Explore Influence of Differential Operator in DE Mutation with Unrestrained Method to Generate Mutant Vector

Hao Liu[1], Han Huang[1,2], and Shusen Liu[1]

[1] School of Software Engineering South China University of Technology Guangzhou, P.R. China
[2] Department of Management Sciences, College of Business City University of Hong Kong, Hong Kong
liuhaoscut@gmail.com, hhan@scut.edu.com

Abstract. Differential Evolution (DE) is an efficient optimizer in current use. Although many new DE mutant vectors have been proposed by alter the differential operator, there are few works studying the differential operator's effect in DE algorithm. This paper proposes a correlation between the DE performance and the mutant vector. That is, for a particular mutant vector, increase the number of differential operator would influence the performance of the algorithm linearly. These mutant vectors are evaluated by 23 benchmarks selected from Congress on Evolutionary Computation (CEC) competition. Additionally, this paper proposes an unrestrained method to generate mutant vector. Unlike the old method selects mutually exclusive individuals, the new method allows same individuals appear repeatedly to generate mutant vector. This new method could enhance the potential diversity of the population and improve the performance of DE in general. *abstract* environment.

Keywords: Differential Evolution (DE), differential operator, mutant vector generation.

1 Introduction

Since 1995 Storn and Price proposed DE, it is accepted widely as an excellent and reliable function optimizer. DE is a special case of evolutionary algorithm, it distinguished to other EAs because it generates offspring by a scaled difference perturb vector. Ferrante Neri and Ville Tirronen[1] have given an overview of DE, Swagatam Das[2] has given a conclusion of recent years development and future trend of DE. In forestall research, except noise problems [3][4], DE achieved excellent result to most benchmarks. In previous CEC, DE is best as an applicable evolutionary algorithm. Although recent papers show some strong EA like restart covariance matrix adaptation ES (CMA-ES) outperforms classical and adaptive DE at CEC2005 competition, DE is still outstanding to solve real-valued test functions.

As DE is simple and effective, various mutate vectors have been proposed to get further optimization. Storn and Price have suggested a mutant vector family and many other scientists have expanded the vector family. It is easy to perceive that some vectors are improved by add one more differential operator, however, few papers have pay attention to the performance tendency of DE with various differential operator. With this consideration, this paper explores the relation between DE performance and the number of differential operator in mutant vector.

In the following part this paper proposes a new method to generate mutant vector. Classical DE select individuals in the population randomly and subtract other randomly selected individuals to gain perturb parameter. This paper proposes an unrestrained method which could enhance potential diversity of the population. The new method improves the performance of DE steadily. In this paper, 2.1 concludes framework of DE, 2.2 presents classical mutate vectors and the author gives new mutate vectors, 2.3 proposes a new method to generate effective mutate vector. Part 3.1 introduces experiment benchmarks and 3.2 presents the result of experiments and analysis. The paper gives a conclusion in part 4.

2 Benchmark Optimization by DE

2.1 Framework of DE

DE optimization means to find the minimum value of objective function f(x). This problem can be encoded as a NP population with D dimension parameters vector $X = [x_1, x_2, \ldots, x_{NP}]$,initial population distributed in search space S randomly. The goal of the algorithm is to find out $x_{min} \in S$, by using pre-prepared benchmarks.

The framework of DE:

a) Initialization

For each individual with D dimension at G generation $x_i = \{x_{i1}^G, x_{i2}^G, \ldots, x_{iD}^G\}$, there have a certain range for each dimensions, the initial population should randomly distributes in a prescribed space.

b) Mutation

DE employs a mutate vector to perturb individuals to get a mutation in the search space. The classical vector could be expressed as

$$V_i^G = x_{r1}^G + F(x_{r2}^G - x_{r3}^G) \tag{1}$$

r1, r2, r3 are integers randomly selected in the range [1, NP].

c) Crossover

After mutation, for each individual xi crossover operation is used to generate a trial vector $U_i^G = [u_{1i}^G, u_{2i}^G, \ldots\ u_{Di}^G]$,. This paper uses binomial crossover to generate trail vector.

d) Select

In this step the select operation if the trail vector has less or equal benchmark value than the target vector, the trail vector would replace the parent as a member of the offspring, otherwise the target vector would remain in the next generation.

b)c)d)would be repeated until certain criterion is met.

2.2 Construct New Mutation Family

The difference vector (1) has only one scaled difference to perturb the population, which is known as DE/rand/1. Storn and Price suggest other difference vectors known as:

1.$DE/best/1 : V_i^G = x_{best} + F(x_{r2}^G - x_{r3}^G)$
2.$DE/best/2 : V_i^G = x_{best} + F(x_{r2}^G - x_{r3}^G) + F(x_{r4}^G - x_{r5}^G)$
3.$DE/rand/2 : V_i^G = x_{r1}^G + F(x_{r2}^G - x_{r3}^G) + F(x_{r4}^G - x_{r5}^G)$
4.$DE/cur - to - best/1 : V_i^G = x_i^G + F(x_{best}^G - x_{r1}^G) + F(x_{r2}^G - x_{r3}^G)$
5.$DE/rand - to - best/1 : V_i^G = x_{r1}^G + F(x_{best}^G - x_{r2}^G) + F(x_{r3}^G - x_{r4}^G)$

r1, r2, r3, r4, r5 are integers randomly selected in the range [1, NP]. is the best individual in G generation. These vectors could be seen at [6].

It is easy to perceive rand/1, rand/2 and best/1, best/2 all achieve acceptable result and they are in regular pattern. Based on this observation, this paper gives a hypothetical that increase the number of differential operator would have influence on the performance on DE algorithms. These vectors are expanded by adding difference parameters to rand/3 and best/3. Moreover, cur-to-best/1 and cur-to-rand/1 vector could derive new vectors best-to-cur/1, rand-to-cur/1 by changing the parameter positions. In the same theory of expand best/1, cur-to-best/1 and rand-to-best/1 could expand to cur-to-best/2 and rand-to-best/2, etc.

These new vectors are listed below:

6.$DE/best/3 : V_i^G = x_{best} + F(x_{r2}^G - x_{r3}^G) + F(x_{r4}^G - x_{r5}^G) + F(x_{r6}^G - x_{r7}^G)$
7.$DE/rand/3 : V_i^G = x_{r1}^G + F(x_{r2}^G - x_{r3}^G) + F(x_{r4}^G - x_{r5}^G) + F(x_{r6}^G - x_{r7}^G)$
8.$DE/best - to - cur/1 : V_i^G = x_{best} + F(x_i^G - x_{r1}^G) + F(x_{r2}^G - x_{r3}^G)$
9.$DE/cur - to - rand/1 : V_i^G = x_i^G + F(x_{r1}^G - x_{r2}^G) + F(x_{r3}^G - x_{r4}^G)$
10.$DE/rand - to - cur/1 : V_i^G = x_{r1}^G + F(x_i^G - x_{r2}^G) + F(x_{r3}^G - x_{r4}^G)$
11.$DE/cur{-}to{-}best/2 : V_i^G = x_i^G + F(x_{best}^G - x_{r1}^G) + F(x_{r2}^G - x_{r3}^G) + F(x_{r4}^G - x_{r5}^G)$
12.$DE/rand - to - best/2 : V_i^G = x_{r1}^G + +F(x_{best}^G - x_{r2}^G) + F(x_{r3}^G - x_{r4}^G) + F(x_{r5}^G - x_{r6}^G)$

To prove this hypothesis, the following new mutant vectors will be tested in part 3.

These vectors perhaps not excellent enough to handle test functions, but they are useful to present performance of vectors with different difference parameter.

As the vector has a simple structure, change the value of control parameter or change the different operator are common methods to optimize the vector.

2.3 Improved Method to Generate Mutant Vector

Since 1950s, with the idea of using Darwinian principles to solve problems, evolutionary computation emerges distant ideas as a competitive discipline. This part we propose a new method to generate mutant vector and try to explain it by

Darwinian principles. In biological, the mutant vector means a sudden change in the gene characteristics of a chromosome [2]. This has proposed for a long time and it is widely accepted. To evolutionary computing, mutation is a perturb factor, a parent vector with a mutation operation generates an offspring, the mutant vector is also called donor vector.

Classical method selects r1,r2,r3,r4,r5,r6,r7, and are mutually exclusive integers from the range [1,NP][6] [8] [9]. For each mutant vector these integers would generate again. However, this method declines performance of DE. As these 7 parameters are mutually exclusive, it decreases the diversity of the perturb vector. The new method ignores the restriction that selects integers mutually exclusive, same integers can appear repeatedly in one mutant vector.

Fig1 illustrates the development of the new method. Use rand/2 as example, if x_{r1},x_{r2},x_{r3},x_{r4},x_{r5}, are all different,x_{r1},$F(x^G_{r2} - x^G_{r3})$,and $F(x^G_{r4} - x^G_{r5})$, are not zero. But this restriction is unreasonable. Restrict integers mutually exclusive predefined a scope to mutant the population and the new scope is smaller than the search space.

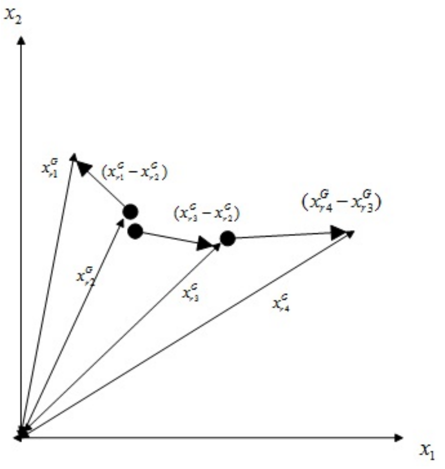

Fig. 1. Illustrating DE mutant vector scheme in 2-D parametric space. The new method increases the potential diversity of the population, $F(x^G_{r2} - x^G_{r3})$,is the increased choice of the Difference Vector.

In biological evolution, each individual would influence the evolution of the whole group. Bases on this theory, previous DE algorithms select all integers mutually exclusive. However, this approach is one-sided understanding of Darwinian principle. At biosphere, new offspring influenced by other individuals, but the number of other individuals is undecided. As all factors selects from the parent population which means new mutation generates by deformation and combination of parent individual. Some genes are called recessive genes would not certainly show their influence. Fixing the number of differences decreases the diversity of population individuals. So in this paper we select r1, r2, r3, r4, r5, r6, r7 randomly in [1, NP], even all of them are same is acceptable, but remember the distinct individual should different to the base vector , this restrict ensures the mutant vector would not be same to its parent -which is useless in the biological evolution process.

This simplified strategy not only simplifies the mutant vector generate process but also improves the performance of DE because it enriches the potential perturb diversity.

3 Experiment Result

3.1 Extend Benchmark Functions and Parameter Setting

Ferrante Neri and E. Mezura-Montes [1] [7] have explored the performance of some mutate vectors. It suggests that no one vector can achieve best result to all problems [7]. But this result is artless and sketchy, the experiment by using only 13 benchmarks is not sufficient as many new benchmarks are given now. This paper use comprehensive benchmarks to experiment different mutate vectors. 23 benchmarks are used to experiment various vectors. f1-f6 are unimodal functions, f7-f13 are multimodal functions with many local minima, f14-f20 are multimodal functions with a few local minima, especially f18-f20 are multimodal functions with deceiving, above functions are seen in [8]. f21-f23 are rotated functions. These rotated functions are generate by f7,f10,f11 multiply an orthogonal matrix.. As DE does not perform good on noise problem mentioned, in this paper no noise problems is discussed.

In the traditional sense, the mutation scale factor F, the crossover constant CR and population size NP is control parameters. The effects of them are well studied. To ensure all DE performance are in the same conditions, this paper set NP=100, dimension=30 as constant value, F=0.5 and CR=0.9. Iteration number of each function is given in TABLE1and TABLE 2.

3.2 Performance Comparison of Various Difference Vectors

This part we compare the result between DE with various differential operator. As cur-to-best/2 and rand-to-best/2 does not achieve good result and restricted by paper's length, we does not list performance of these two vectors in Table2. With 6 classical vectors and expanded 5 vectors, 11 vectors are listed in Table1 -Table4.

The result of various benchmarks shows that best/2 is best to handle unimodal functions and rand/1 is best to handle multimodal functions with many local minima. To unimodal functions with a few local minima, best/2 is weak in the vector family.

Here we give some details of these vectors to different benchmarks:

To f1-f6 unimodal functions, best/2 achieves best result, following is best-to-cur/1, rand/1, rand-to-best/1. Best/1 is worst.

Table 1. Experiment Result of Rand/1,Rand/2,Rand/3,Best/1,Best/2 and Best/3 over 50 Independent Runs

bench mark	gener ation		rand/1	rand/2	rand/3	best/1	best/2	best/3
f1	1500	Mean	2.71e-19	1.30e-03	6.20e+03	1.89e+03	**5.56e-49**	3.55e-06
		Std.Dev	1.79e-19	5.36e-4	1.05e+03	9.29e+02	**9.11e-49**	2.42e-06
f2	2000	Mean	4.18e-13	2.53e-02	5.69e+01	1.08e+01	**3.53e-34**	8.67e-04
		Std.Dev	2.59e-13	1.16e-02	4.91e+00	2.79e+00	**4.11e-34**	6.26e-04
f3	5000	Mean	1.74e-16	9.78e+00	1.94e+04	5.271e+03	**8.72e-49**	1.58e-01
		Std.Dev	1.25e-016	5.35e+00	3.84e+03	1.80e+003	**1.52e-48**	1.01e-01
f4	5000	Mean	5.74e-001	1.28e-03	4.09e+01	2.91e+001	**1.58e-07**	4.84e-04
		Std.Dev	1.245e+00	4.65e-04	3.82e+00	4.22e+000	**2.69e-07**	2.96e-04
f6	1500	Mean	0	0	6.12e+03	2.02e+003	0	0
		Std.Dev	0	0	9.12e+02	6.29e+002	0	0
f7	3000	Mean	1.55e+002	1.65e+02	2.82e+02	2.00e+002	**3.72e+01**	1.94e+02
		Std.Dev	1.063e+01	1.12e+01	1.16e+01	8.66e+00	1.28e+01	1.65e+01
f10	1500	Mean	**1.24e-010**	1.31e-02	1.41e+01	9.14e+000	6.07e-01	9.61e-04
		Std.Dev	**4.91e-011**	3.36e-03	6.33e-001	1.79e+000	7.59e-01	3.37e-04
f11	2000	Mean	**0**	2.77e-02	3.99e+01	2.05e+001	9.22e-03	5.34e-02
		Std.Dev	**0**	9.53e-02	6.18e+00	1.01e+001	1.13e-02	1.35e-01
f12	1500	Mean	**2.10e-020**	1.56e-03	3.24e+12	6.01e+010	1.35e-01	1.11e-03
		Std.Dev	**2.34e-020**	1.71e-03	2.11e+12	1.29e+011	3.05e-01	3.74e-03
f13	1500	Mean	**1.62e-019**	4.03e-03	1.47e+13	7.27e+011	6.33e-02	4.92e-05
		Std.Dev	**1.32e-019**	3.04e-03	5.69e+12	8.72e+011	2.68e-01	7.10e-05
f21	3000	Mean	1.77e+002	1.76e+02	1.78e+02	**8.32e+001**	**6.49e+01**	1.72e+02
		Std.Dev	1.43e+001	1.65e+01	1.32e+01	**1.55e+001**	**1.49e+01**	2.19e+01
f22	1500	Mean	**1.29e-006**	8.62e-03	2.04e+01	2.01e+001	4.20e-01	2.09e+01
		Std.Dev	**3.94e-006**	2.41e-03	6.60e-001	7.72e-002	6.56e-01	7.64e-02
f23	2000	Mean	**0**	4.95e-02	9.27e+001	1.59e+001	1.13e-02	1.99e-02
		Std.Dev	**0**	1.51e-01	2.32e+01	7.85e+00	1.09e-02	6.69e-02

To f7-f13 multimodal functions with many local minima, rand/1 achieves best result, following is ran-to-best/1, cur-to-best/1, best/3. best/2 is worse and other vectors achieve little optimize. f7 is special as best/2 achieves best and next is best/3.

To f14-f20 multimodal functions with a few local minima, in f14-f17, all vectors achieve similar result, in f18-f20, best/2 and best-to-cur/1 achieve best result, next is rand/1, others perform weak.

To observe rotated benchmarks, the sequence of these vectors does not change, but their convergence ability is weak. Specially, in f23 best/1 does not achieve best result any more, but cur-to-best is outstanding.

Moreover, best-to-cur/1 achieves similar result to best/2 but perform weak on multimodal functions, it is interesting that its brother vector cur-to-best/1 perform in contrary. To rand-to-cur/1, the brother of best/2, perform worse than best2 but still good in the whole family.

Table 2. Experiment Result of Cur-to-best/1,Bets-to-cur/2,Rand-to-best/1,Best-to-rand/1 and Rand-to-cur/1 over 50 Independent Runs

benchmark	generation		Cur-to -best/1	Best-to -cur/1	Rand-to -best/1	Best-to -rand/1	Rand-to -cur/1
f1	1500	Mean	1.17e-08	7.04e-044	3.19e-009	5.89e-003	2.10e-001
		Std.Dev	1.55e-008	7.41e-044	2.96e-009	2.59e-003	1.73e-001
f2	2000	Mean	1.12e-005	3.11e-031	3.79e-006	1.45e-001	1.10e+000
		Std.Dev	6.31e-006	2.58e-031	1.90e-006	1.05e-001	4.07e-001
f3	5000	Mean	6.61e-009	1.45e-048	1.71e-006	5.67e-001	6.91e+001
		Std.Dev	7.1e-009	3.26e-048	1.48e-006	4.74e-001	3.35e+001
f4	5000	Mean	6.02e-004	4.34e-007	1.12e-007	9.59e-002	8.52e-002
		Std.Dev	8.28e-004	4.87e-007	2.29e-007	4.13e-001	2.37e-002
f6	1500	Mean	0	0	0	0	2.50e-001
		Std.Dev	0	0	0	0	4.33e-001
f7	3000	Mean	1.55e+002	2.34e+001	1.75e+002	1.98e+002	2.11e+002
		Std.Dev	7.49e+000	7.31e+000	1.61e+001	1.25e+001	1.05e+001
f10	1500	Mean	3.74e-005	7.95e-001	1.94e-005	3.10e-002	3.21e-001
		Std.Dev	1.05e-005	9.22e-001	6.16e-006	1.01e-002	1.14e-001
f11	2000	Mean	3.57e-003	9.48e-003	1.48e-003	1.41e-002	1.15e-001
		Std.Dev	4.64e-003	7.71e-003	3.60e-003	5.45e-002	2.05e-001
f12	1500	Mean	4.50e-009	1.19e-001	1.65e-009	6.39e-003	1.11e-03
		Std.Dev	6.25e-009	1.95e-001	1.31e-009	8.57e-003	3.29e-001
f13	1500	Mean	5.49e-004	3.75e-001	5.33e-009	1.49e-002	4.46e-001
		Std.Dev	2.39e-003	9.45e-001	7.19e-009	9.52e-003	2.25e-001
f21	3000	Mean	1.79e+002	1.76e+002	1.63e+002	1.75e+002	1.72e+02
		Std.Dev	1.93e+001	2.08e+001	2.32e+001	2.13e+001	1.80e+001
f22	1500	Mean	2.09e+001	1.14e+000	2.09e+001	2.09e+001	3.69e-001
		Std.Dev	5.24e-002	9.41e-001	7.20e-002	4.32e-002	1.52e-001
f23	2000	Mean	4.92e-003	1.17e-002	3.08e-003	2.36e-002	1.06e-001
		Std.Dev	7.69e-003	1.29e-002	5.82e-003	1.02e-001	1.74e-001

Bases on the experiment we could gain some useful concludes:

Increase more than two difference parameter would decrease the performance of the perturb vector. To bets/n, best/2 is best and best/1 is weak, if n is bigger than two, increase difference parameter would decrease the performance of perturb vector. To rand/n, rand/1 performs weak inmultimodal function and rand/2 get best result. Because best/1 with only one difference parameter it is constrained by limited search ability, with more difference parameter, although it is outstanding in search ability but performs weak in convergence ability. In this theory rand/1 is easy to entrap into local minima.

Rand/1 performs best to unimodal functions and best/2 is best to multimodal functions. Specially, vectors with parameter perform excellent in the vector family.

cur-to-best/1 and this series mutant vectors performs not best in mutate family. But these vectors present exceptional features that same parameter in different position lead to vary results. This feature should further study.

3.3 Performance of New Method Generates Mutant Vector

Restricted by the length, this paper presents part of the result comparison between new method and old method in Table 5 and Table 6. According to the experiment result, the new method optimizes the algorithm steadily. The new method optimizes unimodal functions, multimodal functions with many local minima, multimodal functions with deceiving and rotated functions well.

The new method achieves better performance in unimodal functions. To multimodal functions with many local minima, best/2 use the new method does not optimize the performance as obvious as unimodal functions done, but the rand/1 and cur-to-best/1 achieve outstanding optimized results. To multimodal functions with a few local minima, all the mutant vectors achieves same results, the new method achieves same-level result, too. To rotated benchmarks, many DE achieve weak results, but the new method still achieves obvious optimized results.

4 Conclusion

Under the standard DE framework, considerable research has put forward many mutant vectors. This paper proposed the linear relation between the algorithm performance and the differential operator. According to experiment result, increasing the differential operator would not certainly optimize the performance of DE. In contrary, adds more than two difference parameters would decrease the convergence ability. Some of the mutant vector like cur-to-best/1 and rand-to-best did not perform best in the mutate family, but it is exceptional and worth further study because in these mutate vectors parameters at different position lead to different performance.

Moreover, we use rand/1, rand/2 and best/2, three best mutant vectors to test the new method. The new method is more effective and achieves better result than old method. The new method could optimize the performance of DE in general. DE with three different mutant vectors all achieve optimized result.

References

1. Neri, F., Tirronen: Recent Advances in Differential Evolution: A Survey and Experimental Analysis. Artif. Intell. Rev. 33(1), 61–106 (2010)
2. Das, S., Suganthan, P.N.: Differential evolution: A survey of the state-of-the-art. IEEE Trans. on Evolutionary Computation (2011), doi:10.1109/TEVC.2010.2059031
3. Caponio, A., Neri, F.: Differential Evolution with Noise Analyzer. In: Giacobini, M., Brabazon, A., Cagnoni, S., Di Caro, G.A., Ekárt, A., Esparcia-Alcázar, A.I., Farooq, M., Fink, A., Machado, P. (eds.) EvoWorkshops 2009. LNCS, vol. 5484, pp. 715–724. Springer, Heidelberg (2009)
4. Liu, B., Zhang, X., Ma, H.: Hybrid differential evolution for noisy optimization. In: Proc. IEEE Congr. Evol. Comput., vol. 2, pp. 1691–1698 (2009)

5. Wang, Y., Cai, Z., Zhang, Q.: Differential evolution with composite trial vector generation strategies and control parameters. IEEE Trans. 15(1), 55–66 (2011)
6. Qin, A.K., Huang, V.L., Suganthan, P.N.: Differential evolution algorithm with strategy adaptation for global numerical optimization. IEEE Trans. Evol. Comput. 13(2), 398–417 (2009)
7. Mezura-Montes, E., Vel azquez-Reyes, J., Coello Coello, C.A.: A comparative study of differential evolution variants for global optimization. In: Proc. Genet. Evol. Comput., pp. 485–492 (2006)
8. Brest, J., Greiner, S., Boskovic, B., Mernik, M., Zumer, V.: Self-adapting control parameters in differential evolution: A comparative study on numerical benchmark problems. IEEE Trans. Evolut. Comput. 10(6), 646–657 (2006)
9. Epitropakis, G., Tasoulis, D.K., Pavlidis, N.G., Plagianakos, V.P., Vrahatis, M.N.: Enhancing differential evolution utilizing proximity-based mutation operators. IEEE Trans. Evol. Comput., 99–119 (2011)