Maximizing a Nonlinear Function using Evolutionary Strategies and Parameter Tuning

Bc. Klára Murínová
ICSM
FEI of STU in Bratislava
Bratislava, Slovakia
qmurinova@stuba.sk

Bc. Michal Čopjan
ICSM
FEI of STU in Bratislava
Bratislava, Slovakia
xcopjan@stuba.sk

doc. Ing. Kvetoslava Kotuliaková, PhD.
IMICT
FEI of STU in Bratislava
Bratislava, Slovakia
kvetoslava.kotuliakova@stuba.sk

***Abstract*—Maximization and minimization of functions are problems that occur in most scientific fields. There are several methods for solving them and one such method is the evolutionary strategy algorithm, which is an algorithm inspired by the biological theory of evolution. The algorithm works with a set of candidate solutions, and in each iteration, these solutions are crossed and mutated to create new ones. The newly created solutions are then evaluated and the best solutions are moved to the next iteration. This process is repeated until the optimal solution is found or another terminating condition occurs. In this paper, we implemented such an algorithm and used it to maximize the chosen function. We also found the best combination of parameters for which the algorithm gives the best result.**

***Index Terms*—evolution strategy, maximization, optimization.**

I. Introduction

In the wild, there is only one rule: stronger wins. The strength of an individual can be described by its speed, size or even its coloring. These strong individuals have a better chance to reproduce, ensuring that the next generations will be better adapted to the unforgiving environment. But how might we apply this concept to finding the maximum of a nonlinear function? This principle inspired the emergence of evolution algorithms. Stochastic strategies that search for optimal solutions to problems by refining the ”population” of possible solutions from generation to generation. In our case, we focus on the use of evolutionary algorithms to find the maximum of a nonlinear function, where the value of an individual is not determined by its strength or size, but by the result obtained by fitting it to the function. The higher the value, the ”stronger” the individual and the more likely it is to carry its properties to the next generation.

II. Methodology

A. Problem description

Maximization (or minimization) of a function is the problem of finding the parameters for which the function has the largest (or smallest) value. This problem occurs in most scientific fields. For example, in the field of artificial intelligence, the loss function of neural networks is minimized during their training. For testing our implementation we choose the following function:

$$f(x,y) = (0.5+|y|)^{-2} + cos(2\pi xy) + 10(|x+1|+1)^{-1} \quad (1)$$

B. Evolution strategy approach

Evolution strategies consist of key steps crucial to the optimization process, each impacting solution quality and speed. The algorithm follows a cyclic process, shown in figure 1, where steps build on each other and are repeatedly executed in each iteration until the termination criterion is fulfieled [1].

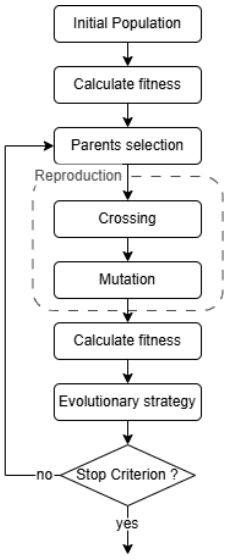

Fig. 1. The flow of an evolution strategy algorithm

1) Generating the initial population P_{init}

The first step is to generate the individuals of the initial population $P_{init} = \mathbf{x}_1, ..., \mathbf{x}_n$. Each individual is represented by a vector of real numbers $\mathbf{x}_i = (x_i, y_i, \sigma_{x_i}, \sigma_{y_i})$, the individual parameters are generated from a predefined range of values [2].

The parameters x and y:

- define the individual's position in the optimized function and affect its quality.

The parameters σ_x and σ_y are:

- non-negative values (dispersions) of a normal distribution that control mutation rate, ensuring population diversity.

Then each individual $\mathbf{x}_i$ is evaluated using the fitness function:

$$F(x_i, y_i) = \frac{(1-\epsilon) \cdot f(x_i, y_i) + f_{\text{max}} \cdot \epsilon - f_{\text{min}}}{f_{\text{max}} - f_{\text{min}}} \tag{2}$$

The fitness function is used to assess each individual x_i on how successful it is in a given environment. The function normalizes the outputs in the interval $(0, 1)$ where values closer to 1 indicate that the individual fits the function better.

- parameter ϵ: minimum fitness assigned even to the worst individual (figure 4), ensuring it stays in the process and supports population diversity.
- $f(x, y)$: evaluates individual performance based on x_i and y_i fitted to the function.
- f_{max} and f_{min}: estimated values of the local maximum and minimum, used for normalization [2].

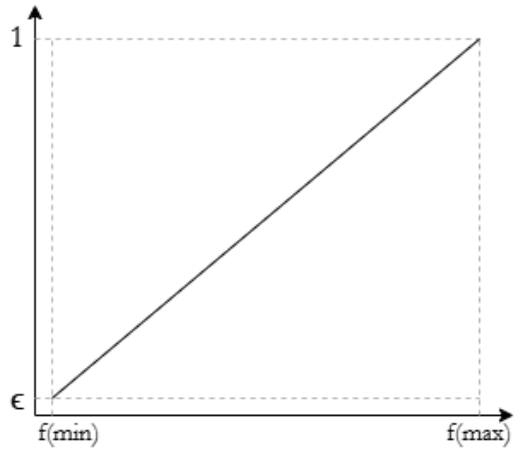

Fig. 2. Fitness function

2) Selection of parents μ for the generation G_i

From the population P, μ parents are selected for the generation $G_i (i = 0, 1, ..., n)$ using the roulette wheel.

It is a probability-based selection method based on fitness. Conceptually, this is equivalent to repeatedly spinning a single-arm roulette wheel, where the sizes of the holes reflect the selection probabilities.

For each individual $\mathbf{x}_i$, its fitness probability is calculated:

$$P(\mathbf{x}_i) = \frac{\text{fitness}(\mathbf{x_i})}{\sum_j \text{fitness}(\mathbf{x_j})} \tag{3}$$

where $\sum_j \text{fitness}(\mathbf{x}_j)$ is the total fitness of all individuals in the population. For selection, the cumulative probability distribution is computed as a list of values: $[a_1, a_2, ..., a_\mu]$, where

$$a_i = \sum_{j=1}^{i} P_{\text{x}_\text{j}} \tag{4}$$

assuming that $a_\mu = 1$.

- P_{x_j} is the selection probability of individual i based on fitness.
- This cumulative distribution covers the interval $[0, 1]$.
- If we randomly generate a number $r \in [0, 1]$, we search for which interval a_i falls into. Individual i is selected if: $a_i - 1 < r <= a_i$

This ensures that fitter individuals have a higher selection chance, but randomness still gives weaker ones a chance, supporting diversity; highly fit individuals can be selected multiple times [2].

3) Reproduction

After the selection of μ parents, reproduction occurs to produce new λ offspring. In evolutionary strategies (ES), reproduction is composed of two main steps: crossover and mutation.

3.1 Crossover

In evolutionary strategies, there are a number of methods to implement crossover, the key is:

- How many parents μ are used to produce one offspring λ_i :
  a) Local (local crossover)
  Each λ_i offspring is created from two parents, denoted e.g. as μ_1 and μ_2, which are selected from the whole set of μ parents using the roulette wheel mechanism (as explained above). It is possible that one parent will be selected twice, so that it will cross with itself.
  b) Global (global crossover)
  One λ_i offspring is created by combining information from all μ parents.

- How their parameters $x_i, y_i, \sigma_{x_i}, \sigma_{y_i}$ are combined, which the offspring will inherit:
  a) Discrete recombination
  Each parameter of the offspring $(x_i, y_i, \sigma_{x_i}, \sigma_{y_i})$ is randomly taken from one of the parents.
  b) Average recombination
  The parameters of the offspring are calculated as the arithmetic mean of the values of all participating parents [2].

3.2 Mutation

Mutation is the second essential step in reproduction and is crucial for maintaining diversity in a population. While selection and crossover exploit known good solutions, mutation enables exploration of new areas, preventing the algorithm from getting stuck in local maxima. Evolutionary strategies mutate both x_i, y_i and dispersions $\sigma_{x_i}, \sigma_{y_i}$, adapting mutation size to the population's distance from the optimum [2].

1) Mutation of dispersions $\sigma_{x_i}, \sigma_{y_i}$

The formula for mutation variance is:

$$\sigma'_{x_i} = \sigma_{x_i} \cdot e^{N(0,\sigma_G)} \tag{5}$$

- σ'_{x_i} - the mutated value
- $N(0, \sigma_G)$ - random value from a normal distribution with mean 0 and global variance σ_G
- The variance (σ_{x_i}) can never be negative - so the exponential form $e^{N(0,\sigma_G)}$ is used, which always returns a positive number
- The value of σ_G (global mutation variance) is mostly set as a constant, ofte $\sigma_G = 1$

2) Mutation of x_i, y_i parameters

When the mutated variance values are ready, then the x_i, y_i parameters themselves are changed according to:

$$x_i = x_i + N(0, \sigma'_{x_i}) \tag{6}$$

A normally distributed random value with standard deviation σ'_{x_i} is added to x_i [2].

4) Evolution Strategies (ES)

After the reproduction phase, when λ offspring are produced, it is the turn of selection, which determines who will advance to the next generation of G_i. Evolution strategies define the set of individuals selected for the new population P. We applied two basic strategies: $ES(\mu, \lambda)$ and $ES(\mu+\lambda)$.

4.1 Non elistic strategy $ES(\mu, \lambda)$

This strategy selects the new population exclusively from the offspring λ, with parents μ discarded. Offspring are ranked by fitness, and the best μ are selected for the next generation. It's important that $\lambda > \mu$ to ensure diversity, which can help avoid local optima and aid in finding the global optimum. This strategy is illustrated in figure 3.

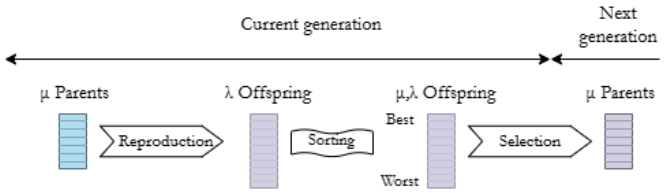

Fig. 3. $ES(\mu, \lambda)$ strategy

4.2 Elitist strategy $ES(\mu + \lambda)$

This elitist strategy forms the next population from the combined pool of parents and offspring $\mu + \lambda$, individuals by fitness are chosen. Preserving high-quality solutions and promoting faster, more stable convergence [3]. This approach is illustrated in figure 4.

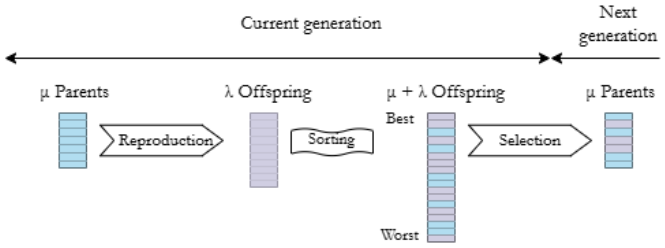

Fig. 4. $ES(\mu + \lambda)$ strategy

C. Parameter tuning

To find the best parameters for the algorithm we used the grid search technique. In this technique, multiple values are chosen for each parameter and then all combinations of these values are tested. For each combination, an average fitness value is calculated and the combination with the best value is declared as the best combination. Since our algorithm is stochastic and thus may not always give the optimal result (for example, it may get stuck at a local maximum), for each combination of parameters we ran the algorithm 10 times and took the average fitness value of the best individual. The parameters that we used and tasted in grid search are listed in table 1.

TABLE I
PARAMETER GRID

Parameter	Values
Number of parents	8, 16, 32
Number of offspring	16, 32, 64
Number of generations	1000
Estimate function range	(-1, 15)
Init x range	(-10, 10), (-100, 100)
Init y range	(-10, 10), (-100, 100)
Init σ_x range	(0.01, 0.1), (0.1, 0.15), (0.3, 1)
Init σ_y range	(0.01, 0.1), (0.1, 0.15), (0.3, 1)
σ_G	1
ϵ	0.01, 0.05, 0.1, 0.3
Evolution strategy	$ES(\mu, \lambda)$, $ES(\mu + \lambda)$
Crossover strategy	DISCRETE, AVERAGE
Crossover scope	LOCAL, GLOBAL

III. Results

After running the grid search over the defined parameter space, we tested 7666 parameter combinations. Each combination was run 10 times, and we used the average fitness of the best individual as the final score.

A fitness of 0 appeared in 432 combinations, all due to invalid configurations where the non-estimating strategy ($ES(\mu, \lambda)$) didn't meet the required $\lambda > \mu$. These were excluded from further analysis.

TABLE II
Simplified Distribution of Fitness Scores Across Parameter Combinations

Fitness Interval	Number of Combinations
0.0	432
(0.0 – 0.4)	449
(0.4 – 0.6)	270
(0.6 – 0.8)	1817
(0.8 – 0.9)	2127
(0.9 – 1.0)	3113
1.0	1516

We focused on the 1516 combinations with a fitness of exactly 1.0 to identify the importance of each parameter and its best values (figure 5)

From the resulting data, we observed the following trends:

- Number of parents: better results were achieved with more parents.
- Number of offspring: higher performance was observed with more offspring.
- Initial y range: the narrower range performed better.
- Epsilon (mutation rate): the moderate value showed slightly better results.
- Selection strategy: better results were achieved using parents and offspring.
- Crossover strategy: discrete crossover performed better.

The following parameters showed almost no significant difference in performance across their values: initial x range, initial sigma values (x and y) and crossover scope.

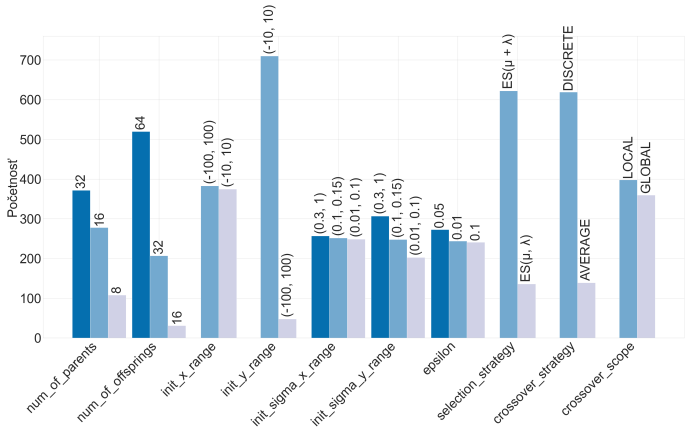

Fig. 5. Importance of parameters and best values

We ran the algorithm one more time with the best found parameters, which are listed in Table 3. The average fitness over generations is shown in Figure 6. We then plugged the x (-1) and y (approximately 0) of the resulting individual into our function and its value was 15 (formula 7).

TABLE III
Best Parameters

Parameter	Values
Number of parents	32
Number of offspring	64
Number of generations	1000
Estimate function range	(-1, 15)
Init x range	(-100, 100)
Init y range	(-10, 10)
Init σ_x range	(0.3, 1)
Init σ_y range	(0.3, 1)
σ_G	1
ϵ	0.05
Evolution strategy	$ES(\mu + \lambda)$
Crossover strategy	DISCRETE
Crossover scope	LOCAL

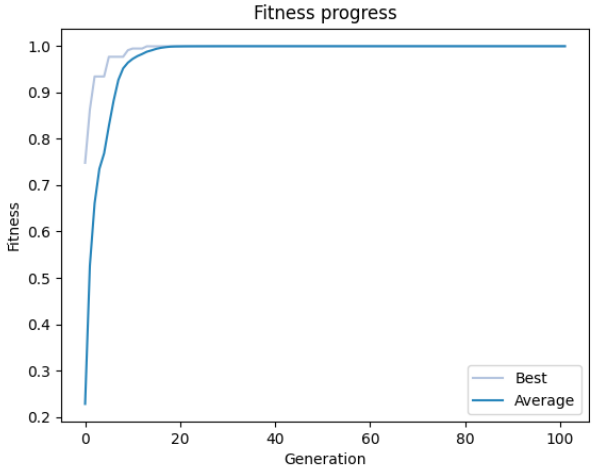

Fig. 6. Diagram od $ES(\mu, \lambda)$

$$f(-1,0) = (0.5 + |0|)^{-2} + \cos(2\pi - 1 \times 0) + 10(|-1+1|+1)^{-1} = 15 \quad (7)$$

IV. Discussion

Based on our research, we confirmed that, as we initially expected, key parameters like the evolution strategy selection have a major impact on maximizing performance. Specifically, the ES($\mu + \lambda$) strategy proved more effective than other variants.

We also confirmed that the function depends more heavily on the y-values, making the y-range parameter highly important, while the x-range showed little influence, just as we had predicted.

On the other hand, we were surprised by the importance of the number of parents and offspring, higher values clearly led

to better results. In contrast, parameters like sigma-x, sigma-y, and various epsilon (mutation) settings showed almost no significant effect on performance.

When we plugged the optimized x and y values into the objective function, we achieved results very close to our initially estimated global maximum.

Overall, we consider our research a success.

V. Conclusion

We successfully implemented the evolution strategy algorithm and found the best combinations of its parameters. From the results it is clear that the parameters that made the most difference are evolution strategy, crossover strategy and initial y range. Among the parameter combinations for which the average fitness of the best individual was 1, the majority use the $ES(\mu + \lambda)$ evolution strategy, the average crossover strategy, and a (-10, 10) range for y. Using the best parameters, we get the global maximum of our function, with the value of 15. Overall, by performing the series of experiments, we showed the power of ES in finding extremes of complicated functions and documented the effects of algorithm's parameters as well.

VI. Acknowledgement

The author hereby declare that the article was founded by grand ERASMUS-EDU-2023-CBHE-STRAND-2, ID: 101129022

References

[1] DEAP Developers (n.d.) ES Function Minimization Example. DEAP Documentation.
[2] A.E. Eiben, J.E. Smith (2015) Introduction to Evolutionary Computing. Springer.
[3] Hans-Georg Beyer (2007) Evolution strategies. Scholarpedia, 2(8):1965.

This page is intentionally left blank.