Synergization of Different Improvements in Differential Evolution

Neha Agarwal, Anil Kumar Dhami and Joyeeta Neogi
Department of Computer Science & Engineering
Motilal Nehru National Institute of Technology Allahabad
Allahabad, India-211004
{nehaagarwal.jk@gmail.com, anildhami.jk@gmail.com, joyeetan68@gmail.com}

Abstract — An optimization algorithm is a famous choice for solving the complex real world problems. Differential evolution is a robust and powerful evolutionary algorithm for solving the global optimization problem in continuous search space. It has been applied in many engineering and scientific areas. Efficiency of DE truly depends on strategy chosen for trial vector generation the associated control parameter values. In this paper we suggest a synergized improvement for DE which is a combination of two different modifications in DE. Opposition based learning (OBL) method and trigonometric mutation operator is combined together. OBL method provides a good initial population which has fast convergence rate and trigonometric mutation operator makes it more robust. So the combined resulting algorithm improves the overall convergence speed of the algorithm. Experimental results on traditional benchmark functions have shown that synergizing process produces better result.

Keywords: Evolutionary algorithm, differential evolution, mutation operation, nonlinear optimization

I. INTRODUCTION
Optimization is a spontaneous process which is confronted in daily life of researchers, users, and organizations. Optimization is defined as of selecting the best alternative among a given set of options. In optimization problem different decision alternatives exist. In the past few decades several optimization techniques have been developed. Differential Evolution is a stochastic evolutionary optimization algorithm. It is more efficient as compared to the other evolutionary algorithms like genetic algorithm, evolutionary strategy and evolutionary programming. It has been successfully applied in many engineering fields like mechanical engineering [1], [2], communication [3], and pattern recognition [4]. There are three main operators which are involved in DE algorithm including selection, mutation and crossover like the genetic algorithm. Exploration and exploitation are the two important features which decide the success of any evolutionary algorithm. These features are contradictory in nature. Exploration helps in maintaining diversity i.e. it ensures that the solution space is enough explored to get a global solution. On the other hand, exploitation helps to focus on the region around the best solution. DE achieves these two goals by different strategies. In DE mutation and crossover operators favors these two features. Lots of research has been done
for the improvement of performance of DE and even a slight variant improves its performance better. The main aim of this paper is to synergize some of the improvements which had already improved its performance in various respects like its convergent rate.

This paper focuses on two main features of algorithm namely initial population generation and the mutation operator. When there is no prior information available, population based method works well for generating the initial population. Opposition based method is used to generate the initial population which has fast convergence rate. An experimental result has shown that using this method for population generation quality of results has been obtained and the convergence rate of the algorithm has also been improved. This method generates a random number and simultaneously its opposite number. Similarly, trigonometric mutation method has also been proven a good choice for mutation which makes the algorithm more robust. In this paper, these two methods have been combined together which makes algorithm better. Results have shown that proposed synergized differential algorithm (SDE) performs better as compared to these two improved algorithms in individual. However, despite having several attractive features, DE also has some flaws and it does not perform according to the expectation.

This paper is organized as follows: Section 1 gives the brief introduction about DE. Section 2 covers the description about DE and its different operator briefly. Section 3 describes the review of the related work which is done in DE. Section 4 elaborates about synergized differential evolution and section 5 shows the experimental results and finally section 6 concludes about the paper.

II. BASIC DIFFERENTIAL EVOLUTION

Differential evolution (DE), proposed by Storn and Price [5], is a simple and efficient algorithm for solving global optimization problems in the continuous search space. Like any other evolutionary algorithm differential evolution also is a population based stochastic search algorithm. The algorithm starts with a population NP, which is randomly generated in D dimensional real-valued parameter vectors. Each D dimensional vector, also called as individual, forms a candidate solution to the multidimensional optimization problem. Consecutive generations of the DE algorithm is represented as \( G = 0, 1, 2, \ldots G_{\text{max}} \). As the generation evolves value of the parameter vector changes. A vector \( \mathbf{X}_{i,G} = \{ x_{1,i,G}, x_{2,i,G}, x_{3,i,G}, \ldots, x_{n,i,G} \} \), represents the \( i^{th} \) individual in the current population G. The \( j^{th} \) component of the \( i^{th} \) vector can be initialized using the equation:

\[
X_{j,i} = X_{j,\text{min}} + \text{rand}(0,1) \ (X_{j,\text{max}} - X_{j,\text{min}})
\]

For each parameter vector, there may be a certain range within which value of the parameter should lie for the better results. Each candidate solution is evolved to final solution by using three different operators, mutation, crossover, and selection.

A. Mutation

After the population initialization process, mutation operator is applied. In this step, a donor vector is created. For this, three distinct parameter vectors \( \mathbf{X}_{r_1}, \mathbf{X}_{r_2}, \mathbf{X}_{r_3} \) are selected as a target vectors from the current generation. The indices \( r_1, r_2, r_3 \) are mutually exclusive integers, chosen randomly from the population [1, NP]. Now the difference of any two of these three vectors is multiplied by the scaling factor F which gives the value of the donor vector as shown in figure 1. The value of mutation constant lies in the range [0.4, 1].

![Figure 1: Mutation Operator](image)

Various methods for obtaining the donor vector are given below:

- DE/rand/1: \( \mathbf{V}_{i,G} = \mathbf{X}_{r_1,G} + F \ast (\mathbf{X}_{r_2,G} - \mathbf{X}_{r_3,G}) \)
- DE/rand/2: \( \mathbf{V}_{i,G} = \mathbf{X}_{r_1,G} + F^a (\mathbf{X}_{r_2,G} - \mathbf{X}_{r_3,G}) + F (\mathbf{X}_{r_4,G} - \mathbf{X}_{r_5,G}) \)
A. Selection of control parameters

DE has mainly three control parameters namely, population size NP, mutation factor F and the crossover constant CR. A lot of research has been done so far to improve the ultimate performance of algorithm by control parameters. In original DE authors kept all three control parameters fixed during the evolution process and stated that the population size should be in between 5D and 10D, where D represents the dimension of the problem and mutation factor F to be 0.5 for good results [6]. Further research has shown that the value of F should be 0.6 and the population size and CR should be in range of 3D and 8D and in between [0.3, 0.9] respectively [7]. Later on for solving real life problems author [8] suggested values of F to be in range [0.4, 0.95] with F=0.9 and CR= [0, 0.2]. Thus we see that conflicting proposals have been proposed related for manually choosing the control parameters, so adaptive tuning of control parameters came into picture in which the value of control parameters is tuned automatically iteration by iteration. In article [9] proposes encoding of CR into each individual and simultaneously evolving it with other search variables which uses the Gaussian distribution the value of F is generated for each individual. FaDE [10] incorporates fuzzy logic in DE in which inputs incorporate the relative function values and individuals of successive generations to adapt the parameters for the mutation and crossover operations. Article [11] uses a parameter adaptation for DE (ADE) based on the idea of controlling the population diversity, and implemented a multi-population approach.

B. Hybridization of DE with other evolutionary algorithm:

DE is mostly hybridized with Particle swarm optimization (PSO) which is also stochastic based optimization technique. In [13] author proposed algorithm named SDEA in which the DE perturbation approach is used to adapt particle positions and particles positions are updated only if their offspring have better fitness. In [14] author proposed a hybrid version of BareBones PSO and DE called it BBDE, in which they combined the concept of barebones PSO with self adaptive DE strategies. [15] developed a hybrid DE with PSO and applied it to the black box optimization benchmarking for noisy functions. In [16] hybrid DE and PSO called DEPSO was proposed, where three alternative updating strategies are used.

C. Other Improvements

Besides the above two improvements various modifications has also been done. In [17] triangle differential evolution was proposed which uses triangle mutation operator. Paper [18] introduced the concept of opposition based method for initial population generation which has been proven very good for many optimization problems. Some more improvements is also shown in [19][20][21]. Hence we see that lots of work has been done for the performance improvement of DE.

IV. SYNERGIZED DIFFERENTIAL EVOLUTION

This section basically describes the proposed work. An attempt has been made to improve the performance of differential evolution algorithm by combining different improvements together. Synergizing is a process of combining different techniques so that the resulting algorithm improves the overall convergent speed of the algorithm. In the synergized algorithm two different improvements has been combined together. Firstly opposition based method is used for initial population generation because initial population generation technique plays an important role for the success of population based algorithm. Then the triangle mutation operator is used for maintaining the diversity. So these two concepts are described as following:

A. Opposition based optimization for Initial population generation

In opposite based learning method the core idea is to consider a number and its corresponding opposite number in order to get a better population. This is based on the concept of opposite numbers.

Opposite point based random numbers: If \( a \in [l, u] \) is a real number then its opposite number is \( o_a \) can be defined as:

\[
o_a = l + u - a
\]

where \( l \) and \( u \) are the upper and lower bounds and this analogy can be extended to \( n \) dimensional as in [10]. If \( A = (a_1, a_2, \ldots, a_n) \) be a point in \( n \) dimensional space where \( a_i \in [l_i, u_i] \) \( \forall i \in 1, 2, \ldots, n \), then the opposite point in the \( n \) dimensional space can be defined as:

\[
o_a = l + u - a_i
\]

Opposition-based optimization: By applying the opposite point concept, opposite point optimization working can be described as:

- Let \( A = (a_1, a_2, \ldots, a_n) \) be a candidate solution in \( n \) dimensional space within range \( a_i \in [l_i, u_i] \) and \( f(A) \) be its fitness function.
- Opposite point of \( A \) is obtained according to the opposite point definition, which is \( o_A \)
- Calculate fitness of both points i.e. of \( f(A) \) and \( f(o_A) \).

For maximization problem, function having more fitness will be selected to constitute a population i.e. If fitness \( f(A) < f(o_A) \) then replace \( A \) with \( o_A \). Otherwise, continue with \( A \). Here \( A \) and \( o_A \) is evaluated at the same time.

B. Trigonometric Mutation Operator

Trigonometric mutation operator is a local search operator, which was introduced into the differential evolution algorithm, called as trigonometric differential evolution (TDE) which works as follows:
if rand[0, 1] < mutation rate (Γ)

\[ V_{i,G+1} = \frac{(X_{r1}+X_{r2}+X_{r3})}{3} + (p_2 \cdot p_1)(X_{r1}-X_{r2}) + (p_3 \cdot p_2)(X_{r2}-X_{r3}) + (p_1 \cdot p_3)(X_{r3}-X_{r1}) \]

Where

- \( p_1 = \left| \frac{f(X_{r1})}{p} \right| \)
- \( p_2 = \left| \frac{f(X_{r2})}{p} \right| \)
- \( p_3 = \left| \frac{f(X_{r3})}{p} \right| \)

else \[ V_{i,G+1} = X_{r1} + F \cdot (X_{r2} - X_{r3}) \]

This trigonometric mutation gives a method to maintain balance between the convergence rate and the robustness of the algorithm. As a result, convergence velocity of the DE algorithm is accelerated in such a way so that better solutions can be obtained within an acceptable convergence time. Here mutation rate allows a convenient way to keep a good balance between fast convergence and global optimum searching ability.

V. SYNERGIZED DIFFERENTIAL EVOLUTION ALGORITHM (SDE)

Steps of Synergized differential evolution (SDE) can be described as:

**Step1**: Randomly initialize a population \( NP \), in \( n \) dimensional space within range of lower and upper bounds \( X_{j, i} = X_{j, \min} + \text{rand}(0,1) \cdot (X_{j, \max} - X_{j, \min}) \)

**Step 2**: Generate opposite number of \( X_{j, i} \), form the population \( NP' \), i.e. \( X'_{j, i} = X_{j, \max} + X_{j, \min} - X_{j, i} \)

**Step 3**: Calculate the objective function value for \( f(X_{i}) \) and \( f(X'_{i}) \)

**Step 4**: Select \( n \) best individual from the population \([NP, NP']\) based on its fitness

**Step 5**: Select three distinct vectors and generate donor vector \( V_{i,G+1} \), according to eq (1)

**Step 6**: Combine each \( X_{i,G} \), with \( V_{i,G+1} \), to generate a trial vector \( U_{i,G+1} \), as

\[ U_{j,G+1} = \begin{cases} v_{j,G+1} & \text{if } \text{rand} \text{ } j \leq \text{CR} \land j = k \\ X_{j,G} & \text{otherwise} \end{cases} \]

**Step 7**: Calculate fitness of trial vector \( U_{i,G+1} \)

**Step 8**: Select best among the trial vector \( U_{i,G+1} \) and the target vector \( X_{i,G} \), as

\[ X_{i,G+1} = \begin{cases} U_{j,G+1} & \text{if } f(U_{i,G+1}) \leq f(X_{i,G}) \\ X_{i,G} & \text{otherwise} \end{cases} \]

**Step 9**: Go to step 3 and repeat up to step 8 until the termination criteria is satisfied.

VI. EXPERIMENTAL RESULTS & DISCUSSION

To test the performance of this fused algorithm, various benchmark functions have been used which are listed in Table 1. As this algorithm is a combination of two different improvements of DE so this is first compared to the basic DE then with the other two approaches. In the experiment following parameters are used: population size, \( N_p = 100 \), dimension of the problem, \( D = 30 \), mutation constant \( F = 0.85 \) and crossover constant \( C_r = 0.9 \) is used. In order to make a fair comparison total number of function evaluation is kept fixed. We calculated the error and standard deviation after the function evaluation process is terminated. Total function calculation (NFC) is calculated to check the performance of algorithm.

Table 2 contains the data for the number of function calculation for various benchmark functions. Data of NFC shows that SDE algorithm has better results. Table 3 contains the standard deviation value for different algorithms.

---

676
A. Benchmark Functions

<table>
<thead>
<tr>
<th>S.No.</th>
<th>Function</th>
<th>Definition</th>
<th>Range</th>
<th>Minimum</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td>Sphere function</td>
<td>( f_1(x) = \sum_{i=1}^{n} x_i^2 )</td>
<td>( x \in [5.12,5.12] )</td>
<td>( \text{min}(f_1)=f_1(0,.....0)=0 )</td>
</tr>
<tr>
<td>2.</td>
<td>Rosenbrock function</td>
<td>( f_2(x) = \sum_{i=1}^{n-1} (100 \cdot (x_{i+1} - x_i^2)^2 + (1 - x_i)^2) )</td>
<td>( x \in [2, 2] )</td>
<td>( \text{min}(f_2)=f_2(1,1,..1)=0 )</td>
</tr>
<tr>
<td>3.</td>
<td>Rastragin function</td>
<td>( f_3(x) = 10n + \sum_{i=1}^{n} x_i^2 - 10 \cos(2\pi x_i) )</td>
<td>( x \in [5.12,5.12] )</td>
<td>( \text{min}(f_3)=f_3(0,0,..0)=0 )</td>
</tr>
<tr>
<td>4.</td>
<td>Step function</td>
<td>( f_4(x) = \sum_{i=1}^{n} \lfloor x_i + 0.5 \rfloor )</td>
<td>( x \in [100,100] )</td>
<td>( \text{min}(f_4)=0 )</td>
</tr>
<tr>
<td>5.</td>
<td>Eggholder function</td>
<td>( f_5(x) = \sum_{i=1}^{n} x_i \sin \sqrt{</td>
<td>x_{i+1} - x_i - 47</td>
<td>} - (x_i + 47) \cdot \sin \sqrt{</td>
</tr>
<tr>
<td>6.</td>
<td>Michalewicz function</td>
<td>( f_6(x) = \sum_{i=1}^{n} \sin(x_i) \sin(i x_i^2 / \pi)^{2m} )</td>
<td>( x \in [0, \pi], m=10 )</td>
<td>( \text{min}(f_6)=1.8031 )</td>
</tr>
</tbody>
</table>

VII. CONCLUSION

Differential evolution is a powerful optimization technique that has been applied to a wide range of optimization problems. Nevertheless, its performance can be enhanced with the help of certain modifications. The present research article focuses on the concept of synergism. It combines two different techniques in order to increase its efficiency and robustness. Mainly, the combination of different strategies is done in the following ways: firstly, one improved version of the algorithm being used as a pre-optimizer for the initial population, secondly incorporating the mutation operator of another improved DE algorithm as a local search operator.

<table>
<thead>
<tr>
<th>Function</th>
<th>Dimension</th>
<th>DE</th>
<th>ODE</th>
<th>MDE</th>
<th>SDE</th>
</tr>
</thead>
<tbody>
<tr>
<td>F1</td>
<td>30</td>
<td>1700</td>
<td>2100</td>
<td>1900</td>
<td>1300</td>
</tr>
<tr>
<td>F2</td>
<td>30</td>
<td>8500</td>
<td>6700</td>
<td>3900</td>
<td>3800</td>
</tr>
<tr>
<td>F3</td>
<td>30</td>
<td>7980</td>
<td>5000</td>
<td>7600</td>
<td>5000</td>
</tr>
<tr>
<td>F4</td>
<td>30</td>
<td>4000</td>
<td>3400</td>
<td>3500</td>
<td>3350</td>
</tr>
<tr>
<td>F5</td>
<td>30</td>
<td>1750</td>
<td>1650</td>
<td>1360</td>
<td>1030</td>
</tr>
<tr>
<td>F6</td>
<td>30</td>
<td>2000</td>
<td>1980</td>
<td>1670</td>
<td>1600</td>
</tr>
<tr>
<td>Function</td>
<td>Dimension</td>
<td>DE</td>
<td>ODE</td>
<td>TDE</td>
<td>SDE</td>
</tr>
<tr>
<td>----------</td>
<td>-----------</td>
<td>---------</td>
<td>---------</td>
<td>----------</td>
<td>----------</td>
</tr>
<tr>
<td>F1</td>
<td>30</td>
<td>1.0450028e+002</td>
<td>1.9201195e+002</td>
<td>4.0388158e+001</td>
<td>1.332903e+001</td>
</tr>
<tr>
<td>F3</td>
<td>30</td>
<td>3.5422190e+009</td>
<td>2.9446391e+009</td>
<td>3.6643818e-001</td>
<td>3.996760e-001</td>
</tr>
<tr>
<td>F4</td>
<td>30</td>
<td>1.5685608e-002</td>
<td>1.4533225e-001</td>
<td>1.6736344e-001</td>
<td>1.345543e-001</td>
</tr>
<tr>
<td>F5</td>
<td>30</td>
<td>1.0212099e+004</td>
<td>7.2455754e+003</td>
<td>1.4955172e+006</td>
<td>5.24080e+004</td>
</tr>
<tr>
<td>F6</td>
<td>30</td>
<td>6.6033544e+007</td>
<td>5.3421565e+007</td>
<td>5.622552e+006</td>
<td>3.5454522e+005</td>
</tr>
</tbody>
</table>

REFERENCES