A Study on Self-adaptation in the Evolutionary Strategy Algorithm

Noureddine Boukhari, Fatima Debbat, Nicolas Monmarché, Mohamed Slimane

▶ **To cite this version:**

Noureddine Boukhari, Fatima Debbat, Nicolas Monmarché, Mohamed Slimane. A Study on Self-adaptation in the Evolutionary Strategy Algorithm. 6th IFIP International Conference on Computational Intelligence and Its Applications (CIIA), May 2018, Oran, Algeria. pp.150-160, 10.1007/978-3-319-89743-1_14 . hal-01913901

HAL Id: hal-01913901

https://inria.hal.science/hal-01913901v1

Submitted on 7 Nov 2018

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Distributed under a Creative Commons Attribution 4.0 International License

A Study on Self-adaptation in the Evolutionary Strategy Algorithm

Noureddine Boukhari[1], Fatima Debbat[2], Nicolas Monmarché[3] and Mohamed Slimane[4]

[1, 2] Department of Computer Science, Mascara University, Mascara, Algeria
[3,4] Laboratoire Informatique (EA6300), Université François Rabelais Tours, 64, avenue Jean Portalis, 37200 Tours, France

boukhari.noureddine@gmail.com, debbat_fati@yahoo.fr, nicolas.monmarche@univ-tours.fr, mohamed.slimane@univ-tours.fr

Abstract. Nature-inspired algorithms attract many researchers worldwide for solving the hardest optimization problems. One of the well-known members of this extensive family is the evolutionary strategy ES algorithm. To date, many variants of this algorithm have emerged for solving continuous as well as combinatorial problems. One of the more promising variants, a self-adaptive evolutionary algorithm, has recently been proposed that enables a self-adaptation of its control parameters. In this paper, we discuss and evaluate popular common and self-adaptive evolutionary strategy (ES) algorithms. In particular, we present an empirical comparison between three self-adaptive ES variants and common ES methods. In order to assure a fair comparison, we test the methods by using a number of well-known unimodal and multimodal, separable and non-separable, benchmark optimization problems for different dimensions and population size. The results of this experiments study were promising and have encouraged us to invest more efforts into developing in this direction.

Keywords: meta-heuristics, evolutionary algorithms, evolution strategy, parameter control, self-adaptation.

1 Introduction

The most successful methods in global optimization are based on stochastic components, which allow escaping from local optima and overcome premature stagnation. A famous class of global optimization methods is ES. They are exceptionally successful in continuous solution spaces. ES belong to the most famous evolutionary methods for black-box optimization, i.e., for optimization scenarios, where no functional expressions are explicitly given and no derivatives can be computed [1] [2]. ES imitate the biological principle of evolution [3] and can serve as an excellent introduction to learning and optimization. They are based on three main mechanisms oriented to the process of Darwinian evolution, which led to the development of all species. Evolutionary concepts are translated into algorithmic operators, i.e., recombination, mutation, and selection. This paper focuses on the self-adaptation in evolution strategies

(ES). Primarily, the self-adaptation has been gaining popularity due to the flexibility in adaptation to different fitness landscapes [8].
This method enables an implicit learning of mutation strengths in the real-valued search spaces. Self-adaptation bases on the mutation operator that modifies the problem variables using the strategy parameters to search the problem and parameter spaces simultaneously [8]. In line with this, the best values of problem variables and strategy parameters survive during the evolutionary process.

Evolutionary algorithms (EAs) are intrinsically dynamic, adaptive processes [21]. Therefore, setting the strategy parameters, controlling the behavior of these algorithms, fixed during the run is in contrast with an idea of evolution on which bases an evolutionary computation (EC). In line with this, the evolution of evolution has been developed by Rechengerg [10] and Schwefel [6], where the strategy parameters are put into a representation of individuals and undergo operations of the variation operators. As a result, the values of strategy parameters those modify the problem variables, which are the most adapted to the fitness landscape, as determined by the fitness function of the problem to be solved during the evolutionary search process. However, the process of the simultaneously evolution of strategy parameters together with the problem variables is also named a self-adaptation in EC.

2 Evolution Strategies

Evolution strategies derive inspiration from principles of biological evolution. We assume a population, P, of so-called individuals. Each individual consists of a solution or object parameter vector $x \in R n$ (the visible traits) and further endogenous parameters, s (the hidden traits), and an associated fitness value, f(x). In some cases the population contains only one individual. Individuals are also denoted as parents or offspring, depending on the context. In a generational procedure [4]. First, one or several parents are picked from the population (mating selection) and new off- spring are generated by duplication and recombination of these parents; then, the new offspring undergo mutation and become new members of the population; finally, Environmental selection reduces the population to its original size.

2.1 Pseudo code For Evolutionary Algorithms

Using these ideas a computer algorithm can be developed to analyze a problem and its data to achieve an optimal solution to that problem, as shown in Fig. 2. First, an initial population P(t) is generated randomly and evaluated. Second a mutation technique is applied to adjust the children to a new set P'(t). The fitness of those children is evaluated and then children are chosen from the set of parents P(t) and children P'(t) to form the new parent set P(t+1). The loop terminates if either the maximum number of iterations are reached or the desired solution is found. As outlined here by Castro: [17]

Pseudo code For Evolutionary Algorithms

```
Initialize P (t)
 Evaluate P (t)
 While not Terminate do
        P′(t) = mutate P(t)
        Evaluate P′(t)
        P(t+1) = P′(t)
        or best of {P′(t) U P(t)}
 t = t + 1
 Loop
```

3 Mutation And Parameter Control

Mutation introduces small, random and unbiased changes to an individual. These changes typically affect all variables. The average size of these changes depends on endogenous parameters that change over time. These parameters are also called control parameters, or endogenous strategy parameters, and define the notion of "small", for example via the step-size σ. In contrast, exogenous strategy parameters are fixed once and for all, for example parent number μ. Parameter control is not always directly inspired by biological evolution, but is an indispensable and central feature of evolution strategies [4]. Control parameters are encoded into chromosomes and undergo actions by the variation operators (e.g., crossover and mutation) using self-adaptive parameter control. The better values of parameter variables and control parameters have more chances to survive and reproduce their genetic material into the next generations. This phenomenon makes EAs more flexible and closer to natural evolution [5]. This feature was firstly introduced in ES by Schweffel [6]. Parameter control addresses only one side of the parameter setting, where the strategic parameters are changed during the run. In contrast, when the parameters are fixed during the run, an optimal parameter setting needs to be found by an algorithm's developer. Typically, these optimal parameters are searched during a tuning. In general, the taxonomy of parameter setting according to Eiben and Smith [7] is as illustrated in Fig. 2 Sample Heading (Third Level). Only two levels of headings should be numbered. Lower level headings remain unnumbered; they are formatted as run-in headings.

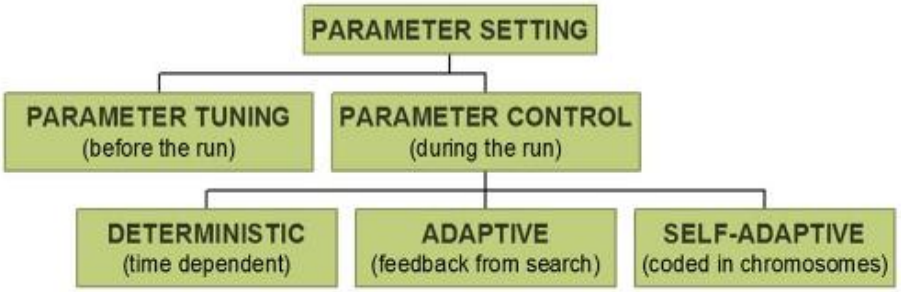

Figure 1 Parameter setting in CI algorithms

3.1 Deterministic parameter control

Deterministic parameter control [23]means that the parameters are adjusted according to a fixed time scheme, explicitly depending on the number of generations t. It may be useful to reduce the mutation strengths during the evolutionary search, in order to allow convergence of the population. There are many examples where parameters are controlled deterministically. Efren Mezura- Montes et al.[22] make use of deterministic parameter control for differential evolution based constraint handling.

3.2 Adaptive parameter control

Adaptive parameter control methods make use of rules the practitioner defines for the heuristic. A feedback from the search determines magnitude and direction of the parameter change. An example for an adaptive control of endogenous strategy parameters is the 1/5-th success rule for the mutation strengths by Rechenberg[10]. Rchenberg's rule adapts the mutation strengths in the following way: the whole population makes use of a global mutation strength r for all individuals. If the ratio of successful candidate solutions is greater than 1/5-th, the step size should be increased, because bigger steps towards the optimum are possible, while small steps would be a waste of time. If the success ratio is less than 1/5-th the step size should be decreased. This rule is applied every g generations [3].

3.3 Self-Adaptation

This article focuses on the self-adaptation in evolution strategies (ES). Primarily, the self-adaptation has been gaining popularity due to the flexibility in adaptation to different fitness landscapes [8]. This method enables an implicit learning of mutation strengths in the real-valued search spaces. Self-adaptation bases on the mutation operator that modifies the problem variables using the strategy parameters to search the problem and parameter spaces simultaneously [9]. In line with this, the best values of problem variables and strategy parameters survive during the evolutionary process.

There are different strategy parameters in self-adaptive EAs, e.g., probability of mutation pc, probability of crossover pm, population size Np, etc. In order to better adapt the parameter setting to the fitness landscape of the problem, the self-adaptation of strategy parameters has been emerged that is tightly connected with a development of so named evolution strategies (SA-ES) [10,11,9]. SA-ES were especially useful for solving the continuous optimization problems [12].

As each EAs, the SA-ES also consists of the following components [7]:

- representation of individuals,
- evaluation function,
- mutation,
- crossover,
- Survivor selection.

4 Next Generation Selection Strategies

Two different selection processes were used to pick the next generation of data to be sent through the algorithm. In standard evolutionary strategy literature “given μ parents generating λ offspring (λ >= μ)” [17] The simplest way is to use the (μ,λ)-ES technique where the new set of children is used as the parents for the next iteration, the algorithm in Fig. 1 would use P(t+1) = P’(t). A second technique is to use (μ+λ)-ES technique [17], by looking at the fitness of the parents and the fitness of the new children and sorting their resulting finesses from best to worst. A next generation set is created from the top μ best parents and children, the algorithm in Fig. 1 would use P (t+1) =best of {P’ (t) U P (t)}.

5 Mutation Operators

In classical ES, three types of mutation operators have been applied: an uncorrelated mutation with one step size, an uncorrelated mutation with n-step sizes and correlated mutation using a covariance matrix [7]. In the first mutation, the same distribution is used to mutate each problem variable, in the second, different step sizes are used for different dimensions, while in the last mutation, not only the magnitude in

each dimension but also the direction (i.e., rotation angle) are taken into account [13,14]. Self-adaptation of control parameters is realized in classical evolution strategies (ES) using the appropriate mutation operators controlled by strategy parameters (i.e. mutation strengths) that are embedded into representation of individuals. The mutation strengths determine the direction and the magnitude of the changes on the basis of the new position of the individuals in the search space is determined. This article analyzes the characteristics of classical mutation operators, like uncorrelated mutation with one step size and uncorrelated mutation with n step sizes [15], and the comparison with newest approaches and improvements of self-adaptation strategies like the correlation and hybridization in the recent literature.

As schematically illustrated in Table 1, parameters that yield good solutions with high probability are selected as survivors. It is quite reasonable, at least for local optimization. Concerning selection of the set of parameters to be adapted, we are chosen in our experiments three options from Back (1996) :

5.1 Uncorrelated Mutation with One Step Size (type 1)

As this algorithm loops through each iteration, all parents are being mutated by the same single step size. Using the same standard deviation σ to create all children will have the result that "lines of equal probability density of the normal distribution are hyper-spheres in an l-dimensional space."[17]

- Chromosomes: $\langle x_1,\ldots,x_n, \sigma \rangle$
- $\sigma' = \sigma \bullet \exp(\tau \bullet N(0,1))$
- $x'_i = x_i + \sigma' \bullet N(0,1)$
- Typically the "learning rate" $\tau \propto 1/ n^{½}$

5.2 Uncorrelated Mutation with Individual Step size (type 2)

As this algorithm loops through each iteration, each parent is being mutated by an individual unique step size. Using the different standard deviation σi to create each child will have the result that "lines of equal probability density of the normal distribution are hyper-ellipsoids." [17]

- Chromosomes: $\langle x_1,\ldots,x_n, \sigma_1,\ldots, \sigma_n \rangle$
- $\sigma'_i = \sigma_i \bullet \exp(\tau' \bullet N(0,1) + \tau \bullet N_i (0,1))$
- Two learning rate parmeters: τ' overall learning rate and τ coordinate wise learning rate
- $\tau \propto 1/(2 n)^{½}$ and $\tau \propto 1/(2 n^{½})^{½}$

5.3 Correlated Mutation (type 3)

As this algorithm loops through each iteration, each parent is being mutated by an individual unique step size. Just like the previous uncorrelated mutation with individual step sizes, the formulas use a different standard deviation σi to create each child. The correlated mutation formulas add an additional step of introducing rotation angles to "describe the coordinate rotations necessary to transform the uncorrelated mutation vector to a correlated mutation vector. Now the previous hyper-ellipsoids can be rotated randomly at angle α (t) to give the data more freedom of movement through the plane. ". [17]

- Chromosomes: $\langle x_1,\ldots,x_n, \sigma_1,\ldots, \sigma_n, \alpha_1,\ldots, \alpha_k \rangle$ where $k = n \bullet (n-1)/2$
- and the covariance matrix C is defined as:
- $c_{ii} = \sigma_i^2$
- $c_{ij} = 0$ if i and j are not correlated
- $c_{ij} = ½ \bullet (\sigma_i^2 - \sigma_j^2) \bullet \tan(2\,\alpha_{ij})$ if i and j are correlated

The mutation mechanism is then:

- $\sigma'_i = \sigma_i \bullet \exp(\tau' \bullet N(0,1) + \tau \bullet N_i(0,1))$
- $\alpha'_j = \alpha_j + \beta \bullet N(0,1)$
- $\boldsymbol{x}' = \boldsymbol{x} + \boldsymbol{N(0,C')}$
- $\boldsymbol{C'}$ is the covariance matrix $\boldsymbol{C}$ after mutation of the α values
- $\tau \propto 1/(2n)^{½}$ and $\tau \propto 1/(2n^{½})^{½}$ and $\beta \approx 5°$

6 Numerical and Equations Experiments

As discussed above, the ES achieve self-adaptive search in different manners. To understand similarities and differences of both mechanisms, first a comparison of them through numerical experiments has been carried out. In the experiments, the ES types are applied to several test functions picked from [20]. Usually, a comparative study of different optimization methods is performed by using a set of test functions from the literature. In this paper, five well-known benchmark functions are used. We have chosen two unimodal and three multimodal functions; the functions can also be grouped into separable or non-separable. These are the Sphere function, the generalized Rosenbrock's function, the generalized Rastrigin's function, the Ackley's function, and the Griewank's function. The above functions are defined respectively as:

$$f_{Sphere}(x) = \sum_{j=0}^{D-1} x_j^2, |x_j| \leq 5 \; and \; f_{Sphere}(0,0,\ldots 0) = 0 \qquad (1)$$

$$f_{Rosenbrock}(x) = \sum_{j=0}^{D-2} \left(100\left(x_{j+1} - x_j^2\right)^2 + (x_j - 1)^2\right), \quad (2)$$

$$|x_j| \leq 5 \; and \; f_{Rosenbrock}\,(1,1,\ldots 1) = 0$$

$$f_{\text{Rastrigin}}(x) = \sum_{j=0}^{D-1} \left(\mathrm{x}_j^2 - 10\cos(2\boldsymbol{\pi x_j}) + 10\right), |x_j| \leq 5 \quad (3)$$

$$and \; f_{Rastrigin}\,(0,0,\ldots 0) = 0$$

$$f_{\text{Ackley}}(x) = -20exp\left(-\frac{1}{5}\sqrt{\frac{1}{D}\sum_{j=0}^{D-1}\mathrm{x}_j^2}\right) - exp\left(\frac{1}{D}\sum_{j=0}^{D-1}\cos(2\boldsymbol{\pi x_j})\right) + 20 + \quad (4)$$

$$\exp(1), |x_j| \leq 5 \; and \; f_{Ackley}\,(0,0,\ldots 0) = 0$$

$$f_{\text{Griewank}}(x) = \frac{1}{4000}\sum_{j=1}^{D} x_j^2 - \prod_{j=1}^{D}\cos\left(\frac{x_{ij}}{\sqrt{j}}\right), |x_j| \leq 5 \quad (5)$$

$$and \; f_{Griewank}\,(0,0,\ldots 0) = 0$$

The sphere function is one of the simplest benchmarks. It is a continuous, unimodal and separable problem. The generalized Rosenbrock's global optimum lies inside a parabolic shaped flat valley. It is easy to find the valley but convergence to the global optimum is difficult. This problem is unimodal and non-separable. The generalized Rastrigin function is a complex multimodal separable problem with many local optima. The Ackley's function is a multimodal non-separable problem and has many local optima and a narrow global optimum. The fifth function is highly multimodal and non-separable. It has many widespread local minima, which are regularly distributed.

6.1 Algorithm Parameters Used for Experiments

The ES has many options that should be selected considering both the difficulty of the problems to be solved and the resources available for computation. Based on the recommendation by Back (1996), the following options are adopted:

- The comma strategy with 15 parents and 100 children, i.e., (15,100)-ES is used.

- The discrete recombination is used for the decision variables, and the panmictic intermediate recombination for the standard deviations and the rotation angles.
- Recommended values for parameters τ , $\tau 0$, and β are used.
- All the initial values of αij are set at $\pi/4$.
- We used experiments with 500 and 2000 number of iterations.
- The results of the all experiments are averaged over 20 independent runs.

Table 1. Best normalized optimization results in five benchmark functions. The values shown are the minimum objective function values found by each algorithm with 500 iterations, averaged over 20 Monte Carlo simulations.

Benchmark function	D	*Type1 One step size*	*Type 2 N step size*	*Type3 Correlated*	*Adaptive 1/5 rule*	*Adaptive Cumulative*
Sphere	5	4.0546e-114	**4.9457e-178**	2.6385e-108	9.9912e-54	9.5627e-91
	10	3.4808e-85	**1.5605e-106**	9.6233e-42	1.378e-15	4.9752e-54
	20	**1.531e-61**	3.2016e-40	0.0028815	0.21719	1.1019e-37
Rosenbrock	5	0.21096	0.025719	**1.9657e-06**	0.10707	0.055967
	10	4.8468	**0.021611**	0.058623	2.5602	1.3705
	20	15.6707	**9.9448**	42.3292	25.2971	12.6153
Rastrigin	5	**0**	**0**	**0**	**0**	**0**
	10	0.99496	1.9899	0.99496	**0**	**0**
	20	3.9798	7.9597	**2.3175**	9.6734	3.9798
Ackely	5	**0**	**0**	**0**	**0**	**0**
	10	3.5527e-15	**0**	3.5527e-15	3.5909e-10	**0**
	20	**3.5527e-15**	**3.5527e-15**	0.032326	0.22965	**3.5527e-15**
Griewank	5	**0**	**0**	**0**	**0**	**0**
	10	**0**	**0**	0.007396	**0**	**0**
	20	**0**	**0**	0.019238	1.5486e-07	**0**

Table 2 Best normalized optimization results in five benchmark functions. The values shown are the minimum objective function values found by each algorithm with 2000 iterations, averaged over 20 Monte Carlo simulations.

Benchmark function	D	*Type1 One step size*	*Type 2 N step size*	*Type3 Correlated*	*Adaptive 1/5 rule*	*Adaptive Cumulative*
Sphere	5	**0**	**0**	**0**	5.2256e-204	**0**
	10	**0**	**0**	**0**	9.4191e-51	1.6795e-219
	20	**2.097e-248**	1.7093e-147	1.0321e-15	0.0005046	1.0846e-150
Rosenbrock	5	0.021332	0.00014261	**1.1442e-05**	0.0044214	0.00059227
	10	0.023147	4.5627	**0.0023164**	0.10187	0.045193
	20	12.3151	3.2985	**2.0112**	6.4291	3.8319
Rastrigin	5	**0**	**0**	**0**	**0**	**0**
	10	**0**	0.99496	0.99496	9.9496	0.99496
	20	2.9849	5.9698	**2.3175**	5.9721	2.9849
Ackely	5	**0**	**0**	**0**	**0**	**0**
	10	3.5527e-15	**0**	**0**	**0**	**0**
	20	**3.5527e-15**	**3.5527e-15**	**3.5527e-15**	0.0020593	**3.5527e-15**
Griewank	5	**0**	**0**	**0**	**0**	**0**
	10	**0**	**0**	**0**	**0**	**0**
	20	**0**	**0**	**0**	**0**	**0**

From table 1 and 2 we summarize how many times each algorithm get best results:

Table 3 Statistical results

ES algorithms	*Type1 One step size*	*Type 2 N step size*	*Type3 Correlated*	*Adaptive 1/5 rule*	*Adaptive Cumulative*
Best times	16	20	18	11	16

6.2 Behavior in Higher-Dimensional Search Spaces

For problems in the five-dimensional search space, all the algorithms succeeded in finding good solutions. Especially, the ES Type 3 and Type 2, show good performances and the auto-adaptation process outperform the adaptation one. Increasing the dimension of the search space, performances of the ES algorithms get poorer because the non-separability of the test function becomes serious. The results show clearly that the ES type 2 and 3 is more sufficient and perform better in comparing Sphere and

Rosenbrock functions. Hence , using the rotated angle recommendation give the ES forces to reach a good solutions as well as the strategy parameter with individual step sizes help ES to improve solutions as shown between ES type 1 and type 2.

6.3 Behaviors for Multi-Modal Objective Functions

The results show that the auto-adaptation ES mostly get hard in finding the optimum. The best solutions found by the ESs often stayed at points that are not even local optima. This result suggests that the mechanism of the self-adaptive mutation may have some difficulty in multi-modal objective functions. Since the ES has many options and parameters, and the above experiment uses only a single test function and the recommended options and parameters in literature (Back, 1996), more efficient search may be achieved by the ES by adjusting the options and parameters. It is a subject of further study as well as more detailed examination of the behavior of the self-adaptive mechanism.

7 Conclusion and Future Work

The self-adaptation of strategy parameters is the most advanced mechanism in ES, where the mutation strength strategy parameters control the magnitude and the direction of changes generated by the mutation operator. There are three mutation operators in classical ES, as follows: the uncorrelated mutation with one step size, the uncorrelated mutation with n step sizes and the correlated mutations.

This paper analyses the characteristics of these three mutation operators. In last decades several studies and papers focus at the self-adaptation as tools to improves evolutionary strategy algorithm performance, for that many techniques are used, either mathematics such including geometric function to improve solutions in multi-modal functions [18], incorporating machine learning technique to expect suitable parameters which leads to best solutions[2]. Although the research domain of the ES seems to be already explored and, by this flood of the new nature-inspired algorithms every day, unattractive for the developers, the results of our experiments showed the opposite. Additionally, hybridization adds to a classical self-adapting ES the new value. As the first step of our future work, however, the next step is to extend this comparative study with another well-known meta-heuristics such as PSO, DE, HS in order to recognize the weakness of the self-adaptation ES for eventual future work improvement by hybridization with other local search techniques.

Regardless of the problems mentioned, self-adaptation is a state-of-the-art adaptation technique with a high degree of robustness, especially in real-coded search spaces and in environments with uncertain or noisy fitness information.

References

1. Fister, S. Fong, and I. Fister jr.: A Novel Hybrid Self-Adaptive Bat Algorithm, the Scientific World Journal Volume 2014, paper 709738
2. Kramer, O., : Machine Learning for Evolution Strategies, Springer Journal Vol./2016
3. Kramer, O., Ciaurri, D.E., Koziel, S.: Derivative-free optimization. In: Computational Optimization and Applications in Engineering and Industry. Springer (2011)
4. Nikolaus Hansen, Dirk Arnold, Anne Auger. Evolution Strategies. Janusz Kacprzyk; Witold Pedrycz. Handbook of Computational Intelligence, Springer, 2015.
5. Deb, K.: Multi-Objective Optimization Using Evolutionary Algorithms. John Wiley & Sons, Inc., New York (2001)
6. Schwefel, H.P.: Numerische Optimierung von Computer–Modellen mittels der Evolutions strategie. Birkh¨auser, Basel (1977)
7. Eiben, A., Smith, J.: Introduction to Evolutionary Computing. Springer, Berlin(2003)
8. Back, T., Hammel, U., Schwefel, H.-P.: Evolutionary Computation: Comments on the History and Current State. IEEE Trans. Evolutionary Computation 1(1), 3–17(1997)
9. Beyer, H.-G., Deb, K.: On self-adaptive features in real-parameter evolutionary algorithms. IEEE Trans. Evolutionary Computation 5(3), 250–270 (2001)
10. Rechenberg, I.: Evolutionsstrategie, Optimierung technischer Systeme nach Prinzipien der biologischen Evolution. Frommann-Holzboog, Stuttgart (1973)
11. Schwefel, H.P.: Numerische Optimierung von Computer-Modellen mittels der Evolutions strategie. Birkh¨auser, Basel (1977)
12. Fister, I., Mernik, M., Filipi˘c, B.: Graph 3-coloring with a hybrid self-adaptive evolutionary algorithm. Comp. Opt. and Appl. 54(3), 741–770 (2013).
13. Hansen, N.: The CMA evolution strategy: A tutorial. Vu le 29 (2005).
14. Igel, C., Hansen, N., Roth, S.: Covariance matrix adaptation for multi-objective optimization. Evolutionary Computation 15(1), 1–28 (2007).
15. Fister, and I. Fister jr.: On the Mutation Operators in Evolution Strategies, Chapter Adaptation and Hybridization in Computational Intelligence.
16. Silja Meyer-Nieberg : Dissertation, self-adaptation in evolution strategies(2007)
17. L. N. Castro, Fundamentals of Natural Computing, Chapter 3, Taylor and Francis Group, LLC. New York, 2006
18. Sandra DeBruyne and Devinder Kaur, "Comparison of Uncorrelated and Correlated Evolutionary Strategies with Proposed Additional Geometric Translations" , proceedings of the International Conference on Genetic and Evolutionary Computing , GEM 13, held at WORLDCOMP 2013 Congress, , Las Vegas , USA. (2013)
19. Z. Song, A. Kusiak, "Evolutionary Strategy and Applications" Univerity of Iowa. (2009)
20. H. Kita. A comparison study of self-adaptation in evolution strategies and realcoded genetic algorithms. Evolutionary Computation, 9(2):223–241, 2001.
21. Eiben, A.E., Hinterding, R., Michalewicz, Z.: Parameter control in evolutionary algorithms. Trans. Evol. Comp. 3(2), 124–141 (1999)
22. Mezura-Montes E, Palomeque-Ortiz AG (2009) Self-adaptive and deterministic parameter control in differential evolution for constrained optimization. Constraint-Handl Evol Optim 189:95–120.
23. Kramer, O., :Evolutionary self-adaptation: a survey of operators and strategy parameters, Springer Journal Vol./2010.
24. S K Goudos, Konstantinos,a comparative study of common and self-adaptive deferencial evolution strategies, World Conference on Information Technology WCIT 2010.