†</td>
</tr>
<tr>
<td>$f_{12}$</td>
<td>2.19e+004±1.55e+004</td>
<td>1.95e+004±1.42e+004</td>
<td>5.26e+006±2.60e+005†</td>
<td>1.78e+007±8.33e+005†</td>
</tr>
<tr>
<td>$f_{13}$</td>
<td>7.02e+001±1.75e+001</td>
<td>7.15e+001±1.75e+001</td>
<td>1.71e+001±3.36e+000†</td>
<td>6.16e+001±7.10e+000†</td>
</tr>
<tr>
<td>$f_{14}$</td>
<td>4.67e+001±4.05e+001</td>
<td>4.72e+001±2.41e+001†</td>
<td>4.75e+001±1.67e+001†</td>
<td>4.77e+001±1.48e+001†</td>
</tr>
<tr>
<td>$f_{15}$</td>
<td>8.05e+002±1.39e+002</td>
<td>9.33e+002±3.06e+001†</td>
<td>7.96e+002±1.42e+002</td>
<td>9.96e+002±3.54e+001†</td>
</tr>
<tr>
<td>$f_{16}$</td>
<td>7.49e+002±1.09e+002</td>
<td>8.68e+002±7.04e+001†</td>
<td>5.84e+002±6.87e+001†</td>
<td>7.24e+002±1.33e+001†</td>
</tr>
<tr>
<td>$f_{17}$</td>
<td>7.29e+002±3.27e+001</td>
<td>9.26e+002±4.38e+001†</td>
<td>1.03e+003±1.29e+001†</td>
<td>1.14e+003±9.29e+000†</td>
</tr>
<tr>
<td>$f_{18}$</td>
<td>1.18e+003±1.17e+001</td>
<td>1.18e+003±9.91e+000</td>
<td>1.26e+003±4.14e+000†</td>
<td>1.28e+003±4.42e+000†</td>
</tr>
<tr>
<td>$f_{19}$</td>
<td>1.15e+003±1.13e+001</td>
<td>1.16e+003±1.30e+001†</td>
<td>1.24e+003±4.01e+000†</td>
<td>1.26e+003±4.94e+000†</td>
</tr>
<tr>
<td>$f_{20}$</td>
<td>1.14e+003±1.51e+001</td>
<td>1.16e+003±1.58e+001†</td>
<td>1.17e+003±1.01e+001†</td>
<td>1.22e+003±1.11e+001†</td>
</tr>
<tr>
<td>$f_{21}$</td>
<td>1.20e+003±1.29e+001</td>
<td>1.20e+003±1.16e+001†</td>
<td>1.22e+003±8.00e+000†</td>
<td>1.27e+003±5.11e+000†</td>
</tr>
<tr>
<td>$f_{22}$</td>
<td>1.16e+003±9.61e+000</td>
<td>1.17e+003±1.29e+001†</td>
<td>1.17e+003±5.73e+000†</td>
<td>1.20e+003±5.36e+000†</td>
</tr>
<tr>
<td>$f_{23}$</td>
<td>1.22e+003±1.42e+001</td>
<td>1.22e+003±1.43e+001†</td>
<td>1.23e+003±7.38e+000†</td>
<td>1.28e+003±6.98e+000†</td>
</tr>
<tr>
<td>$f_{24}$</td>
<td>1.13e+003±1.20e+001</td>
<td>1.14e+003±1.77e+001†</td>
<td>1.15e+003±2.65e+000†</td>
<td>1.17e+003±3.43e+000†</td>
</tr>
<tr>
<td>$f_{25}$</td>
<td>1.25e+003±1.59e+001</td>
<td>1.28e+003±2.11e+001†</td>
<td>1.38e+003±7.13e+000†</td>
<td>1.39e+003±5.46e+000†</td>
</tr>
</tbody>
</table>

$\dagger$ PA-jDE performs significantly better than the algorithm at a 0.05 level of significance by the paired samples Wilcoxon signed rank test.

† PA-jDE performs significantly worse than the algorithm at a 0.05 level of significance by the paired samples Wilcoxon signed rank test.

Figure 3: The change of $d_G$ and the error result of the best individual with the evolutionary progress in a run of the PA-jDE algorithm for the test function $f_{15}$ in 100-dimensions.
local optimum and continue to evolve because of the population adaptation.

To study the working mechanism of population adaptation, we apply PA-jDE to minimize the multimodal and hybrid composition function $f_{15}$ in 100-dimensions. Fig. 3 shows the change of $d_C$ (see Eq. (1)) and the error result of the best individual with the evolutionary progress in a run. When the population diversity is poor, $d_C$ is very small (see Fig. 3a) and the algorithm stops evolving (see Fig. 3b). Then PA is executed and the population diversity is enhanced. $d_C$ increase greatly (see Fig. 3a). The algorithm continues to evolve, and the error result decreases (see Fig. 3a). From the figure, we can see that when the population diversity is poor, PA can enhance it. The new population generated by PA is beneficial to make the best individual get better results.

4. CONCLUSIONS

From the possible moves given by a population, some moves are beneficial in the search for the optimum while some others are ineffective and result in a waste of computational effort. Therefore, increasing the population size will increase the diversity of possible movements, promoting the exploration of the search space. However, the probability to find the correct search direction decreases considerably. If the population size is small, although the convergence of the algorithm is fast, the population would be very likely to get trapped in a local optimum. To enhance the performance of DE with small population size, this paper has studied the population adaptation on DE and proposed a method of population adaptation, called PA.

Through measuring the Euclidean distances between individuals of a population, PA can identify the moment when the population diversity is poor or the population is in a stagnation. When the moment is identified, PA can regenerate the population by normal random numbers. The new values are generated nearby the best individual with a large probability but far way from it with a small probability. The experimental results show that PA can enhance the population diversity when the population diversity is poor and improve the performance of jDE algorithm. PA can improve the performance of the algorithms whose population sizes are small. Even if its population size is small, the jDE algorithm with population adaptation also has a superior performance than several other peer algorithms for high-dimension function optimization.

5. ACKNOWLEDGMENTS

Thanks for the funding of the National Natural Science Foundation of China (No.61075063) and the Special Fund for Basic Scientific Research of Central Colleges, China University of Geosciences (Wuhan) (No.CUGL100230, No.G1323521289).

6. REFERENCES


