Communications on Applied Electronics (CAE) – ISSN : 2394-4714
Foundation of Computer Science FCS, New York, USA
Volume 4– No.2, January 2016 – www.caeaccess.org

Differential Evolution with Alternating Strategies: A Novel Algorithm for Numeric Function Optimization

Mohammad Shafiul Alam
Ahsanullah University of Science and Technology
Dhaka-1208, Bangladesh

Md. Tawseef Alam
Ahsanullah University of Science and Technology
Dhaka-1208, Bangladesh

Farniba Khan
Ahsanullah University of Science and Technology
Dhaka-1208, Bangladesh

A. A. Fattah Islam
Ahsanullah University of Science and Technology
Dhaka-1208, Bangladesh

Md. Rasel Kabir
Ahsanullah University of Science and Technology
Dhaka-1208, Bangladesh

ABSTRACT
The Differential Evolution (DE) is a prominent meta-heuristic algorithm that has been successfully employed to numerous complex and diverse problems from the fields of mathematics, science and engineering. DE belongs to the evolutionary family of algorithms which is based on the Darwinian theory of natural selection and evolution. DE maintains a population of candidate solutions and uses the vector differences between randomly picked candidate solution vectors to produce new, improved solutions to advance its evolutionary optimization process, generation by generation. This paper introduces a novel DE-variant — the DE with Alternating Strategies (DE-AS) and evaluates its performance using a number of benchmark problems on numeric function optimization. DE-AS effectively combines the exploitative and explorative characteristics of five different DE-variants by randomly alternating and executing these DE-variants in a single algorithm. The experimental results indicate that DE-AS can perform better than many other existing DE-variants on most of the benchmark functions, in terms of both final solution quality and convergence speed.

Keywords
Evolutionary algorithm, differential evolution, exploitation and exploration, numeric function optimization.

1. INTRODUCTION
During the last few decades, several stochastic, heuristic and meta-heuristic algorithms have emerged which showed significant success to deal with many complex scientific, mathematical and engineering problems, such as continuous optimization [1]–[3], combinatorial optimization [4], multi-objective optimization [5], industrial process control [6], engineering design [7], design of digital IIR filters [8], PID controllers [9], machine learning [10] and so on [11]. The Differential Evolution (DE) is a recently introduced meta-heuristic algorithm which belongs to the family of Evolutionary Algorithms (EAs) and is successfully employed to many complex search and optimization problems.

Like most other population based meta-heuristic algorithms, the DE is usually resilient and robust against premature convergence and fitness stagnation. The reason is that the population of individuals (i.e., candidate solutions) can usually preserve sufficient amount of diversity and explorative search capability, which is necessary to continue the search space explorations around the locally optimal points without being trapped anywhere around them. However, the opposite scenario has also been observed (e.g., [12]–[14]) when the pool of candidate solutions completely lost their diversity and the optimization procedure got stuck around some locally optimal points, which is known as 'premature convergence' in the literature of EAs. The reason behind premature convergence is more exploitation at the cost of reduced explorations. But increasing explorations at the expense of decreased exploitations might not be the solution, because this usually leads to slow and unacceptable convergence speed. This is why a balance between the explorative and exploitative operations is desired for good results and satisfactory convergence speed.

There exist a number of variations of the differential evolution algorithm, as briefly presented in the section 3. However, all of them are biased, either towards more exploration or towards more exploitation. This paper introduces a novel DE-variant — the Differential Evolution with Alternating Strategies (DE-AS) that involves five different DE-variants, three exploitative and two explorative, to bring a balance between explorations and exploitations. DE-AS deploys the two explorative DE variants during the early phase of its execution when more exploration is desired, followed by the three exploitative DE-variants during the late generations when exploitation and fine-tuning are necessary. Such an approach tries to ensure a good balance between exploitations and explorations in order to achieve improved results with satisfactory convergence speed.

The rest of this paper is organized as follows. Section 2 explains the numeric function optimization problem. Section 3 briefly describes the DE algorithm, with its five different strategies. Section 4 introduces the proposed DE-AS algorithm. Section 5 presents the parameter settings and experimental setup of all the different DE-variants and makes a comparison of their performance on ten different complex, high dimensional unimodal and multimodal functions. Finally, section 6 concludes the paper with a brief discussion on DE-AS, followed by some suggestions as directions for further research with DE-AS.

2. FUNCTION OPTIMIZATION PROBLEM
Many real world problems can be formulated as a function optimization problem of the parameters that takes values from some continuous domain, i.e., the continuous function optimization problems. A continuous function optimization problem can be formalized as follows.

Communications on Applied Electronics (CAE) – ISSN : 2394-4714
Foundation of Computer Science FCS, New York, USA
Volume 4– No.2, January 2016 – www.caeaccess.org

$$\underset{\boldsymbol{x}}{\text{Minimize}}\ f(\boldsymbol{x});\ \text{subject to}\ \boldsymbol{x} \in S$$

The objective is to find a vector $\boldsymbol{x}_{min}$ such that $f(\boldsymbol{x}_{min}) \leq f(\boldsymbol{x})$ for all $\boldsymbol{x} \in S$. Here, the search space S is a bounded subset of $\boldsymbol{R}^n$ and the objective function to be optimized is $f(.)$, which is an n-dimensional real valued function. The objective is to optimize $f(.)$ over its parameter $\boldsymbol{x}$. Each element x_i of the vector $\boldsymbol{x}$ is a real valued variable, i.e., $\boldsymbol{x} = [x_1, x_2, \ldots, x_n]^T$

The task of numeric function optimization is generally referred with many different names, such as real parameter optimization, continuous optimization, or simply, numeric optimization. However, all of them actually refer to the general task of finding a solution across a real valued, (usually) multi-dimensional search space such that the solution gives the best value, i.e., minimum or maximum value, of an objective function, depending on whether it is a minimization or maximization task. This solution should have not only the best objective function value around its local neighborhood, but also the best objective value over all the feasible solutions across the entire search space.

3. DIFFERENTIAL EVOLUTION

Like other population based meta-heuristic algorithms, DE maintains a population of N vectors, each one representing a candidate solution which is an n-dimensional real valued vector $\boldsymbol{x}_i = [x_{i1}, x_{i2}, \ldots, x_{in}] \in S$. Here, $S \subset \boldsymbol{R}^n$ and $i = 1, 2, \ldots, N$. This population of candidate solutions gradually evolves, generation by generation, by the DE operators of mutation and crossover, as explained later in this section. During every generation, DE uses its mutation operation on every individual vector $\boldsymbol{x}_{i,G}$ (also called the target vector) to produce the mutated vector $\boldsymbol{v}_{i,G}$, then crossover operation on the target and mutated vectors to produce the trial vector $\boldsymbol{u}_{i,G}$, followed by the selection operation on the target and trial vectors to select the better one of them for the next generation.

Mutation Operation:

For each target vector $\boldsymbol{x}_{i,G}$ (i.e., individual candidate solution of current generation G), an associated mutated vector $\boldsymbol{v}_{i,G} = [v_{i1,G}, v_{i2,G}, \ldots, v_{in,G}]$ is produced by any of these strategies.

Strategy DE/rand/1:

$$\boldsymbol{v}_{i,G} = \boldsymbol{x}_{r_1,G} + F \,.\, (\boldsymbol{x}_{r_2,G} - \boldsymbol{x}_{r_3,G})$$

Strategy DE/rand/2:

$$\boldsymbol{v}_{i,G} = \boldsymbol{x}_{r_1,G} + F \,.\, (\boldsymbol{x}_{r_2,G} - \boldsymbol{x}_{r_3,G}) + F \,.\, (\boldsymbol{x}_{r_4,G} - \boldsymbol{x}_{r_5,G})$$

Strategy DE/best/1:

$$\boldsymbol{v}_{i,G} = \boldsymbol{x}_{best,G} + F \,.\, (\boldsymbol{x}_{r_1,G} - \boldsymbol{x}_{r_2,G})$$

Strategy DE/best/2:

$$\boldsymbol{v}_{i,G} = \boldsymbol{x}_{best,G} + F \,.\, (\boldsymbol{x}_{r_1,G} - \boldsymbol{x}_{r_2,G}) + F \,.\, (\boldsymbol{x}_{r_3,G} - \boldsymbol{x}_{r_4,G})$$

Strategy DE/current-to-best/1:

$$\boldsymbol{v}_{i,G} = \boldsymbol{x}_{i,G} + F \,.\, (\boldsymbol{x}_{best,G} - \boldsymbol{x}_{i,G}) + F \,.\, (\boldsymbol{x}_{r_1,G} - \boldsymbol{x}_{r_2,G})$$

In these strategies, the indices r_1, r_2, r_3, r_4, r_5 are random integers which are different from the current index i and also, different from each other. They are generated uniformly at random from $[1, N]$. The vector $\boldsymbol{x}_{best,G}$ is the best individual of the current generation G which has the highest fitness value. The parameter F of the algorithm takes values from (0, 1+) and acts as a scaling factor for the vector differences.

Crossover operation

DE employs the crossover operation after applying the mutation operation on each individual. The crossover is done between each pair of target vector $\boldsymbol{x}_{i,G}$ and the corresponding mutated vector $\boldsymbol{v}_{i,G}$ to produce a new trial solution vector $\boldsymbol{u}_{i,G} = [u_{i1,G}, u_{i2,G}, \ldots, u_{in,G}]$ using the following method.

for $j = 1, 2, \ldots, n$

$$u_{ij,G} = \begin{cases} v_{ij,G}, & \text{if } \left(rand_j[0,1] \leq CR \text{ or } j = j_{rand}\right) \\ x_{ij,G}, & \text{otherwise} \end{cases}$$

The CR above is the crossover rate, which is a user specified parameter of DE in the range of [0,1). The random index j_{rand} is randomly picked from $[1, n]$ to ensure that the trial vector $\boldsymbol{u}_{i,G}$ is different from the original vector $\boldsymbol{x}_{i,G}$ (i.e., current target vector) by at least one parameter.

Selection operation:

The selection procedure of DE is a simple, greedy fitness based selection scheme between each pair of target vector and its associated trial vector. The fitness value (i.e., reciprocal of the function value for a function minimization problem) of each trial vector $\boldsymbol{u}_{i,G}$ is computed and compared with the fitness value of the corresponding target vector $\boldsymbol{x}_{i,G}$. If the trial vector $\boldsymbol{u}_{i,G}$ has smaller or equal function value (for a minimization problem) than the target vector $\boldsymbol{x}_{i,G}$, then the trial vector $\boldsymbol{u}_{i,G}$ will replace the original vector $\boldsymbol{x}_{i,G}$ in the population and $\boldsymbol{x}_{i,G}$ will be deleted from the population. Otherwise, the original target vector $\boldsymbol{x}_{i,G}$ will be kept and the trial vector $\boldsymbol{u}_{i,G}$ will be deleted.

4. PROPOSED ALGORITHM: DE WITH ALTERNATING STRATEGIES (DE-AS)

This section introduces the novel DE-variant — Differential Evolution with Alternating Strategies (DE-AS). The motivation behind DE-AS is to balance between explorations and exploitations to avoid premature convergence and to locate the neighborhood of the global minimum. To achieve this objective, DE-AS employs all the existing five DE-variants, as mentioned in the previous section, during its execution. In the following paragraph, a brief description of the proposed DE-AS algorithm is presented.

The main approach of DE-AS is to combine all the five different DE-variants, discussed in the previous section 3, and to execute them randomly during its different generations. The DE-variant that is to be executed in the current generation is picked at random, either from the explorative DE-variants or from the exploitative DE-variants, depending on the current explorative/exploitative need. Among the five DE-variants, the first two (DE/rand/1 and DE/rand/2) are mainly explorative, while the remaining three (DE/best/1, DE/best/2 and DE/current-to-best/1) are exploitative. The need for explorations is usually high during the initial generations. This is why DE-AS randomly selects either of its two explorative variants (i.e., DE/rand/1 and DE/rand/2) during the first two-thirds of its runtime. For the final one-third of its runtime, DE-AS randomly selects any of its three exploitative variants (i.e., DE/best/1, DE/best/2 and DE/current-to-best/1).

To further clarify the DE-AS algorithm, a concrete example is presented here. Suppose the predefined specified runtime for a problem is set to 1500 generations. For this problem, DE-AS

randomly picks either of DE/rand/1 or DE/rand/2 during the first 1000 generations (i.e., two-thirds of its runtime). For the last 500 generations (i.e., one-third runtime), DE-AS randomly picks either of DE/best/1, DE/best/2 and DE/current-to-best/1. The random selection of a DE-variant is done regularly, during every generation. The randomization during every generation ensures a balanced and proper mix of explorations and exploitations across the total generations of DE-AS.

5. EXPERIMENTAL STUDIES

To evaluate the performance of DE-AS and to compare it with other DE-variants, this paper uses a standard benchmark suite of numeric function optimization problems, consisting of four unimodal and six multimodal, high dimensional functions [1]–[3], [16]. Table 1 presents a brief overview on each of these benchmark functions. More details on each function can be found in [1]. All these benchmark functions are complex and high dimensional functions. For the six multimodal functions $f_5 - f_{10}$, the search algorithm must possess both exploitative and explorative characteristics so that it can explore the locally optimal points without being trapped around any of them. Some of the multimodal functions can have hundreds of local minima, even when the dimensionality is just two or three. Their number of local optima increases exponentially with the number of their dimensions, which makes their optimization extremely difficult. For example, the Schwefel 2.26 function f_5 has exponentially many locally minimal points which are very deep and very far from the single global minimum. The Ackley function f_7 has one narrow global minimum basin, but with exponentially many minor local minima. The Griewank function f_8 has a component creating linkage among the variables, which complicates the search by perturbing any subset of the variables. Any technique that tries to optimize each variable separately without considering the others will fail for this function. Since the number of local minima increases exponentially with the number of dimensions, the high dimensionality (i.e., $D = 30$) of all the multimodal functions make them extremely difficult to be optimized by any algorithm. Because of the existence of numerous locally minimal points, the search algorithm may easily get trapped around any of them, missing the single global minimum.

The experimental results of DE-AS and two other existing DE-variants (i.e., the DE/best/1 and DE/rand/1) are presented in Table 2. The common parameters of all three algorithms are the population size N, which is set to 100. The no. of maximum generations is different for the different functions, as shown in Table 2. For the unimodal functions, which are considered relatively easier than the multimodal functions, the runtime is set to 1000 generations. For the multimodal functions, the runtime is set to various generations, from 1500 to 9000 generations, based on the complexity of the functions. The values in Table 2 indicate the error (i.e., the difference between global minimum and the minimum possible function value found by the DE-variants during the final generation). The important observations on the results are summarized in the following few points.

- Out of the 10 benchmark functions $f_1 - f_{10}$, DE-AS performed best on as many as five functions, while DE/best/1 performed best on four. On one function (f_{10}), all three algorithms performed equally well.
- In comparison to DE/rand/1, the performance of DE-AS is almost always better (9 out of 10 functions) or similar (only on one function, i.e., f_{10})
- For two functions — f_4 and f_7 all the three algorithms fail to locate the global minimum and show signs of premature convergence. However, DE-AS shows somewhat better results on both these functions.
- An overall performance of the algorithms can be compared based on their mean absolute error (MAE) values over the unimodal functions $f_1 - f_4$ and multimodal functions $f_5 - f_{10}$. The MAE of DE-AS is the smallest on both the function families, which indicates that DE-AS is overall best for both unimodal and multimodal functions.
- From the viewpoint of exploration vs. exploitation phenomenon, the DE/best/1 variant is the most exploitative one. This becomes apparent from its extremely low error values for the relatively easier unimodal functions (i.e., f_1, f_2) and some multimodal functions (f_6 and f_8). This indicates DE/best/1 might be more suitable than DE-AS for the task of fine tuning and pinpointing the global minimum for easier functions. However, DE-AS is more explorative and might be the better choice for more complex multimodal functions.

To summarize the experimental findings, DE-AS is more explorative than DE/best/1, and hence it shows best overall performance (i.e., smaller MAE) over both unimodal and multimodal functions. DE-AS consistently outperforms the DE/rand/1 variant, showing superior results and higher convergence speed for all the functions. However, the DE/best/1 variant is found to be the most exploitative one and shows best results by fine-tuning, but only on a few relatively easier functions.

Table 1: The benchmark functions used in experimental studies. Here, *D*: dimensionality of the function, *S*: search space, f_{min}: function value at the global minimum, *C*: function characteristics with the following values — *U*: Unimodal *M*: Multimodal, *S*: Separable, *N*: Non-separable.

No	Function	*C*	*D*	*S*	f_{min}
f_1	Sphere	*US*	30	$[-100, 100]^D$	0
f_2	Schwefel 1.2	*UN*	30	$[-100, 100]^D$	0
f_3	Schwefel 2.21	*US*	30	$[-100, 100]^D$	0
f_4	Rosenbrock	*UN*	30	$[-30, 30]^D$	0

f_5	Schwefel 2.26	*MS*	30	$[-500, 500]^D$	–12569.5
f_6	Rastrigin	*MS*	30	$[-5.12, 5.12]^D$	0
f_7	Ackley	*MN*	30	$[-32, 32]^D$	0
f_8	Griewank	*MN*	30	$[-600, 600]^D$	0
f_9	Penalized	*MN*	30	$[-50, 50]^D$	0
f_{10}	Penalized2	*MN*	30	$[-50, 50]^D$	0

Table 2: Performance comparison of the proposed DE-AS and two other DE variants on the benchmark functions. The values indicate the error (i.e., difference between global minimum and the minimum found function value by the DE variants) on the different functions. The best performance (i.e., minimum error) on each function is marked with boldface font.

No	f_{min}	Generations	DE/best/1	DE/rand/1	DE-AS	Best Performance by
f_1	0	1000	**8.57e–37**	1.39e–09	6.28e–14	DE/best/1
f_2	0	1000	**5.97e–34**	1.87e–07	1.31e–11	DE/best/1
f_3	0	1000	15.38	1.10	**6.60e–01**	DE-AS
f_4	0	1000	24.07	21.87	**19.51**	DE-AS
f_5	–12569.5	9000	7.69e+03	1.34e–02	**4.15e–04**	DE-AS
f_6	0	5000	**1.48e–201**	2.31e–64	4.37e–86	DE/best/1
f_7	0	1500	20.01	20.02	**19.88**	DE-AS
f_8	0	2000	**5.78e–82**	6.48e–27	8.60e–35	DE/best/1
f_9	0	1500	7.39e–14	7.40e–14	**7.33e–14**	DE-AS
f_{10}	0	1500	2.61e–03	2.61e–03	2.61e–03	All Similar
Mean Absolute Error (Unimodal Functions $f_1 - f_4$)			9.86e+00	5.74e+00	**5.04e+00**	DE-AS
Mean Absolute Error (Multimodal Functions $f_5 - f_{10}$)			1.29e+03	3.34e+00	**3.17e+00**	DE-AS

6. CONCLUSION

This paper presents a novel variant of the standard differential evolution algorithm — the Differential Evolution with Alternating Strategies (DE-AS) and evaluates its performance on a standard suite of benchmark problems on numeric function optimization. The experimental results indicate that DE-AS can achieve very good results, outperforming some other existing DE-variants on most of the functions. There might be several possible future research directions based on this study. Firstly, some explorative meta-heuristic algorithms can reliably locate the neighborhood of the global minimum for more complex functions like f_4 and f_7. Hybridizing them with DE-AS might make it more robust and resilient against premature convergence and fitness stagnation. Secondly, the DE-variants should be compared on easier low dimensional functions to gain further insights on their strength and weakness. This might help to hybridize them more effectively for improved results and better resilience against premature convergence. Thirdly, the possibility of improving the final solution quality might be investigated by using an efficient local searcher after the execution of DE-AS is over. This may make its performance even better. Finally, DE-AS is applied only on the continuous functions. It would be interesting to study how well it can perform on many other existing problems, especially the discrete and real world problems.

7. REFERENCES

[1] D. Karaboga and B. Basturk, On the performance of artificial bee colony (ABC) algorithm, *Applied Soft Computing* **8** (1) (2008) 687–697.

[2] D. Karaboga, An idea based on honey bee swarm for numerical optimization, Erciyes University, Kayseri, Turkey, *Technical Report-TR06*, 2005.

[3] X. Yao, Y. Liu and G. Lin, "Evolutionary programming made faster", *IEEE Transactions on Evolutionary Computation* **3** (2) (1999) 82–102.

[4] S. Sobti and P. Singla, Solving travelling salesman problem using bee colony based approach, *Int. Journal of Engg. Research and Technology* **2** (6) (2013) 186–189.

[5] K. Naidu, H. Mokhlis and A.H.A. Bakar, Multiobjective optimization using weighted sum Artificial Bee Colony algorithm for Load Freq. Control, *Int. Jour. of Electrical Power and Energy Systems* **55** (2) (2014) 657–667.

[6] R. Mukherjee, D. Goswami and S. Chakraborty, Parametric optimization of Nd:YAG laser beam machining process using artificial bee colony algorithm, *Journal of Industrial Engineering*, vol. 2013, Article ID 570250, 15 pages, 2013. DOI: 10.1155/2013/570250.

[7] H. Garg, Solving structural engineering design optimization problems using an artificial bee colony algorithm, *Journal of Industrial and Management Optimization*, **10** (3) (2014) 777–794.

[8] Z. Zhao, D. Yin and Y. Jiang, Improved bee colony algorithm based on knowledge strategy for digital filter design, *International Journal of Computer Applications*, **47** (2) (2013) 241–248.

[9] A. Mishra, A. Khanna, N. Singh and V. Mishra, Speed control of DC motor using bee colony optimization, *Universal Journal of Electrical and Electronic Engineering* **1** (3) (2013) 68–75.

[10] A. Karegowda and M. Darshan, Optimizing feed forward neural network connection weights using artificial bee colony algorithm, *International Journal of Advanced Research in Computer Science and Software Engineering* **3** (7) (2013) 452–454.

[11] A. Bolaji, A. Khader, M. Betar and M. Awadallah, Bee colony algorithm, its variants and applications: A survey, *Journal of Theoretical and Applied Technology* **47** (2) (2013) 434–459.

[12] T. Park and K. R. Ryu, A Dual population genetic algorithm for adaptive diversity control, *IEEE Trans. Evolutionary Computation* **14** (6) (2010) 865–884.

[13] R. K. Ursem, Diversity guided evolutionary algorithms, in *P*roc. 7*th Int. Conf. Parallel Problem Solving from Nature* (*PPSN*), 2002, pp. 462–474.

[14] J. Lampinen and I. Zelinka, On stagnation of the differential evolution algorithm, in *Proc. 6th Int. Mendel Conf. Soft Computing*, Brno, Czech Republic, 2000, pp. 76–83.

[15] T. Bäck and H.–P. Schwefel, "An overview of evolutionary algorithms for parameter optimization", *Evolutionary Computation* **1** (1) (1993) 1–23.

[16] W. Lee and W. Cai, A novel artificial bee colony algorithm with diversity strategy, in *Proc. 7th Int. Conf. Natural Computation*, 2011, pp. 1441–1444.