Date of publication MAR-29, 2023, date of current version AUG-18, 2022
www.computingonline.net/ computing@computingonline.net

Print ISSN 1727-6209
Online ISSN 2312-5381
DOI 10.47839/ijc.22.1.2881

Cross-Selection Based Evolution Strategies

LARYSA O. KHILKOVA[1]
[1]V.Dahl East Ukrainian National University, Ukraine (e-mail: LarisaHilkova@gmail.com)
Corresponding author: Larysa O. Khilkova (e-mail: LarisaHilkova@gmail.com)

ABSTRACT A search for an optimal value of a complex multi-dimensional continuous function is still one of the most pressing problems. The genetic algorithms (GA) and evolution strategies (ES) are methods to solving optimization problems that is based on natural selection, the process that drives biological evolution. Our goal was to use evolutionary optimization methods to find the global optimal value (minimum) of a non-smooth multi-dimensional function with a large number of local minimums. We took several test functions of different levels of complexity and used evolution strategies to solve the problem. The standard evolution strategies, which work well with smooth functions, gave us various points of local minimums as a solution, without finding the global minimum, for the complex function. In our work, we propose a new approach: the cross-selection method, which, in combination with previously developed methods - adaptive evolution strategies, gave a good result for the searth for the global minimum the complex function.

KEYWORDS Evolution Strategies, Cross-Selection Method, Optimization, Evolutionary Operators, Mutation, Selection, Recombination, Population, Generation, Fitness-Function.

I. INTRODUCTION

Many disciplines involve optimization at their core. In physics, systems tend towards their lowest energy state subject to physical laws. In business, corporations are focused on maximizing profits. In biology, the fittest organisms are more likely to survive. Mathematically, each of these processes is described using an objective function that depends on various parameters. Depending on the optimization problem, relevant parameters might include effjciency, safety, phisical metrics or precision. Optimization problems are often posed as a search in a space defined by a set of coordinates - values of these parametrs.

The basic optimization problem is:

$$F(x) \rightarrow opt, \, x \in X.$$

Here, X is a feasible set, a solution search domain, and x is a design point. A design point can be represented as a vector of values corresponding to difgerent design variables. An n-dimensional design point is written:

$$x = (x_1, x_2, ..., x_n)$$

Any value of x from among all points in the feasible set X that minimizes the objective function is called a solution or minimizer.

A huge number of methods are devoted to solving this problem: direct methods, stochastic methods, methods of discrete and continuous optimisation. In recent years, evolutionary methods: genetic algorithms and evolution strategies have been added to these methods (see [1], [2] and their bibliography). Which algorithm is the best? There is no reason to prefer one algorithm over another for all optimization problem. If one algorithm performs better than another algorithm to one class of problems, then it will perform worse to another one.

Our goal was to find the global optimal value (minimum) of a complex function with a large number of local minima. We took several functions of different levels of complexity: a spherical function (Fig.1(a)), Ackley function (Fig.1(b)) and Weierstrass function (Fig.2) for the test. These functions are defined and continuous in R^n. The Weierstrass function is the most complex a non-smooth multi-dimensional function with a large number of local minimums. The majority of the optimization methods cannot find minimum of this function.

We used classical and new, proposed in this work, evolutionary algorithms, namely evolution strategies, to solve the problem.

The evolutionary algorithms (EA) originate in the 70s of the 20th century. The processes that drive biological evolution were the basis of them. There are three main variants of the evolutionary algorithms: genetic algorithms, evolutionary programming and evolution strategies. The evolution strategies (ES), the third main variant of EA, were founded by students at the Technical University of Berlin (TUB) (see [3], [4]). Namely, it is the third direction that our work is devoted to. The standard evolution strategies: $(1+1)$, $(\lambda+\mu)$, (λ,μ), simple and adaptive, which quickly find a solution for smooth functions, for example spherical function, cannot find a global solution the Weierstrass function.

We propose a new approach: the Cross-Selection Method. This method, as well as other evolutionary algorithms, is based on a biological idea. We act like a biologist who wants to improve the fitness of a plant: we take two selected good specimens and cross them with each other. The resulting offspring can have very high fitness. The cross-selection method in combination with adaptive evolution strategies, which we called cross-selection adaptive evolution strategy (CS+AES), gave a good result for the searth for the global minimum of Weierstrass function.

II. MATERIAL AND METHODS

A. STATEMENT OF THE PROBLEM

Suppose we have to optimize some objective or quality function F, called the fitness-function, with respect to a set of decision variables or control parameters $x := \{x_1, x_2, ..., x_n\}$, which in the context of evolution strategies often referred to as object parameters

$$F(x) \to opt,\ x \in X.$$

X is the search domain, it can be discrete or continuous, finite or infinite. For example, the real-valued n-dimensional space R^n, the space of integers Z^n, or the binary space B^n can be used as a search domain. In our work, the search domain

$$X \in R^n.$$

As the optimal value, we consider the minimum

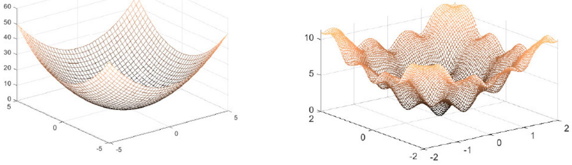
$$F(x) \to min,\ x \in X.$$

B. TEST FUNCTIONS

We took three n-dimensional test fitness-functions: spherical, Ackley and Weierstrass functions. In Fig.1, 2 you can see plots of these test functions for $n = 2$.

- The spherical function (Fig.1(a)):

$$f(x) = \sum_{i=1}^{n} x_i^2, \quad x_i \in [-5.12, 5.12]. \tag{1}$$



(a) (b)

Figure 1. (a) Spherical function, (b) Ackley function

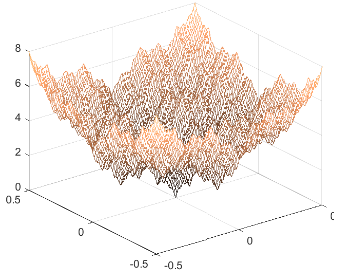


Figure 2. Weierstrass function

This smooth function with a single extremum is often used as a test function for various combined ([1], [5]–[7]) algorithms. This is a good preliminary test to check the performance of any optimization algorithm.

- The Ackley function (Fig.1(b)):

$$f(x) = 20 + e - 20\exp\left(-0.2\sum_{i=1}^{n}\frac{x_i^2}{n}\right) - \exp\left(\sum_{i=1}^{n}\frac{cos 2\pi x_i}{n}\right),\ x_i \in [-2, 2]. \tag{2}$$

This test function was proposed in [8]. Numerous local extrema of this smooth test function make it a rather difficult optimization problem of finding the global extremum.

- The Weierstrass function (Fig.2):

$$f(x) = \sum_{i=1}^{n}\left\{\sum_{k=0}^{k_{max}}\left[a^k\cos(2\pi b^k(x_i+0.5))\right]\right\} - n\sum_{k=0}^{k_{max}}\left[a^k cos(\pi b^k)\right],\ x_i \in [-0.5, 0.5], \tag{3}$$
$$a = 0,5,\ b = 3,\ k_{max} = 20.$$

This test function was proposed in [9]. This multidimensional function is very difficult to optimize, it has many local extrema, in addition, it has an interesting property: by $n \to \infty$, it is everywhere continuous, but nowhere differentiable. The vast majority of classical optimization algorithms cannot find the global extremum of the Weierstrass function. We use this function to test the performance of the considered evolution strategies.

Test functions have different levels of complexity for optimization: the simple spherical function with a single minimum, the smooth Ackley function with many local minima, and the most complex non-smooth Weierstrass function with a huge number of local minima, which is practically impossible to optimize. We are testing our algorithms on these functions.

All these functions have a global minimum at the origin $x^* = 0$ and

$$F_{min} = F(x^*) = 0.$$

The main criterion of quality for all considered evolution strategies is a order of approach of solution to the known global minimum.

C. BASIC EVOLUTION STRATEGY (ES)

Evolution strategies operate on populations $\mathfrak{P}(\kappa) = \{\alpha^1, \alpha^2, ..., \alpha^\kappa\}$ of individuals α^i $(i = 1, ..., \kappa)$ of a size κ. Each individual α^i can be considered as a union of object parameters $x^i = \{x_1^i, x_2^i, ..., x_n^i\}$, a set of strategy parameters s^i and a value fitness-functions $F^i = F(x^i)$

$$\alpha^i := (x^i, s^i, F^i).$$

We assume that the individuals in the population are ordered according to the value of the fitness-function

$$\mathfrak{P}(\kappa) = \{\alpha^1, \alpha^2, ..., \alpha^\kappa\},\ F^1 \gg F^2 \gg ... \gg F^\kappa.$$

So, α^1 has the best fitness-function.

The goal of the evolution strategy is to obtain a population $\mathfrak{P}^{(g)}(\kappa)$ whose first individual is a solution to the optimization problem with a given precision.

Denote:

1) $\mathfrak{P}_p^{(g)}(\mu)$ is a parent population at the g-th stage of evolution (the number of a generation), consisting of μ members;
2) $\mathfrak{P}_c^{(g)}(\lambda)$ is a child population at the g-th stage of evolution, consisting of λ members.

Basic algorithm ES can be represented as the following pseudo-code:

Algorithm 1 Basic ES

```
$g := 0$
Initialize $\mathfrak{P}_p^0(\mu)$ := a random population
**while not** (Stop criteria) **do**
    $\mathfrak{P}_c^{(g)}(\lambda)$ := Evolution operators $(\mathfrak{P}_p^{(g)}(\mu))$
    $\mathfrak{P}_p^{(g+1)}(\mu)$ := Selection $(\mathfrak{P}_p^{(g)}(\mu), \mathfrak{P}_c^{(g)}(\lambda))$
    $g := g + 1$
**end while**
```

The main evolutionary operator applied to the parent population is *mutation*. It is the only operator on the parent population for ES$(1 + 1)$. In strategies like ES $(\mu + \lambda)$, mutation is supplemented by a recombination operator. The selection operator is used in all ES and sets up the direction of evolution in the object parameter space.

As termination conditions the standard stopping rules can be used:

- Resource criteria:
  - maximum number of generations;
  - maximum program run time.
- Convergence criteria:
  - in the space of fitness-values;
  - in the space of object parameters;
  - in the space of strategy parameters.

Our goal is to obtain a solution to the optimization problem with high precision. We will look at several different ESs and discuss how they fulfill this problem for the considering us functions.

D. EVOLUTIONARY OPERATORS

In the evolution strategies we are considering, three evolutionary operators are used:

- selection operator;
- mutation operator;
- recombination operator.

1) Selection

The selection operator with different variations is used in all evolutionary algorithms. Just as in the wild world, an animal that is more adapted to the conditions of the surrounding world has a higher chance of producing offspring, so in ES we leave the set of object parameters that give the best value of the fitness-function for the production of future generations.

The selection operator that selects μ individuals of the next generation can be defined as follows:

$$\text{Selection}(\mu) := \{\alpha_\gamma^1, \alpha_\gamma^2, ..., \alpha_\gamma^\mu\},$$

where the α_γ^i is the i-th best individual of the γ individuals of the current generation.

The value of γ is defined differently for ES $(\mu+\lambda)$ and ES (μ, λ). In an ES of the first type, μ individuals are selected from the parent and child populations and $\gamma = \mu + \lambda$. This ensures that the best obtained individual always remains in the new population. In ES of the second type, the selection of a new parent population is made only from the child population, i.e. $\gamma = \lambda$ and the condition $\mu \ll \lambda$ is natural. ES of the second type guarantees complete renewal of the population at each evolutionary step.

Both types of selection have their own specific applications. ES (μ, λ) is recommended for use in unbounded spaces, such as the object space $X = R^n$ (see [4]), while ES $(\mu + \lambda)$ are recommended for discrete and finite object spaces (see [10], [11]). We use only ES $(\mu+\lambda)$ in our work.

2) Mutation

The mutation operator is the main one for ES. The general task of the operator is to slightly improve the set of object parameters in such way as to get into the domain of a local extremum. Each new value x is defined as the offset of its old value along the random mutation vector r

$$x^i = x^{i-1} + r. \tag{4}$$

The original ES (see [3], [4]) was designed for discrete problems and used small mutations in the discrete search space. In [12] ES was modified for use in optimization continuous functions, it was proposed to use the vector of normally distributed random values $r := \{N(0,\sigma_1), N(0,\sigma_2), ..., N(0,\sigma_n)\}$ as a random mutation vector. Besides normally distributed mutations, Cauchy mutations ([13]–[16]) have also used in the context of ES and evolutionary programming. Other interesting approach has proposed in [17], in which the authors use the natural gradient to update a parameterized search distribution in the direction of higher expected fitness. Now de facto standard in continuous domain evolutionary computation is Covariance Matrix Adaptation-ES ([18]–[20]), in which the mutation ellipsoids are not constrained to be axis-parallel, but can take on a general orientation.

In our work, we propose to define the mutation vector $r = \{r_1, r_2, ..., r_n\}$ as follows

$$\begin{aligned}
r_1 &= \sigma_1 \cdot \rho \cdot \cos\phi_1 \cdot \cos\phi_2 \cdot \dots \cdot \cos\phi_{n-1};\\
r_2 &= \sigma_2 \cdot \rho \cdot \sin\phi_1 \cdot \cos\phi_2 \cdot \dots \cdot \cos\phi_{n-1};\\
r_3 &= \sigma_3 \cdot \rho \cdot \sin\phi_2 \cdot \dots \cdot \cos\phi_{n-1};\\
&\dots\\
r_n &= \sigma_n \cdot \rho \cdot \sin\phi_{n-1},
\end{aligned} \tag{5}$$

where $\rho \geq 0$, $\phi_1 \in [0, 2\pi]$, $\phi_2, ..., \phi_{n-1} \in \big[-\pi/2, \pi/2\big]$ are hyperspherical coordinates, $\sigma_1, \sigma_2, ..., \sigma_n$ are constants specifying the spread of values for each of the coordinates, $\rho = N(0,1)$, $\rho \geq 0$ is a normally distributed random positive variable, and the angles $\phi_1, ..., \phi_{n-1}$ are uniformly distributed each in its range. We assume that all random variables $\rho, \phi_1, ..., \phi_{n-1}$ are independent. And then the probability density function in hyperspherical coordinates for the mutation vector is equal

$$\begin{aligned}
p(r) &= p(\rho, \phi_1, ..., \phi_{n-1})\\
&= p(\rho) \cdot p(\phi_1) \cdot ... \cdot p(\phi_{n-1})\\
&= \frac{e^{-\rho^2/2\sigma^2}}{\sigma \cdot 2^{n-1/2}\pi^{n+1/2}}.
\end{aligned} \tag{6}$$

Since almost all values of the normally distributed random variable $N(0,1)$ fall within the range $(-3,3)$, then, due to (4)-(6), the random variable x^i is normally distributed inside an ellipsoid with semi-axes $3\sigma_1, 3\sigma_2, ..., 3\sigma_n$ and center at the point x^{i-1} (Fig.3).

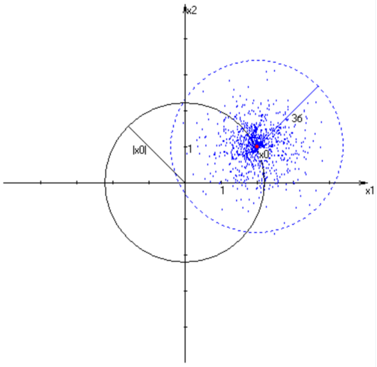


Figure 3. $2D$ cloud simulation result for $\sigma = 1$

If $\sigma_1 = \sigma_2 = ... = \sigma_n = \sigma$, then the mutation is called isotropic. In our work, we use only this type of mutation.

The parameter σ is strategic, it determines the size and distribution of points within the mutational 3σ-cloud and can significantly affect the rate of evolution. σ is a compromise parameter. With large values of σ for a simple test function, you can quickly get a point close to a solution, but it cannot be improved further. For small σ the solution improves very slowly but steadily. Thus, the value of σ must somehow change during evolution in order to obtain the solution of good precision. We assume that the good points of the mutation cloud are the points that fall into the n-ball $R(0, |x^i|)$ (see Fig.3).

To determine the optimal value of σ [**?**] proposed the $1/5$-th rule, which says that about 20% points of the mutational 3σ-cloud should be good, namely they are within the ball $R(0, |x^i|)$. This value varies for various simple functions considered earlier, and is close to 20% (hence the name of the rule). For example, for a spherical function it is 27% (see [**?**]). This value was not found for Weierstrass function, so we will use the general rule. In [12], [21] there are theoretical justifications of the $1/5$-th rule for different ES. Today, almost all ESs use some type of self-adaptation to adjust the algorithm settings. In addition, Beyer and Deb in [22] have shown that even ESs without explicit self-adaptation can exhibit self-adaptive behavior.

The $1/5$-th rule for Weierstrass function works only on the first epochs, then, due to the complexity of the search domain, the percentage of good mutations is usually quite low, and the optimization parameter σ tends to decrease. We used the following modification of this rule (Alg.2), which gave a good result for our function. The value of ε here can be thought of as a value related to the precision of the method (but ε is not a precision).

Algorithm 2 The $1/5$-th ε-rule

```
if the ratio of successful mutations to total mutations is less than $1/5$ then
    $\sigma := C \cdot \sigma$                    {The constant $C = 0.817$}
    if $\sigma < \varepsilon$ then
        $\sigma := \sigma_0$
    end if
else
    $\sigma := \frac{\sigma}{C}$
end if
```

Plots of adjustment of the strategic parameter σ and the

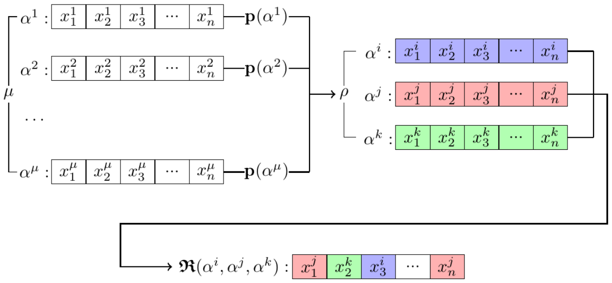

$\alpha^1 : x_1^1 \; x_2^1 \; x_3^1 \; \cdots \; x_n^1$ — $\mathbf{p}(\alpha^1)$
$\alpha^2 : x_1^2 \; x_2^2 \; x_3^2 \; \cdots \; x_n^2$ — $\mathbf{p}(\alpha^2)$
μ
$\cdots$
$\alpha^\mu : x_1^\mu \; x_2^\mu \; x_3^\mu \; \cdots \; x_n^\mu$ — $\mathbf{p}(\alpha^\mu)$
ρ
$\alpha^i : x_1^i \; x_2^i \; x_3^i \; \cdots \; x_n^i$
$\alpha^j : x_1^j \; x_2^j \; x_3^j \; \cdots \; x_n^j$
$\alpha^k : x_1^k \; x_2^k \; x_3^k \; \cdots \; x_n^k$
$\mathfrak{R}(\alpha^i, \alpha^j, \alpha^k) : x_1^j \; x_2^k \; x_3^i \; \cdots \; x_n^j$

Figure 4. Recombination operator

corresponding % of good mutation for different strategies can be found in Section III.

3) Recombination

In evolution strategies ES $(\mu/\rho + \lambda)$ and ES $(\mu/\rho, \lambda)$, presented for the first time in the work [6], in addition to the mutation operator, a recombination operator is used to form a new individual. Like a crossover operator in genetic algorithms, the recombination operator uses information of not one, but several (ρ) parents to form a child, but unlike the crossover, only one child is formed ([23], [24]). If a number of parents is more than 2 ($\rho > 2$), then we speak about multi-recombination.

The mechanism for selecting ρ individuals from the parental population for recombination may differ. In our work we use equiprobable and tournament selection.

In equiprobable selection, all individuals have the same probability of being selected:

$$p(\alpha^1) = p(\alpha^2) = ... = p(\alpha^n) = \frac{1}{\mu}. \tag{7}$$

In tournament selection ([19], [25]), the best individual has a higher probability of being selected. Since in our work we minimize the function and the minimum is reached at the origin of coordinates, then the probability of being selected for each parent individual is determined by the formula

$$p(\alpha^i) = \frac{\frac{1}{F(x^i)+1}}{\sum_{j=1}^{\mu} \frac{1}{F(x^j)+1}}, \quad p(\alpha^1) > p(\alpha^2) > ... > p(\alpha^\mu). \tag{8}$$

Selected ρ parents $\alpha^i, ..., \alpha^j$ form a child $\mathfrak{R}(\alpha^i, ..., \alpha^j)$, randomly passing it the values of their features, as shown in Fig.4. Thus, the child-individual inherits the values of the object parameters of its parents.

E. CONSIDERED EVOLUTION STRATEGIES

In this paper, we consider three types of evolution strategies, whose algorithm schemes are shown in Fig.5, 6:

1) ES $(1+1)$: 1 parent + 1 child (Fig.5(a));
2) ES $(\mu/\rho+\lambda)$: μ parents produce λ children (Fig.5(b));
3) Cross-Selection Method (α, π) + ES: α parents, obtained as a result of some evolution strategy by crossing, produce π ideal children (Fig.6). The Cross-Selection Method is a new method proposed in our work; it operates not with random generations, but with generations obtained as a result of evolution. A description of this method is in Section II-F.

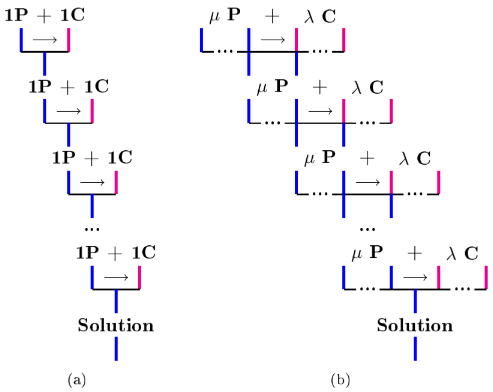


Figure 5. Algorithm (a) ES $(1+1)$, (b) ES $(\mu+\lambda)$ schemes (P – parents, C – children)

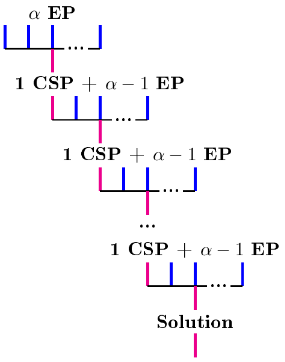


Figure 6. Algorithm CS+ES $(\alpha, 1)$ scheme (EP – new populations, CSP – Cross-Selection population)

1) Evolution Strategy $(1+1)$

In ES $(1 + 1)$, a single child-individual is obtained by mutating a single parent-individual, the best one is selected from the two resulting individuals, which becomes the parent (Fig.5(a)). Interestingly, the first implementations of ES $(1+1)$ were experimental and mutations were performed physically, so considering a large number of children was very laborious (see [**?**]).

2) Evolution Strategy $(\mu/\rho+\lambda)$

In ES $(\mu/\rho+\lambda)$ we deal with μ parents, from which, using evolutionary operators, we form λ children (Fig.5(b)). Each child has ρ parents. Then the best μ individuals become parents of the next generation [6].

F. CROSS-SELECTION METHOD CS (α, π)

The idea of this method came to us, as well as all the previous ones for our predecessors, from biology. The main problem that we encountered for the ES algorithms is the degeneration of the population, i.e. in other words, when all individuals in the population become sufficiently good and close to each other. This is what happens in the world around us, often isolated populations die out because they stop improving.

When a breeder wants to grow a grade of plant with the properties he wants, he takes several parent plants that carry at least one desired property and crosses them in the hope of getting those properties into offspring. This is the main idea of the Cross-Selection Method.

In the Cross-Selection Method, we operate not with parent individuals, but with parent populations.
Denote:
α – number of parent populations,
π – number of populations obtained as a result of Cross-Selection.

In the first epoch, we form and improve by any ES α new Evolution Populations (EP). As a result of evolution, each population is the carrier of the properties we need. Next, we take the best individual from each population for Cross-Selection as a parent. In Sec.III-C we will show that the first representative obtained as a result of the evolution of Cross-Selection Population gives an excellent result. Thus we form π Cross-Selection Populations. For each next epoch, we add $\alpha - \pi$ new Evolution Populations to them. These populations become the parent for the next Cross-Selection Population (Alg.3).

III. RESULTS

In our work, we considere several ESs: ES $(1+1)$ and ES $(10/3+20)$ with a fixed value of σ and with adaptation of this value according to the $1/5$-th rule and $1/5$-th ε rule (Alg.2), with equiprobable and tournament selection. In the Sections III-A, III-B, III-C we will describe how these strategies work with our test functions and show the results of this work on the charts.

Evolution strategies refer to random methods. The results obtained from several runs of the same strategy for the same function may differ. Therefore, for all methods, we present the averaged results obtained from 50 independent runs of the program. This does not apply to plots for σ and % good mutations, for which averaging does not make sense.

A. RESULTS OF ES $(1+1)$

In the ES $(1+1)$ we have one parent, from which one child is obtained by mutation. In our work, we use isotropic mutation, for which the value of the strategic parameter σ is the same for all control parameters. The value of σ can be fixed (for simple ES) or changed during evolution (for adaptive ES). Then, out of the two received individuals, the best one is selected, which becomes the parent for the next generation (Fig.5(a)).

Algorithm 3 CS+ES (α, π)

```
$e := 1$        {1-st epoch}
**for all** $i \in \{1, ..., \pi\}$ **do**
    {Form $i$-th CS population}
    **for all** $j \in \{1, ..., \alpha\}$ **do**
        {The evolution of $j$-th random Population for $j$-th parent}
        $\mathfrak{P}_j^0 :=$ random Population
        $\mathfrak{P}_j := ES(\mathfrak{P}_j^0)$
        $\mathfrak{S}_i^0[j] := \mathfrak{P}_j[1]$
    **end for**
    $\mathfrak{S}_i := ES(\mathfrak{S}_i^0)$ {$i$-th ideal child}
**end for**
**while not** (Stop criteria) **do**
    $e := e + 1$        {$e$-th epoch}
    **for all** $i \in \{1, ..., \alpha\}$ **do**
        **if** $i \le \pi$ **then**
            $\mathfrak{S}^0[i] := \mathfrak{S}_i[1]$ {Parents received as a result of CS}
        **else**
            $\mathfrak{P}_{i-\pi}^0 :=$ random Population
            $\mathfrak{P}_{i-\pi} := ES(\mathfrak{P}_{i-\pi}^0)$
            $\mathfrak{S}^0[i] := \mathfrak{P}_{i-\pi}[1]$ {Parents received as a result of ES of a new random population}
        **end if**
    **end for**
**end while**
$\mathfrak{S} := ES(\mathfrak{S}^0)$ {ES for the Last CS Population}
Solution $:= \mathfrak{S}[1]$ {A solution is the best object of the last CS population}
```

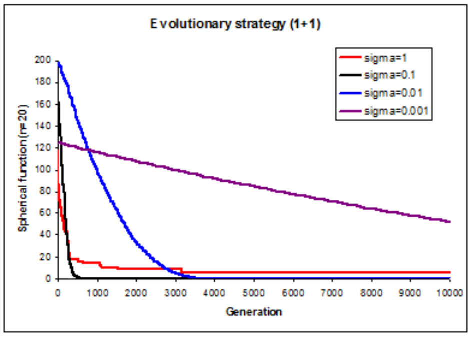


Figure 7. Work ES $(1, 1)$ for various fixed σ

The ES $(1+1)$ is too weak for complex Ackley and Weierstrass functions, but gives a good result for the simple spherical function. Using this strategy as an example, we were able to study the work of the ESs at different σ values (Fig.7) and when adapting σ according to the $1/5$-th rule (Fig.8). In Fig.9 you can see the dependence between σ value and % good mutation.

Based on our results, we can make the following conclusions:

1) ES $(1+1)$ produces a good global minimum search result for simple functions with one local minimum (Fig.7, Fig.8).
2) For different fixed values of σ the results are different (Fig.7). The general trend is that for small values of sigma ($\sigma = 0.001$) the method converges very slowly,

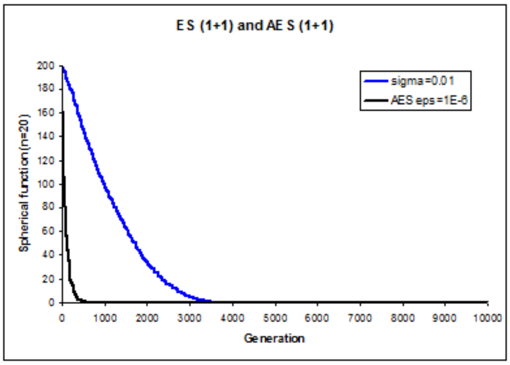


Figure 8. Comparing the work of the best ES $(1, 1)$ for a fixed $\sigma = 0.01$ $(F_{min} \approx 0.00037)$ and AES $(1, 1)$ $(F_{min} \approx 3.2E - 11)$

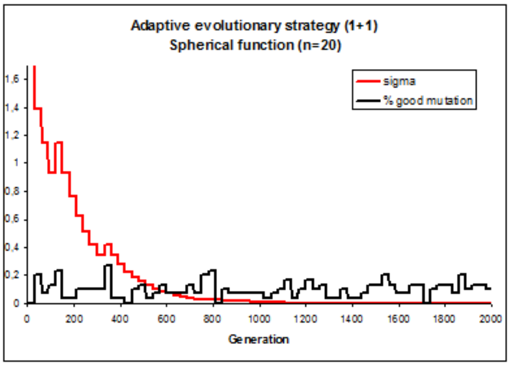


Figure 9. Adaptation of σ and $\%$ good mutation by $1/5$-th rule for spherical function

for large values of sigma ($\sigma = 1$) the solution quickly stops to improve.

3) The result of AES $(1 + 1)$ is better than ES $(1 + 1)$ with fixed σ (Fig.8).
4) When using AES $(1 + 1)$ with adaptation according to the $1/5$-th rule, the value of σ has a pronounced tendency to decrease (Fig.9).

B. *RESULTS OF AES* $(10/3 + 20)$

The second ES we consider is the Adaptive Evolution strategy AES $(10/3 + 20)$ with adaptation of σ by $1/5$-th ε rule. This strategy uses 10 parents to produce 20 children. For each child by equiprobable or tournament selection 3 parents are selected. From these 3 parents, a child is produced by random discrete recombination (Fig.4), which is then subjected to mutation. Out of 30 parents and children, the top 10 are selected to become the parents of the next generation. The general scheme of the ES $(\mu + \lambda)$ algorithm can be seen in Fig.5(b). This method produces excellent results for the spherical function and not bad results for the Ackley and Weierstrass functions. In the Fig.10, 11 we can see the results of these strategies. The AEST with adaptation σ by $1/5$-th $\varepsilon = 10^{-6}$-rule and tournament selection produced the best results for both functions.

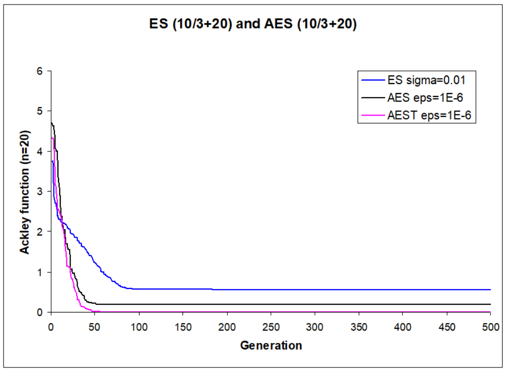


Figure 10. Work of ES $(10/3 + 20)$ with fixed $\sigma = 0.01$, AES (10/3+20) and AEST (10/3+20) for Ackley function

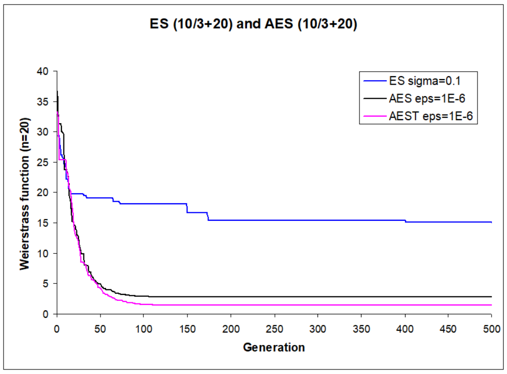


Figure 11. Work of ES $(10/3 + 20)$ with fixed $\sigma = 0.1$, AES $(10/3 + 20)$ and AEST $(10/3 + 20)$ for Weierstrass function

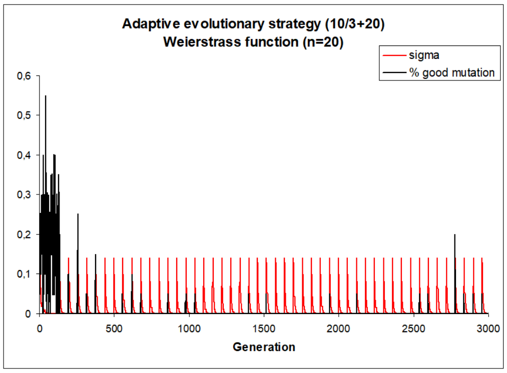


Figure 12. Adaptation of σ and $\%$ good mutation by $1/5$-th ε rule for Weierstrass function

In Fig. 12 we see the change of σ and the corresponding $\%$ good mutation for Weierstrass function. We observe especially large bursts of the $\%$ good mutations in the first epochs, then there may be such bursts, but they are single.

Based on our results, we can make the following conclusions:

1) AES $(10/3 + 20)$ with adaptation σ by $1/5$-th ε rule

Table 1. Results of AEST$(10/3+20)$ for 10 populations after 1000 and 2000 generations for the Weierstrass function

	x_1	x_2	...	x_{20}	$F_{1000}(x)$	$F_{2000}(x)$
1	0	0,0001	...	-0,0002	1,4577	1,4574
2	-0,0001	0	...	-0,0002	1,7953	1,7951
3	0	-0,0001	...	-0,0001	4,8579	4,8571
4	0,0001	-0,0028	...	-0,0001	2,5782	2,5779
5	0	0,0001	...	-0,0001	2,3010	2,2997
6	0	0	...	-0,0001	1,4271	1,4265
7	0,0001	0	...	-0,1192	4,9274	3,8675
8	-0,0023	0	...	0	2,3961	2,3855
9	-0,0001	0,0001	...	0	0,3037	0,3037
10	0	0	...	0	3,3364	3,1524

gives excellent results in finding the global minimum for a simple spherical function with one local minimum and a good result for complex Ackley and Weierstrass functions (Fig.10, Fig.11).

2) Among ES $(10/3+20)$ with fixed σ, AES $(10/3+20)$ with adaptation σ by $1/5$-th ε rule and AEST $(10/3+20)$ with tournament selection, the third method gave a significantly better result for both functions.
3) When σ is adapted by the $1/5$-th ε rule, bursts of $\%$ good mutations, especially in the first epochs of evolution, are possible.
4) For almost all the considered strategies, there is no qualitative improvement of the solution after the 100-th generations.

As we can see, after the 100-th generation, the result practically does not improve. What is the reason for such stabilization, because the optimal result has not yet been found? Tab.1 give us the answer to this question. Let's analyze the results recorded in it, obtained for the Weierstrass function. This analysis led us to the Cross-Selection Method.

We took 10 separate populations and improved them during the first 1000 and then 2000 generations. The results of this evolution are partially shown in the table. Since we know that the minimum of Weierstrass function $F_{min} = F(0,0,...,0) = 0$, we see that the 1-st population has the best value $x_1 = 0$, the 2-nd one has the best value $x_2 = 0$, and the 8-th one has the best values $x_2 = 0$ and $x_{20} = 0$. We also see that each population has not very good properties. Thus, we can consider that each population has fallen into the trap of one of the local minima. The table also clearly shows that the results for 1000 and 2000 epochs differ little. What will produce us a qualitative improvement? We just need to cross the individual obtained as a result of the evolution of population 1 with the individual obtained as a result of the evolution of population 2. If one of the children gets the properties $x_1 = 0$ from the first parent and $x_2 = 0$ from the second, then such a child will most likely be significantly better than their parents. By crossing good parents, we can get the perfect child. Our expectations were fully justified, you can see the result in Sec. III-C.

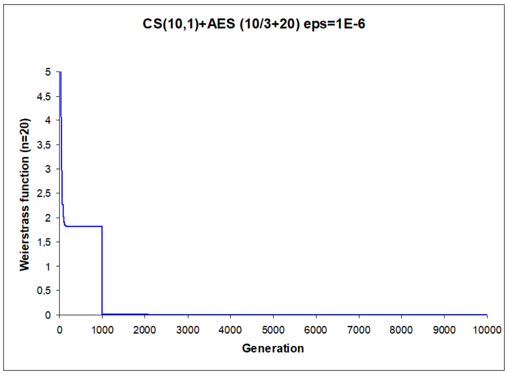


Figure 13. Cross-Selection $(10,1)$ + Adaptive Evolution Strategy $(10/3+20)$ for $20\,D$ Weierstrass function

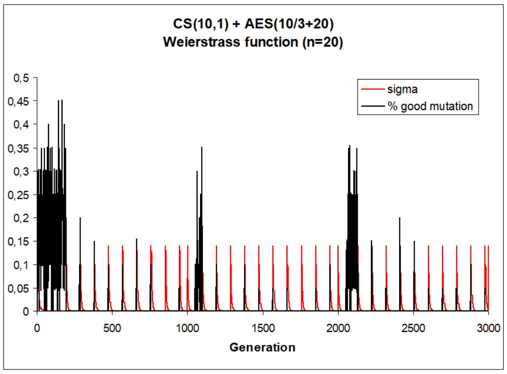


Figure 14. Adaptation of σ and $\%$ good mutation by $1/5\,\varepsilon$ rule for $20\,D$ Weierstrass function

C. RESULTS OF CS $(10,1)$ + AES $(10/3+20)$

As shown in Sec.III-B AES $(10/3+20)$ produces a enough good result. Our next task was to improve this result using the Cross Selection Method. The main focus of this method is not on evolutionary operators, on objective parameters, but on parents. Each parent has already gone through the process of evolution using one of the ES, i.e. he's enough good already. The general scheme of the algorithm can be seen in Fig.6.

In our work with the Cross-Selection Method, we used the Adaptive Evolution Strategy $(10/3+20)$ for 1000 generations (Fig.13). We see a qualitative jump improvement in the result by the first Cross-Selection and a little visible jump improvement in the result by the second one. Also, after each Cross-Selection, we see a $\%$ good mutation burst (Fig.14) for the Weierstrass function.

Using the CS$(10,1)$+AES$(10/3+20)$ method for 20 dimensional Weierstrass function we obtained the following result:

$$F_{2000}(x) = F(0.0001, 0.0002, 0, -0.0001, -0.0002, 0, 0, 0, 0, 0, 0, 0, 0, -0.0002, 0, 0, 0, -0.0001, 0, 0) = 2.63 \cdot 10^{-5}.$$

And this result is certainly very good.

Based on our results, we can make the following conclusions:

1) With each Cross-Selection, we see a qualitative jump in improving the result.
2) A large % of good mutations can be observed after each Cross-Selection (1000, 2000 generation).
3) The number of generations for each cycle before Cross-Selection can be determined by analyzing the convergence of AES (in our case it should be more than 100).
4) The CS+AES method can be used to find the global optimum of a very complex function.

IV. CONCLUSIONS

The our work proposed the Cross-Selection Method, which together with Evolution Strategies, produced a very good result for different test functions. As the most complex test function that checks the capabilities of this method, we took the $20D$ Weierstrass function. Since this function is not smooth and has a huge number of local minima, the vast majority of optimization methods cannot find its global minimum. To solve this problem, we propose our method.

The Cross-Selection Method, like other Evolutionary Algorithms, has a biological basis. We, like biologists-breeders, take a few good individuals with the properties we need and cross them with each other. Any Evolution Strategy can be used to produce good parent -individuals. As such a strategy, we used the Adaptive Evolution Strategy $(10/3 + 20)$, as a result of which we get parents with the properties we need. Further, by crossing several parents, each of which is good in some properties, we can get a child who will take the good properties of each of his parents. This allows you to move to a new search domain. By repeating this procedure several times, we get into the domain of the global minimum.

We believe that the Cross-Selection Method proposed by us can be used for complex multidimensional continuous functions with a large number of local optima, for which other optimizational methods work ineffectively.

References

[1] N. Hansen, D. Arnold, and A. Auger, “Evolution strategies,” in Springer Handbook of Computational Intelligence, 2015, pp. 871–898. [Online]. Available: https://doi.org/10.1007/978-3-662-43505-2_44

[2] D. Simon, Evolutionary optimization algorithms. Biologically-Inspired and Population-Based Approaches to Comp. Intelligence. Wiley, 2020.

[3] I. Rechenberg, Evolutionsstrategie: Optimierung technischer Systeme nach Prinzipien der biologischen Evolution, Dr.-Ing. Thesis. Technical University of Berlin, 1971.

[4] H.-P. Schwefel, Evolutionsstrategie und numerische Optimierung, Dr.-Ing. Thesis. Technical University of Berlin, 1975.

[5] H.-G. Beyer, “Toward a theory of evolution strategies: Some asymptotical results from the (1, $^{+}\lambda$)-theory,” Evol. Comp., vol. 1, no. 2, pp. 165–188, 1993. [Online]. Available: https://doi.org/10.1162/evco.1993.1.2.165

[6] H.-P. Schwefel and G. Rudolph, “Contemporary evolution strategies,” in Advances in Artificial Life, vol. 929, 1995, pp. 891–907. [Online]. Available: https://doi.org/10.1007/3-540-59496-5_351

[7] A. Bienvenüe and O. Francois, “Global convergence for evolution strategies in spherical problems: some simple proofs and difficulties,” Theor. Comp. Sci., vol. 306, no. 1, pp. 269–289, 2003. [Online]. Available: https://doi.org/10.1016/s0304-3975(03)00284-6

[8] D. Ackley, “An empirical study of bit vector function optimization,” in Genetic Algorithms and Simulated Annealing. Pitman Publishing, 1987, pp. 170–215.

[9] J. Liang, P. Suganthan, and K. Deb, “Novel composition test functions for numerical global optimization,” in Swarm Intelligence Symposium, Pasadena, California, 2005, pp. 68–75. [Online]. Available: https://doi.org/10.1109/SIS.2005.1501604

[10] H.-G. Beyer, “Some aspects of the ‘evolution strategy’ for solving tsp-like optimization problems,” in Parallel Problem Solving from Nature, vol. 2, 1992, pp. 361–370.

[11] M. Herdy, “Application of the ‘evolutions strategy’ to discrete optimization problems,” in Parallel Problem Solving from Nature, vol. 496. Springer-Verlag, Berlin, 1990, pp. 188–192.

[12] H.-G. Beyer and H.-P. Schwefel, “Evolution strategies: A comprehensive introduction,” Natur. Comp., vol. 1, no. 1, pp. 3–52, 2002. [Online]. Available: https://doi.org/10.1023/A:1015059928466

[13] C. Kappler, “Are evolutionary algorithms improved by large mutations?” in Parallel Problem Solving from Nature, vol. 1141, 1996, pp. 346–355. [Online]. Available: https://doi.org/10.1007/3-540-61723-x_999

[14] G. Rudolph, “Local convergence rates of simple evolutionary algorithms with cauchy mutations,” IEEE Trans. Evol. Comp., pp. 249–258.

[15] X. Yao and Y. Liu, “Fast evolution strategies,” in Evolutionary Programming VI, 1997, pp. 151–161. [Online]. Available: https://doi.org/10.1007/BFb0014808

[16] X. Yao, Y. Liu, and G. Lin, “Evolutionary programming made faster,” IEEE Trans. Evol. Comp., vol. 3, no. 2, pp. 82–102, 1999. [Online]. Available: https://doi.org/10.1109/4235.771163

[17] D. Wierstra and T. Schaul, “Natural evolution strategies,” J. of Machine Learning Research, vol. 15, pp. 949–980, 2014.

[18] N. Hansen and A. Ostermeier, “Adapting arbitrary normal mutation distributions in evolution strategies: The covariance matrix adaptation,” in IEEE Int. Conf. Evol. Comp., 1996, pp. 312–317. [Online]. Available: https://doi.org/10.1109/ICEC.1996.542381

[19] N. Hansen and A. Ostermeier, “Completely derandomized self-adaptation in evolution strategies,” Evol. Comp., vol. 9, no. 2, pp. 159–195, 2001.

[20] N. Hansen and S. Kern, “Evaluating the cma evolution strategy on multimodal test functions,” in Parallel Problem Solving from Nature, vol. 3242, 2004, pp. 282–291. [Online]. Available: https://doi.org/10.1007/978-3-540-30217-9_29

[21] S. Kern, S. Müller, N. Hansen, D. Büche, J. Ocenasek, and P. Koumoutsakos, “Learning probability distributions in continuous evolutionary algorithms – a comparative review,” Natur. Comp., vol. 3, no. 1, pp. 77–112, 2004. [Online]. Available: https://doi.org/10.1023/B:NACO.0000023416.59689.4e

[22] H.-G. Beyer and K. Deb, “On self-adaptive features in real-parameter evolutionary algorithms,” IEEE Trans. Evol. Comp., vol. 5, no. 3, pp. 250–269, 2001.

[23] I. Rechenberg, “Evolutionsstrategien,” in Simulationsmethoden in der Medizin und Biologie. Springer-Verlag, 1978, pp. 83–114.

[24] M. Mühlenbein and D. Schlierkamp-Voosen, “Predictive models for the breeder genetic algorithm i. continuous parameter optimization,” Evol. Comp., vol. 1, no. 1, pp. 25–49, 1993.

[25] D. Arnold, “Weighted multirecombination evolution strategies,” Theor. Comp. Sci., vol. 361, no. 1, pp. 18–37, 2006.

LARYSA KHILKOVA, PhD, a Docent of the Department of Information Technologies and Programming of V. Dahl East Ukrainian National University. She received a PhD degree in mathematical science in 2018 at Verkin ILTPE of NAS Ukraine and held a postdoctoral position in 2018-2020 at Fraunhofer ITWM (Germany). Her main research interests include homogenization theory and intellectual data analysis.