Differential Evolution with Nearest & Better Option for Function Optimization

Preprint · October 2018

5 authors, including:

Liang Gao
Huazhong University of Science and Technology
420 PUBLICATIONS **4,882** CITATIONS

Xinyu Li
Huazhong University of Science and Technology
99 PUBLICATIONS **1,544** CITATIONS

Bing Zeng
Huazhong University of Science and Technology
15 PUBLICATIONS **99** CITATIONS

Differential Evolution with Nearest & Better Option for Function Optimization

Haozhen Dong
State Key Lab of Digital Manufacturing Equipment & Technology
Huazhong University of Science and Technology
Wuhan, China
1455921@qq.com

Liang Gao
State Key Lab of Digital Manufacturing Equipment & Technology
Huazhong University of Science and Technology
Wuhan, China
gaoliang@mail.hust.edu.cn

Xinyu Li
State Key Lab of Digital Manufacturing Equipment & Technology
Huazhong University of Science and Technology
Wuhan, China
lixinyu@mail.hust.edu.cn

Haorang Zhong
State Key Lab of Digital Manufacturing Equipment & Technology
Huazhong University of Science and Technology
Wuhan, China
1520098925@qq.com

Bing Zeng
XEMC Windpower Company Limited
Xiangtan, China
zengbing2016@126.com

***Abstract*—Differential evolution is the conventional algorithm with the fastest convergence speed, but it may be trapped local optimal solution easily, so many researchers devote themselves into improve DE. Whale swarm algorithm (WSA) is a new algorithm with niching strategy we proposed previously, it's featured with simple mutation strategy and powerful global search capability, but for functions with high dimensions, it converges slower than conventional algorithms. Based on this fact, we proposed a new DE algorithm, called DE with nearest & better option (NbDE). In order to evaluate the performance of NbDE, we compare NbDE with several meta-heuristic algorithms in nine classical benchmark functions with different dimensions. The result have shown that NbDE outperforms other algorithms in convergence speed and accuracy.**

***Keywords*—Differential evolution; Whale swarm algorithm; Niching strategy; Function optimization; DE with nearest & better option (NbDE)**

I. INTRODUCTION

Meta-heuristic algorithms are becoming powerful in solving numerical optimization problems, especially those can hardly be solved by conventional mathematic method, such as the travelling salesman problem [1], routing problem of wireless sensor networks (WSN) [2], etc. These real-world engineering optimization problems often come with a given mathematical model which is featured with strong nonlinearity and multi-coupling [3]. And classical mathematical method, such as Gradient method, Gauss Newton method, are gradient-based, which means that they may be trapped in local optimal solution easily. And for some problems such as multi-objective coupling problems and discrete problem, the gradient can hardly be calculated. Therefore, many meta-heuristic algorithms, such as genetic algorithm (GA), differential evolution (DE), particle swarm optimization (PSO), have become popular methods for solving engineering problems, for the reason that meta-heuristic is not gradient-based and easily to implement. Among the algorithms mentioned above, DE is the algorithm with fastest convergence speed, but it may be trapped local optimal solution easily. In this paper, inspired by niching strategy, we propose a new DE algorithm for function optimization, called as DE with nearest & better option (NbDE), which based on the differential evolution (DE) and whale swarm algorithm (WSA). Here, a brief overview of DE and WSA is presented.

Storn and Price proposed differential evolution (DE) algorithm for function optimization [4]. It contains three parts, called mutation, crossover and selection, which are similar to the famous genetic algorithm (GA). Firstly, a reference vector for each individual, which can be called target vector, is generated by using mutation strategy of DE algorithm. Then, the crossover operation takes place between the target vector and the original vector, and the candidate vector for each individual is created by selecting elements from the target vector and the original one by using crossover method. The last, a comparison between the fitness value of the candidate vector and the original one will determine which one will transmit into the next generation. Since put forward, DE algorithm has gained increasing popularity. Many researchers and engineers have proposed various ideas for using DE to solve real-world optimization problems [5][6][7]. Although DE is featured with fast convergence speed and simple mutation strategy, it often fall into local optimal solution, so improvement for DE have become a hot topic.

WSA [8][9] is a new meta-heuristic algorithm proposed by us previously, which is inspired by communicating behavior of whales. WSA uses a special mutation strategy as follows:

$$v_{i,j}^G = x_{i,j}^G + rand(0, \rho_0 * e^{-\eta d_{X,Y}}) * (y_{i,j}^G - x_{i,j}^G) \quad (1)$$

Where, $v_{i,j}^G$ denotes the j-th elements of candidate vector v_i^G at G iteration and corresponding $x_{i,j}^G$ denotes the j-th elements of x_i^G's position, $y_{i,j}^G$ represents the j-th element of "the nearest and better whale of x_i^G" y_i^G's position at G iteration. The $rand(0, \rho_0 * e^{-\eta d_{X,Y}})$ means generating a random numbers ranges from 0 to $\rho_0 * e^{-\eta d_{X,Y}}$. ρ_0 is the intensity of ultrasound at the origin of source, which can be set to 2 for almost all the cases. e denotes the natural constant and η represents the attenuation coefficient. And $d_{\mathbf{X},\mathbf{Y}}$ is the Euclidean distance between **X** and **Y**. In reference [9], Zeng have proved that η could to be set to 0 for most cases. So the Eq.1 can be simplified to the following form.

$$v_{i,j}^G = x_{i,j}^G + 2 * rand(0,2) * (y_{i,j}^G - x_{i,j}^G) \quad (2)$$

According to Eq.1 and Eq.2, a whale would move positively and randomly under the guidance of its "nearest and better" whale which is close to it, and move negatively and randomly under the guidance of that whale which is quite far away from it, and this can be treated as a new niching method. Simulation results have shown that WSA outperforms several classical niching methods especially in multimodal function optimization. Despite the fact that WSA can maintain population diversity during searching process and has strong local searching ability, drawback also exists, the convergence speed of WSA is relatively slow especially for objective function with high-dimensions. One reason which cause this problem above is that the "nearest & better" option helps us shrink the range of searching, which improve the local search ability but reduce the speed of convergence. So for objective function, some improvements should be considered.

The remainder of this paper is organized as follows: The framework about NbDE method is presented in section 2. In section 3, several benchmarks for comparison and configurations of all the algorithms are introduced. And then the NbDE simulation results is shown, we compared the simulation results get by NbDE with those get by several comparison methods. The last section is the conclusions and topics for further research.

II. THE FRAMEWORK OF PROPOSED NBDE

In this section, the framework of NbDE is introduced. This algorithm implements the mutation strategy "DE/rand-to-nearest & better/2", which derives from the classical DE algorithm and is inspired by WSA.

A. *Mutation*

The mutation strategy "DE/rand/1" is designed for the classical DE [10][11], which have been proved efficient for solving engineer problems in literature [12]. However, drawbacks such as premature convergence also exist, which limited the further application of DE. For this reason, many researchers have proposed their solutions. R. Gamperle [13] have found that DE with "DE/best/2" strategy may outperform the original DE in many problems, and [14] proposed the mutation strategy "DE/best/1" to solve technical problems. The famous "JADE" proposed by Jingqiao Zhang [15] used "DE/current-to-pbest" combining with some other strategies, JADE have achieved a set of satisfactory results for benchmark functions.

The mutation strategy is utilized for creating the candidate vector $v_{i,G}$, "DE/rand/1" is a classical strategy which has been applied in classical DE and some other derived algorithms, and it can be described by the following function:

$$v_i^G = x_{r1}^G + F \times (x_{r2}^G - x_{r3}^G), r1 \neq r2 \neq r3 \neq i \quad (3)$$

Where G denotes the number of iteration, r1, r2, r3 represent the individuals ID which are randomly selected from the current population, F is the mutation operator parameter which is used for scaling the differential vector.

As mentioned above, the WSA mutation strategy can be summarized as follows:

$$v_{i,j}^G = x_{i,j}^G + 2 * rand(0,2) * (y_{i,j}^G - x_{i,j}^G) \quad (4)$$

Basic on the fact above, we proposed a hybrid mutation strategy as follows:

$$v_i^G = x_{r1}^G + rand(0,1) \times (x_{r2}^G - x_i^G) + rand(0,1) \times (y_i^G - x_i^G) \quad (5)$$

Similar to mutation strategies above, G denotes the number of iteration, r1, r2 represent the random individuals ID chosen from the current population, y_i^G denotes the nearest individual with better fitness value of x_i^G and when x_i^G is the best individual of current population we will choose a random individual for y_i^G.

B. *Crossover*

Different from classical DE, we provided a crossover strategy which hybrids binary crossover, exponential crossover and non-crossover operator. First, a random number is generated to determine which crossover operator will be selected. Then the crossover operation is implemented between $v_{i,G}$ and $x_{i,G}$, and the final candidate $v_{i,G}$ will be generated.

1) Binary crossover

The binary crossover of NbDE can be described as follows:

$$v_{i,j}^G = \begin{cases} v_{i,j}^G & if\ Rand \leq CR \\ x_{i,j}^G & otherwise \end{cases}, i = 1,2 \cdots NP; j = 1,2 \cdots D \quad (6)$$

Where Rand denotes a random number with the range from 0 to 1, CR represents the crossover control parameter, NP denotes the size of population and D denotes the dimension of each individual.

TABLE.I THE PSEUDO CODE OF NBDE

Input: An objective function, options of the NbDE. Output: The global optima.	
1:	Begin
2:	Initialize parameters;
3:	Initialize a group of individuals;
4:	Evaluate each individual (calculate their fitness values);
5:	while termination criterion is not satisfied do
6:	For i=1 to NP
7:	Create a new individual v_i^G by Eq.25;
8:	Crossover v_i^G with x_i^G;
9:	Evaluate the new individual v_i^G;
10:	If $f(v_i^G) < f(x_i^G)$
11:	$x_i^{G+1} = v_i^G$;
12:	End If
13:	End for
14:	End

2) Exponential crossover

The exponential crossover of NbDE can be expressed as follows:

$$v_{i,j}^G = \begin{cases} v_{i,j}^G & if\, j_{down} \le j \le j_{up} \\ x_{i,j}^G & otherwise \end{cases}, j_{up} = j_{down} + randi(D) \quad (7)$$

$$randi(D) = sum(rand(1, D) \le CR) \quad (8)$$

Where j_{up} and j_{down} denote the start and end dimension of exponential crossover, $randi(D)$ denotes the number of elements no more than CR in random vector $rand(1, D)$. When $j_{up} > D$, this operator can be written in this form:

$$v_{i,j}^G = \begin{cases} v_{i,j}^G & if\, j_{down} \le j \; or \; j \le j_{up} - D \\ x_{i,j}^G & otherwise \end{cases}, i = ,2 \cdots NP; j = 1,2 \cdots D \quad (9)$$

C. Selection

When a candidate individual v_i^G is created by previous steps, there is a comparison between $f(v_i^G)$ and $f(x_i^G)$, the individual with better fitness function value will survive into the next iteration. This greedy selection can be shown as follows:

$$x_i^{G+1} = \begin{cases} v_i^G & if\ f(v_i^G) \le f(x_i^G) \\ x_i^G & otherwise \end{cases}, i = 1,2 \cdots NP \quad (10)$$

III. SIMULATION OF NBDE

In order to verify the feasibility of proposed NbDE, NbDE is applied to minimize a set of 9 benchmark functions with different dimensions (D=10, D=30, D=50) as shown in Table. 2. NbDE is compared with the famous adaptive DE algorithms JADE, the classic DE/rand/1, the DE/best/2, the classic genetic algorithm (GA), the classic particle swarm optimization (PSO) and the WSA. For fair comparison, all methods are allowed to evaluate the objection functions with maximum 10000D times. Based on the suggestions from original papers, other configurations of all the algorithms mentioned above are shown as follows:

- NbDE: F=rand(0,1); CR=rand(0.4,0.9); NP=40
- DE/rand/1: F=0.5; CR=0.9; NP=30(D=10); NP=100(D=30); NP=200(D=50).
- DE/best/2: F=0.5; CR=0.9; NP=30(D=10); NP=100(D=30); NP=200(D=50).
- JADE [15]: p=0.05; c=0.1; CR=0.9; NP=30(D=10); NP=100(D=30); NP=200(D=50).
- WSA [9]: = 0 ; NP=40.
- GA [16]: CP=0.95; MP=0.05; NP=40(D=10); NP=100(D=30); NP=200(D=50).
- PSO [17]: C1=2.05; C2=2.05; K=0.729; vMax=2; vMin=-2; NP=40(D=10); NP=100(D=30); NP=200(D=50).

All of these methods are implemented with Matlab 2014b and executed on a personal PC with 3.4 GHz Intel Xeon E3-1230-V5 processor, 16 GB RAM and 64-bit Microsoft windows 10 operating system.

All of the test functions mentioned above are calculated by each algorithm for 50 independent runs, and the results shown in Table. 3-8 are organized by the dimensions and evaluation indexes of test functions. In these tables, four significant statistical results included the mean value and standard deviation (STD) of the functions results, the success rate (SR) and its rank are given. For success rate statistics, the value-to-reach (VTR) was set to 1E-02 for F5, while 1E-08 for others. Computing time is not given in this comparison for the reason that it is not a criterion to be investigated here. The NbDE we proposed is slower in limited ranges compared with classical DE because of the calculation of the nearest & better individual in each evaluation which will cost a little computing time.

TABLE. II BENCHMARK FUNCTION DEFINITIONS

Fn	Test Function Name	Bounds	Optimum value
F1	Zakharov	$[-100,100]^D$	0
F2	Schwefel 2.22	$[-10,10]^D$	0
F3	Schwefel 2.21	$[-100,100]^D$	0
F4	Rosenbrock	$[-30,30]^D$	0
F5	Noise Quartic	$[-1.28,1.28]^D$	0
F6	Schwefel 2.26	$[-500,500]^D$	-418.9828872724339D
F7	Rastrigin	$[-5.12,5.12]^D$	0
F8	Ackley	$[-32,32]^D$	0
F9	Griewank	$[-600,600]^D$	0

A. Success Rate

The success rates of NbDE and other algorithms on benchmark functions are listed in Table 3-6. When the two algorithms get the same success rate on a test function, they will get the same ranks over this test function. The last row of this table shows the total ranks of all algorithms, which are the summation of individual ranks on each test function.

TABLE.III SR AND RANKS OF ALL ALGORITHMS WHEN D=10

Fun	NbDE	DE/rand/1	DE/best/2	JADE	WSA	GA	PSO
F1	**1/1**	0/5	0/5	0.26/4	0.98/3	0/5	1/1
F2	**1/1**	**1/1**	**1/1**	**1/1**	0.86/5	0/6	0/6
F3	**1/1**	0/4	**1/1**	0.02/3	0/4	0/4	0/4
F4	**0.98/1**	0/3	0/3	0/3	0/3	0/3	0.74/2
F5	**1/1**	**1/1**	**1/1**	**1/1**	0.96/6	0.92/7	**1/1**
F6	**1/1**	0.46/2	0.42/3	0.24/4	0/5	0/5	0/5
F7	**1/1**	0.14/4	0.98/2	0.94/3	0/5	0/5	0/5
F8	**1/1**	**1/1**	**1/1**	0.9/4	0.72/5	0/7	0.52/6
F9	0.68/2	0.56/3	**0.9/1**	0.46/4	0/5	0/5	0/5
Total Rank	**10**	24	18	27	41	47	35

TABLE.IV SR AND RANKS OF ALL ALGORITHMS WHEN D=30

Fun	NbDE	DE/rand/1	DE/best/2	JADE	WSA	GA	PSO
F1	**1/1**	0/2	0/2	0/2	0/2	0/2	0/2
F2	**1/1**	**1/1**	**1/1**	**1/1**	0.5/5	0/6	0/6
F3	**1/1**	0/2	0/2	0/2	0/2	0/2	0/2
F4	**0.94/1**	0/2	0/2	0/2	0/2	0/2	0/2
F5	**1/1**	0/7	**1/1**	**1/1**	0.2/6	0.66/5	0.98/4
F6	**1/1**	0.54/3	0.58/2	0/4	0/4	0/4	0/4
F7	0.92/2	0.18/3	**1/1**	0/4	0/4	0/4	0/4
F8	**1/1**	**1/1**	**1/1**	0.1/4	0/5	0/5	0/5
F9	0.9/3	1/1	**1/1**	0.84/4	0.44/5	0/6	0/6
Total Rank	**14**	22	**14**	24	35	36	34

As we can see from Table 3-6, NbDE get the 100% success rate on F1, F2, F3, F5, F6, F7, F8 when D=10, on F1, F2, F3, F5, F6, F8 when D=30 and on F1, F2, F5, F8 when D=50. For most functions, NbDE get the highest success rate, but the success rate of NbDE on F9 is only a little bit lower than that of DE/best/2 when D=10. D=30 and D=50 and DE when D=30, and the success on F7 is a little bit lower than that of DE/best/2 when D=30, but is far greater than those of other algorithms.

TABLE.V SR AND RANKS OF ALL ALGORITHMS WHEN D=50

Fun	NbDE	DE/rand/1	DE/best/2	JADE	WSA	GA	PSO
F1	**1/1**	0/2	0/2	0/2	0/2	0/2	0/2
F2	**1/1**	0/5	**1/1**	0.38/3	0.14/4	0/5	0/5
F3	**0/1**	**0/1**	**0/1**	**0/1**	**0/1**	**0/1**	**0/1**
F4	**0.3/1**	0/2	0/2	0/2	0/2	0/2	0/2
F5	**1/1**	0/6	0.02/5	**1/1**	0/6	0.14/4	0.92/3
F6	**0.68/1**	0.28/2	0.02/3	0/4	0/4	0/4	0/4
F7	**0/1**	**0/1**	**0/1**	**0/1**	**0/1**	**0/1**	**0/1**
F8	**1/1**	0/3	1/1	0/3	0/3	0/3	0/3
F9	0.96/2	0/5	**1/1**	0.14/4	0.32/3	0/5	0/5
Total rank	**10**	27	17	21	26	27	26

Therefore, it can be concluded that NbDE outperforms other algorithms on success rate when solving functions. It also can be seen that the best performance of NbDE on success rate because that the total rank of NbDE is much smaller than those of other algorithms when D=10, D=50. We notice that NbDE and DE/best/2 got the same total rank when D=30. But we can see in Table.4 the rank of DE/best/2 on F1, F3, F4 is 2 while the SR of DE/best/2 is 0 which means that DE/best/2 cannot get exactly results on these benchmark functions, but NbDE can get exactly results in most cases, so we can concluded that NbDE outperform DE/best/2 when D=30.

B. Quality of Optima Found

In this part, NbDE is compared with other algorithms in terms of the accuracy of optima found. As we can see from Table. 6-8, we notice that NbDE get the best accuracy of optima found on F1, F3, F4, F6, F7, F8, F9 when D=10, on F1, F2, F3, F5, F6, F8 when D=30, on F1, F2, F3, F4, F6, F8 when D=50.

TABLE.VI QUALITY OF OPTIMA FOUND OF ALL ALGORITHMS WHEN D=10 (MEASUREMENT: MEAN/STD)

Fun	NbDE	DE/rand/1	DE/best/2	JADE	WSA	GA	PSO
F1	**3.3E-40/1.6E-39**	6.8E+3/2.8E+3	2.7E+2/1.4E+2	1.5E-1/8.2E-1	5.3E-10/3.0E-9	2.1E+4/7.7E+3	4.8E-25/2.7E-24
F2	5.3E-39/7.5E-39	5.2E-40/4.9E-40	**7.8E-47/6.5E-47**	3.2E-12/2.2E-11	2.26E+0/1.1E-2	1.3E-3/7.3E-4	1.4E-2/1.5E-2
F3	**6.6E-23/1.1E-22**	3.9E-06/3.0E-6	1.2E-12/6.4E-13	2.6E-2/4.5E-3	4.0E+0/2.2E+0	4.1E-1/2.8E-1	1.8E-2/1.5E-2
F4	**8.0E-2/5.6E-1**	6.7E+0/8.3E+0	1.0E+0/1.9E+0	5.9E+0/2.3E+0	2.0E+1/3.2E+1	1.4E+1/8.8E+0	1.0E+0/1.8E+0
F5	1.2E-3/5.2E-4	3.2E-3/1.2E-3	1.4E-3/5.2E-4	**3.8E-4/2.5E-4**	4.1E-3/3.0E-3	5.6E-3/2.6E-3	1.9E-3/1.6E-3
F6	**-418.98E+1/1.8E-12**	-409.85E+1/1.2E+2	-409.93E+1/9.1E+1	-402.88E+1/1.3E+2	-336.3E+1/2.7E+2	-134.86E+1/2.5E+2	-221.21E+1/5.2E+2
F7	**0/0**	1.4E+0/1.2E+0	2.0E-2/1.4E-1	6.0E-2/2.3E-1	9.4E+0/5.8E+0	7.3E-5/1.4E-4	1.0E+1/3.6E+0
F8	**4.2E-15/8.5E-16**	4.5E-15/5.0E-16	4.4E-15/0	2.3E-2/1.6E-1	3.7E-1/6.8E-1	5.9E-3/3.3E-3	7.9E-1/9.3E-1
F9	**2.1E-4/1.1E-3**	3.3E-3/5.3E-3	7.9E-4/2.4E-3	9.5E-3/1.1E-2	8.6E-2/6.7E-2	6.6E-2/2.6E-2	4.5E-1/2.7E-1

TABLE.VII QUALITY OF OPTIMA FOUND OF ALL ALGORITHMS WHEN D=30 (MEASUREMENT: MEAN/STD)

Fun	NbDE	DE/rand/1	DE/best/2	JADE	WSA	GA	PSO
F1	**5.4E-33/8.6E-33**	6.3E+4/7.5E+3	1.6E+4/2.8E+3	4.0E-2/6.2E-2	2.18E+2/1.5E+3	8.1E+4/1.6E+4	4.6E-3/3.0E-3
F2	**1.1E-50/1.6E-50**	3.1E-15/7.23E-16	2.3E-27/5.7E-28	4.7E-11/3.3E-10	2.39E+0/4.7E+0	2.2E-2/8.6E-3	6.3E-1/5.6E-1
F3	**3.9E-17/8.8E-17**	5.5E+0/4.8E-1	5.6E-4/9.0E-5	8.0E-1/4.1E-1	2.5E+1/6.1E+0	6.1E+0/1.5E+1	2.5E-1/2.1E-1
F4	**2.3E-2/9.5E-1**	6.4E+1/2.0E+1	2.7E+1/1.3E+1	3.6E+1/1.8E+1	1.9E+3/1.3E+4	2.5E+2/2.3E+2	2.8E+1/1.0E+1
F5	**2.0E-3/6.4E-4**	2.7E-2/5.5E-3	6.1E-3/1.4E-3	8.3E-4/2.7E-4	1.1E-1/3.7E-1	9.2E-3/3E-3	4.0E-3/2.4E-3
F6	**-125.69E+2/7.3E-12**	124.89E+2/1.2E+2	125.08E+2/1.0E+2	-107.87E+2/7.0E+2	-729.15E+1/7.0E+2	-261.42E+1/4.0E+2	-628.17E+1/1.2E+3
F7	1.0E-1/3.6E-1	7.8E-1/8.5E-1	**0/0**	4.5E+1/5.0E+0	1.1E+2/3.5E+1	1.3E-1/3.2E-1	4.4E+1/1.4E+1
F8	**6.2E-15/1.8E-15**	5.0E-10/4.5E-10	8.9E-15/1.6E-15	1.4E-5/5.7E-5	5.6E+0/2.2E+0	3.4E-2/1.1E-2	2.6E+0/8.0E-1
F9	1.0E-3/3.3E-3	0/0	**0/0**	7.9E-4/3.2E-3	1.5E-2/1.9E-2	1.1E-1/7.7E-2	1.1E-2/1.3E-2

TABLE.VIII QUALITY OF OPTIMA FOUND OF ALL ALGORITHMS WHEN D=50 (MEASUREMENT: MEAN/STD)

Fun	NbDE	DE/rand/1	DE/best/2	JADE	WSA	GA	PSO
F1	**1.9E-21/7.3E-21**	1.3E+5/1.2E+4	6.1E+4/5.3E+3	8.1E+3/1.4E+3	4.1E+3/4.1E+3	1.4E+5/2.4E+4	4.65E-2/1.5E-2
F2	**1.4E-61/1.6E-61**	2.7E-6/2.8E-7	2.7E-17/3.6E-18	3.2E-6/1.2E-5	1.5E+1/2.6E+1	2.0E-1/5.1E-2	1.6E+0/8.5E-1
F3	**8.9E-2/1.7E-1**	4.2E+1/1.6E+0	1.4E-1/1.4E-2	1.6E+0/4.7E-1	3.6E+1/7.2E+0	2.4E+1/5.3E+0	6.5E-1/3.0E-1
F4	**2.4E-1/9.6E-1**	2.4E+2/1.8E+1	4.7E+1/1.1E+1	7.0E+1/2.8E+1	3.7E+3/1.8E+4	7.8E+2/3.6E+2	6.1E+1/2.9E+1
F5	3.5E-3/1.0E-3	1.3E-1/1.6E-1	1.3E-2/2.0E-3	**1.3E-3/3.1E-4**	1.2E+0/6.8E+0	1.4E-2/3.8E-3	5.1E-3/3.7E-3
F6	**-209.04E+2/7.1E+1**	-207.70E+2/2.1E+2	-196.81E+2/4.6E+2	-111.50E+2/4.4E+2	-111.25E+2/1.0E+3	-383.82E+1/6.0E+2	-102.79E+2/1.5E+3
F7	3.7E+0/1.7E+0	6.5E+2/6.5+0	**4.5E-2/4.1E-2**	1.7E+2/8.7E+0	2.6E+2/4.6E+1	1.3E+1/3.8E+0	6.5E+1/1.5E+1
F8	**9.3E-15/2.7E-15**	9.3E-2/2.1E-2	2.1E-14/1.4E-15	2.2E-4/1.8E-4	1.2E+1/3.4E+0	8.0E-1/4.9E-1	3.3E+0/4.7E-1
F9	7.9E-4/4.4E-3	7.1E-6/1.6E-6	**0/0**	9.2E-4/3.1E-3	4.1E-2/1.0E-1	9.8E-1/8.1E-2	7.1E-3/7.5E-3

What more, We can also notice that the mean value and standard deviation of F9 reach by NbDE is relatively small(less than 5E-3), considering that the success (SR) of F9 when D=30 and D=50 is even higher, so we can conclude that NbDE have jumped out of local optimal and need more iterations for convergence in most cases. Based on this analysis, we can also make the similar conclusion for F4 when D=50. Table.7 and Table.8 have shown that DE/best/2 get the best result for F7 and F9 when D=30 and D=50, but it can hardly get a feasible solution for F1, F3, F4 when D=30, F1, F3, F4, F5, F6, F7 when D=50. So we can also conclude that NbDE outperforms DE/best/2. We can also find that JADE underperforms in many cases when the number of evaluations is limited into 10000D while it can get satisfactory solutions within more iterations, so we can conclude that JADE converge slower than NbDE. GA and PSO is classical meta-heuristic algorithms for function optimization, in this experiment we can observe that NbDE outperforms GA and PSO. WSA is a new meta-heuristic method proposed by us, it's featured with strong local search ability but its convergence speed is relatively slow especially for high dimensions problems which can also be demonstrated in our experiment. From solutions above, we can find that NbDE outperforms other algorithms and is featured with fast convergence speed and small population size because of the improved mutation and crossover strategy.

IV. Conclusion

A meta-heuristic algorithm called NbDE with nearest & better option, inspired by whale swarm algorithm and differential evolution, is proposed for function optimization in this paper. NbDE is compared with several popular meta-heuristic algorithms on four performance metrics. Experimental results have shown that NbDE outperforms other algorithms in success rate and solution quality of benchmark functions. In the future we will focus on the aspects:

1) Utilizing NbDE to solve engineering optimization problems;
2) Improve NbDE with parallel computing optimization.

Acknowledgment

This research was supported by the National Natural Science Foundation for Distinguished Young Scholars of China under Grant No.51825502, National Natural Science Foundation of China (NSFC) (51775216, and 51721092), Natural Science Foundation of Hubei Province (2018CFA078) and the Program for HUST Academic Frontier Youth Team.

References

[1] Mahi, M., Baykan, Ö.K., Kodaz, H.: A new hybrid method based on particle swarm optimization, ant colony optimization and 3-opt algorithms for traveling salesman problem. Appl. Soft Comput. 30, 484–490 (2015)

[2] Zeng, B., Dong, Y.: An improved harmony search based energy-efficient routing algorithm for wireless sensor networks. Appl. Soft Comput. 41, 135–147 (2016)

[3] Strogatz, S. (2015). Nonlinear Dynamics and Chaos. Boca Raton: CRC Press.Storn, R., Price, K.: Differential Evolution - A Simple and Efficient Adaptive Scheme for Global Optimization Over Continuous Spaces. ICSI, Berkeley (1995)

[5] Qing, A.: Dynamic differential evolution strategy and applications in electromagnetic inverse scattering problems. IEEE Trans. Geosci. Remote Sens. 44(1), 116–125 (2006)

[6] Gao, Z., Pan, Z., Gao, J.: A new highly efficient differential evolution scheme and its application to waveform inversion. IEEE Geosci. Remote Sens. Lett. 11(10), 1702–1706 (2014)

[7] Das, S., Suganthan, P.N.: Differential evolution: a survey of the state-of-the-art. IEEE Trans. Evol. Comput. 15(1), 4–31 (2011)

[8] Zeng, Bing, L. Gao, and X. Li. Whale Swarm Algorithm for Function Optimization. Intelligent Computing Theories and Application. 2017.

[9] Bing Zeng. Whale swarm algorithm with iterative counter for multimodal function optimization, 2018, arXiv:1804.02851v1

[10] R. Storn and K. Price, "Differential evolution a simple and efficient heuristic for global optimization over continuous spaces," J. Global Optimization, vol. 11, no. 4, pp. 341 - 359, 1997.

[11] K. V. Price, R. M. Storn, and J. A. Lampinen, Differential Evolution: A Practical Approach to Global Optimization, 1st ed. New York: SpringerVerlag, Dec. 2005.

[12] "Differential evolution for multiobjective optimization," in Proc. IEEE Congr. Evol. Comput., Dec. 2003, pp. 2696–2703.

[13] R. Gamperle, S. D. Muller, and P. Koumoutsakos, "A parameter study for differential evolution," in Proc. Advances Intell. Syst., Fuzzy Syst., Evol. Comput., Crete, Greece, 2002, pp. 293–298.

[14] U. Pahner and K. Hameyer, "Adaptive coupling of differential evolution and multiquadrics approximation for the tuning of the optimization process," IEEE Trans. Magnetics, vol. 36, no. 4, pp. 1047–1051, Jul. 2000.

[15] J., Z. and C.S. A., JADE: Adaptive Differential Evolution With Optional External Archive. IEEE Transactions on Evolutionary Computation, 2009. 13(5): p. 945-958.

[16] Yuan, X., et al., A Genetic Algorithm-Based, Dynamic Clustering Method Towards Improved WSN Longevity. Journal of Network and Systems Management, 2017. 25(1): p. 21-46.

[17] Bansal J.C. (2019) Particle Swarm Optimization. In: Bansal J., Singh P., Pal N. (eds) Evolutionary and Swarm Intelligence Algorithms. Studies in Computational Intelligence, vol 779. Springer, Cham