Research Inventy: International Journal of Engineering and Science
Vol.16, Issue 3 (March 2026), PP 01-08
Issn (e): 2278-4721, Issn (p):2319-6483, www.researchinventy.com

Comparative Analysis Between Optimization Methods

Daniel Meira Santos Souza[1], Gustavo Aparecido Pita Baggio[2], José Carlos Camargo[3]

[1,2] Discente na Universidade Estadual de Santa Cruz, Rodovia Jorge Amado, Km 16, Ilhéus-BA, Brasil.
[3]Docente na Universidade Estadual de Santa Cruz, Rodovia Jorge Amado, Km 16, Ilhéus-BA, Brasil.
Corresponding Author: dmssouza.ppgmc@uesc.br

Abstract: *This work performs a comparative analysis between three global optimization methods—Luus-Jaakola, Differential Evolution, and Firefly—applied to four benchmark functions with different levels of complexity and multimodality: a unimodal polynomial function, Rastrigin, Six-Hump Camelback, and Goldstein-Price. Computational routines were developed in Python to implement each algorithm, evaluating their performance in terms of accuracy, robustness, and computational efficiency. The results indicate that the Luus-Jaakola method showed rapid convergence and high accuracy in all tested cases, standing out for its simplicity of implementation. Differential Evolution demonstrated excellent stability and global accuracy, with a strong ability to avoid local minima even in highly multimodal functions. The Firefly method exhibited good overall performance, but showed greater sensitivity to control parameters and specific difficulty in the Rastrigin function, where it converged to a local minimum. The comparison highlights the advantages and limitations of each approach, providing support for the appropriate choice of optimization methods in engineering applications involving complex surfaces and inverse problems.*

Keywords: *Optimization, Luus-Jaakola, Differencial Evolution, Firefly.*

Date Of Submission: 15-03-2026 Date of Acceptance: 31-03-2026

I. INTRODUCTION

Global optimization is an essential tool in various areas of engineering, particularly in inverse problems, where the goal is to estimate parameters or initial conditions from observed data, often characterized by multimodal and non-convex functions. These problems arise in contexts such as modeling dynamic systems, identifying materials, and controlling processes, demanding efficient algorithms that balance computational cost and convergence rate (Silva Neto et al., 2016) (Silva Neto e Moura Neto, 2005).

To evaluate the performance of these methods in terms of computational cost (measured by execution time and number of function evaluations) and convergence (precision and robustness in reaching the global optimum), we applied them to four benchmark functions representative of optimization challenges. The first is a simple fourth-order polynomial function that serves as a unimodal base case. Next, the Rastrigin function, a multimodal function with many local minima. The Six-hump Camelback function, with six local minima (two global), and finally, the Goldstein-Price function, another multimodal function with four local minima. These equations are presented later (Liu et al., 2024) (Jamil and Yang, 2013).

To perform the analysis of the four previous equations, computational routines were constructed to calculate the global minima of the functions through optimization methods. Three methods will be used: The Luus-Jaakola method, the Differential Evolution method, and finally the Firefly method.

The Luus-Jaakola algorithm is classified as a random search method used in optimization problems. This method is based exclusively on the values of the objective function, without requiring the calculation of derivatives or other additional information. Originally proposed in 1973, the procedure starts from a wide initial region in the domain of the function and, at each iteration, generates random solutions within this region. As the process progresses, the search space is gradually reduced, concentrating around the best solutions found. When a new solution surpasses the previous one, the value of the corresponding objective function is recalculated and compared to a predefined tolerance criterion. If this condition is met, the algorithm is terminated, and the point obtained is considered the optimum of the function (Silva, 2019).

The Differential Evolution algorithm is a stochastic optimization method inspired by natural evolutionary processes. Like the Luus–Jaakola method, it seeks optimal solutions without needing derivatives of the objective function, relying solely on the direct evaluation of the obtained values. Originally proposed in 1995 with the intention of solving Chebyshev's polynomial fitting problem, the method operates on a population of candidate solutions, which is iteratively improved through three fundamental operators: mutation, responsible for introducing perturbations in the population vectors; crossover, which combines information between different solutions; and selection, which preserves the best alternatives in each iteration. The evolutionary process is driven by parameters such as population size, mutation factor, and crossover rate, which control the exploration and intensification of the search in the solution space. The algorithm terminates when the stopping criterion is reached, considering the best solution as the optimal point found (Storn and Price, 1997).

The Firefly algorithm is an optimization method inspired by the light attraction behavior observed in fireflies. Similar to the other algorithms presented, it seeks the optimum of an objective function through the interaction between several candidate solutions. In this method, the attractiveness between fireflies is determined by the intensity of the brightness associated with the quality of the solution and the distance between individuals. Thus, less bright fireflies move towards the brighter ones, gradually concentrating the search in the most promising regions of the solution space. Parameters such as population size, attractiveness coefficient, and light absorption coefficient are adjusted to control the swarm dynamics. The process is repeated until the specified tolerance criterion is met, and the best solution obtained is considered the optimum point of the problem (Yang, 2009).

Given the above, the main objective of this work is to identify the advantages and limitations of each method in these functions, with a view to application in inverse engineering problems, such as seismic data inversion or parameter optimization in numerical simulations. The results can guide the selection of algorithms for real-world scenarios, where the balance between efficiency and accuracy is critical.

II. EXPERIMENTAL PROCEDURE

First, let's look at the equations that are the focus of this work. The equations mentioned earlier are:

- **Function 1(Polynomial):** $\boldsymbol{S(x_1,x_2) = x_1^4 + x_2^4 - 3}$
- **Function 2 (*Rastrigin's*):** $\boldsymbol{S(x_1,x_2) = 20 + \left(x_1^2 - 10cos(2\pi x_1)\right) + \left(x_2^2 - 10cos(2\pi x_2)\right)}$
- **Function 3 (*Six-hump camel back*):** $\boldsymbol{S(x_1,x_2) = \left(4 - 2,1x_1^2 + \frac{x_1^4}{3}\right)x_1^2 + x_1x_2 + \left(4x_2^2 - 4\right)x_2^2}$
- **Function 4 (*Goldstein-Price*):** $\boldsymbol{S(x_1,x_2) = S_1(x_1,x_2).S_2(x_1,x_2)}$, where:
  - $\boldsymbol{S_1(x_1,x_2) = 1 + (x_1 + x_2 + 1)^2\left(19 - 14x_1 + 3x_1^2 - 14x_2 + 6x_1x_2 + 3x_2^2\right)}$
  - $\boldsymbol{S_2(x_1,x_2) = 30 + (2x_1 - 3x_2)^2\left(18 - 32x_1 + 12x_1^2 + 48x_2 - 36x_1x_2 + 27x_2^2\right)}$

Function 1 presents no difficulty in evaluating its minimum value. Because it has even powers, it is easy to see that its minimum value will be (-3) when the coordinates x1, x2 have the value (0,0). The other target equations are not so simple to deduce, but the minimum functions are known. Below is Table 1 with the coordinate values where the function has a minimum and the value it assumes at the minimum point.

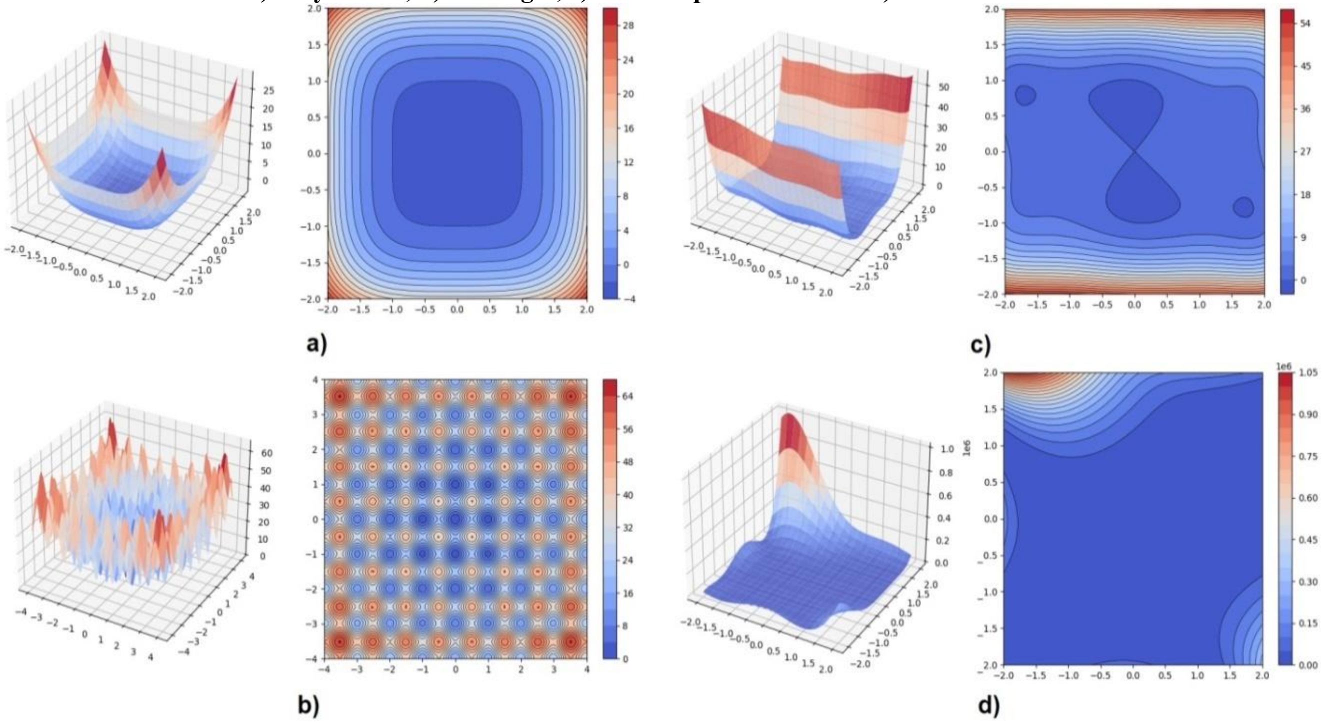
Table 1: Minimum values and their coordinates.

Function name	# Function	x_1	x_2	$S(x_1, x_2)$
Polynomial	1	0	0	-3
Rastrigin's function	2	0	0	0
Six-hump camel back	3	±0.0898	±0.7126	-1.0316
Goldstein-Price's function	4	0	-1	3

The first function will be of great importance for testing the algorithms of the optimization methods. The other functions present characteristics that can easily deviate the convergence of these methods from the correct point, hence the reason for choosing these functions. These and other similar equations can be found in Silva Neto et al., (2016) and are excellent tools for performing optimization tests.

To better understand the challenge posed by these functions, Figure 1 shows the equations graphically along with their topology. This makes it easy to see how these functions can cause the methods to diverge from the desired minimum coordinate.

Figure 1: Graph and Topology of the surfaces of interest.
a) Polynomial, b) Rastrigin, c) Six-hump Camelback e d) Goldstein-Price



The development of the computational routines for the three methods in the Python language then begins. In general, the methods present a certain simplicity in the construction of the code. The pseudocodes of the three methods can be found in Silva Neto et al., (2016). All three methods have global optimization algorithms based on point search (metaheuristics), that is, they work with individual solutions (points in the search space) and use mechanisms (such as mutation, crossover, attraction) to move or generate new points, iteratively improving the quality of these solutions, and which operate in a continuous or discrete search space, evaluating specific points (candidate solutions) in this space. Based on the knowledge of the analytical values of the functions, Table 2 presents the values of the search space applied to each of the functions.

Tabela 2: Search area for each function.

Function #	Domain de x_1	Domain of x_2
1	$-2 \le x_1 \le 2$	$-2 \le x_2 \le 2$
2	$-4 \le x_1 \le 4$	$-4 \le x_2 \le 4$
3	$-2 \le x_1 \le 2$	$-2 \le x_2 \le 2$
4	$-2 \le x_1 \le 2$	$-2 \le x_2 \le 2$

III. RESULTS AND DISCUSSIONS

This chapter presents the numerical results obtained from the implementations of the three proposed optimization methods: Luus-Jaakola, Differential Evolution, and Firefly. Each algorithm was applied to the four test functions described previously, and their performance was evaluated by comparing the minima found with known analytical solutions. The analysis aims to highlight the efficiency of each method, and after conducting the initial tests, it is observed that the Rastrigin function (Function 2) did indeed show variations in the result values according to the variation of the parameters belonging to each method, especially in the Firefly method. Therefore, the results will be presented below, divided by algorithm.

3.1 Luus-Jaakola Algorithm

The Luus-Jaakola algorithm presents as main parameters of the method the maximum number of iterations, the number of internal trials, and the reduction factor (α) of the search region, which were adjusted to balance the exploration of local convergence (Silva, 2019). To evaluate the influence of these parameters on the method's performance, experiments were carried out varying the reduction factor α between 0.1 and 1.0, keeping the other parameters fixed at 1000 iterations and 100 internal trials.

Table 3: Luus-Jaakola results.

Function	Iterations	x_1	x_2	$S(x_1, x_2)$
1	22	$-1.5x10^{-5}$	$-6.1x10^{-5}$	-3.0
2	56	0.0	0.0	0.0
3	53	0.0898	-0.7126	-1.0316
4	48	0.0	-1.0	3.0

Data analysis reveals efficient and consistent performance. The method was able to locate the global minimum in all tested functions, exhibiting residual error close to zero. The number of iterations required for convergence ranged from 40 to 200, a value significantly lower than the maximum limit of 1000 iterations, indicating that the stopping criterion based on the reduction of the search region was reached ahead of schedule.

The tests were applied to Function 2, chosen because it showed greater sensitivity to changes in the parameters, allowing for a clearer observation of the algorithm's convergence behavior. Table 4 summarizes the five best results obtained specifically for Function 2, where the most sensitive behavior to adjustments of α was observed.

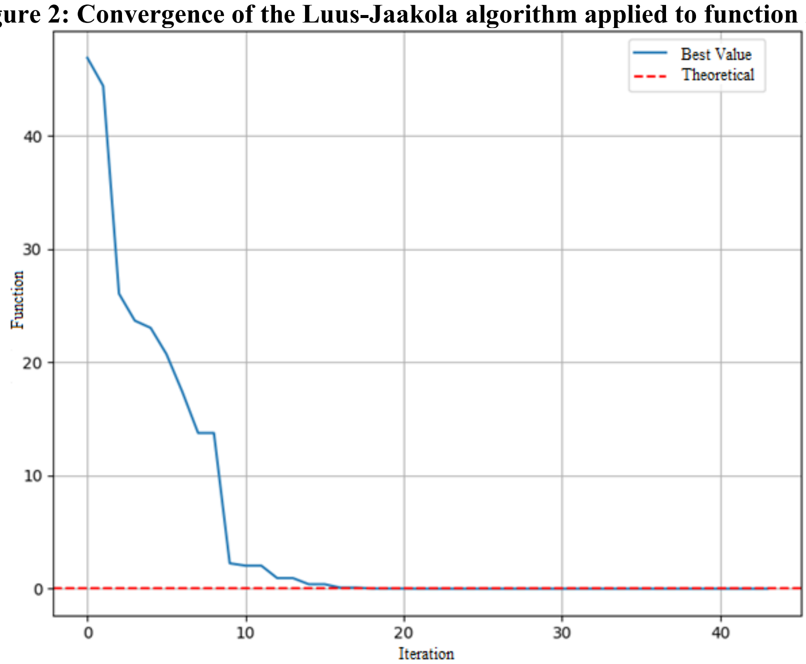
Table 4: Luus-Jaakola results for function 2.

α	Iterations	x_1	x_2	$S(x_1, x_2)$	Valor real	Erro Abs.
0.4	43	0.0	0.0	0.0	0.0	0.0
0.6	59	0.0	0.0	0.0	0.0	0.0
0.7	78	0.0	0.0	0.0	0.0	0.0
0.8	112	0.0	0.0	0.0	0.0	0.0
0.9	198	0.0	0.0	0.0	0.0	0.0

It is noted that the smaller the reduction factor, the faster the algorithm converges, but with less global exploration capacity. On the other hand, higher values of α (such as 0.8 and 0.9) result in slower convergence, as the search region is reduced more gradually. The absolute error remained practically zero in all executions, demonstrating the robustness of the method in determining the optimum point, regardless of the rate of contraction of the search region. The increase in the number of iterations as α grows reflects the expected behavior: the smaller the reduction rate, the faster the algorithm converges, but with the risk of premature stabilization at local minima. Figure 1 shows the convergence curve of the Luus-Jaakola algorithm, graphically illustrating the variation in the value of the objective function as a function of the iterations.

Therefore, it can be concluded that adjusting the reduction factor α plays a decisive role in the method's performance, allowing direct control of the exploration and refinement rate of the search. Detailed analysis shows that values of α = 0.4 provide the best result between execution time and final precision for this function.

Figure 2: Convergence of the Luus-Jaakola algorithm applied to function 2 for α = 0.4.



3.2 Differential Evolution Algorithm

Similarly, the Differential Evolution algorithm also has its 3 main parameters that govern its dynamics: mutation, responsible for introducing variations in the population vectors; crossover, which combines information between different solutions; and selection, which preserves the best candidates at each iteration (Storn and Price, 1997).

For the initial configuration, fixed parameters were adopted for the population size with 50 individuals, for the mutation factor (F) of 0.8, for the crossover rate (CR) of 0.9 and 100 iterations, which guide the evolutionary process towards the optimal solution. Table 5 presents the results obtained after applying the method to the four test functions.

Tabela 5: Results of Differential Evolution.

Function	x_1	x_2	$S(x_1, x_2)$
1	6.9×10^{-5}	-3.0×10^{-6}	-3.0
2	0.0	0.0	0.0
3	0.0898	-0.7126	-1.0316
4	0.0	-1.0	3.0

Differential evolution demonstrated high acceptable accuracy in all four functions tested. In particular, for function 2, which exhibits multimodal behavior and high sensitivity to small variations in parameters, the algorithm was able to locate the global minimum with virtually zero residual error. The good performance is due to the population nature of the method, which preserves the diversity of solutions across generations and thus reduces the risk of premature convergence to local optima.

In order to investigate the impact of the control parameters, specific experiments were conducted on function 2, varying the value of the crossover rate (CR) and the Mutation factor (F), while the other parameters were kept constant: a maximum of 100 iterations and a population size of 50. The results obtained are summarized in Table 6.

Table 6: Results of Differential Evolution for the function 2.

CR	F	x_1	x_2	$S(x_1, x_2)$	Real value	Abs. error
0.0	0.1	0.0	0.0	0.0	0.0	0.0
0.1	0.1	0.0	0.0	0.0	0.0	0.0
0.2	0.1	0.0	0.0	0.0	0.0	0.0
0.3	0.1	0.0	0.0	0.0	0.0	0.0
0.4	0.1	0.0	0.0	0.0	0.0	0.0

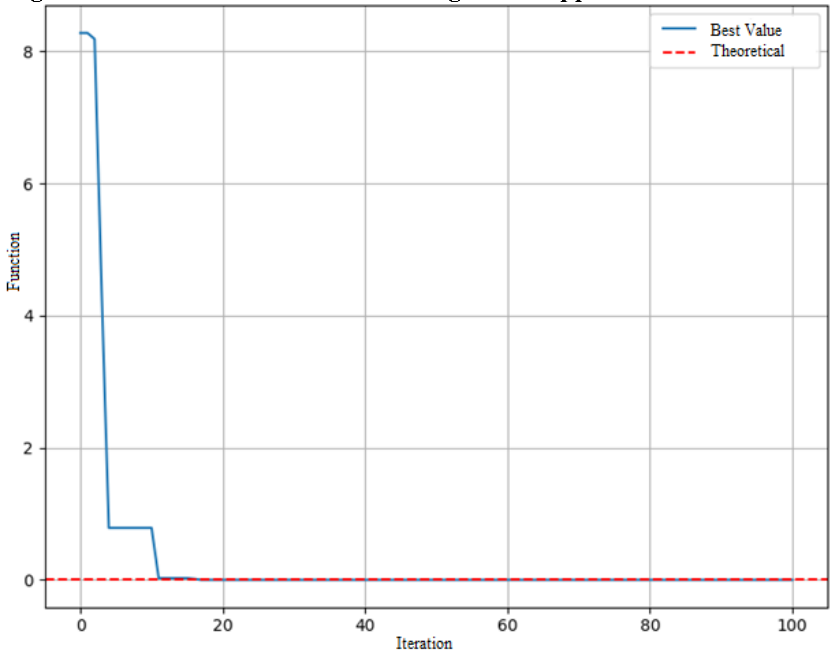
In every test, the algorithm reached the theoretical optimum point (0.0) with zero absolute error, demonstrating stability and accuracy independent of the crossover rate used. However, it is observed that small variations in CR slightly alter the convergence dynamics: lower CR values (such as 0.0 and 0.1) tend to preserve the structure of individuals more, resulting in more exploratory behavior, while intermediate values (such as 0.3 and 0.4) promote slightly faster convergence to the global optimum.

Complete convergence was achieved before the 100-iteration limit in all cases, confirming the efficiency of the evolutionary process. This result indicates that, for function 2, the set of parameters used offers an adequate balance between population diversity and the speed of solution refinement.

Figure 2 illustrates the convergence curve of the Differential Evolution algorithm applied to function 2. A rapid reduction in the objective function value is observed in the first iterations, followed by smooth stabilization around the theoretical minimum value.

Therefore, it can be concluded that the Differential Evolution algorithm showed robust and consistent performance, even under parameter variations. The analysis confirms that, for multimodal problems, its approach offers excellent exploration capabilities and precision in identifying the global optimum.

Figure 3: Convergence of the Differential Evolution algorithm applied to function 2 for *CR*=0.2 e *F*=0.1.



3.3 Firefly algorithm

The third method analyzed has a mechanism based on simulating the social attraction behavior of fireflies. In this model, the attractiveness between individuals is determined by the intensity of brightness and the distance between them. Each firefly is attracted to others that are brighter, and this intensity decreases with increasing distance and the light absorption coefficient (γ). The parameter β_0 defines the maximum attractiveness (when the distance is zero), while α controls the degree of randomness in the movement of individuals, allowing them to escape minimum locals (Yang, 2009) (Silva Neto, et al. 2016).

The configuration parameters adopted were: population size, randomness coefficient (α), attractiveness coefficient (β_0), and light absorption coefficient (γ). Table 7 presents the results obtained after applying the method to the four test functions.

Table 7: Firefly results.

Function	Iterations	x_1	x_2	$S(x_1, x_2)$
1	50	$4.9\text{x}10^{-5}$	$-3.5\text{x}10^{-5}$	-3.0
2	56	0.0	0.0	0.0
3	53	0.0898	-0.7126	-1.0316
4	48	0.0	-1.0	3.0

Analysis of the results reveals satisfactory performance, with the method successfully converging to the global minimum in functions 1, 3, and 4, obtaining highly accurate solutions. However, a critical point of analysis arises in Function 2, where the algorithm converged to a local optimum different from the correct value, failing to reach the known global minimum (S(x) = 0). This function is known for its high multimodality, containing numerous local minima that challenge methods based on deterministic attraction.

In order to investigate the algorithm's behavior more deeply, specific tests were performed for Function 2, varying the parameters α, β_0, and γ. The population size was kept at 10 individuals and the maximum number of iterations at 50. The results obtained are summarized in Table 8.

Based on the results, it is noted that the algorithm presented excellent numerical stability, with an absolute error always less than $3.3\times10{-}4$. Small variations in the control parameters mainly influenced the convergence rate and the final accuracy. When α = 0 (absence of random movement), the method showed more deterministic behavior, converging quickly, but with less ability to escape local minima. Increasing α promoted a more extensive exploration, albeit at the cost of a slight increase in the final error.

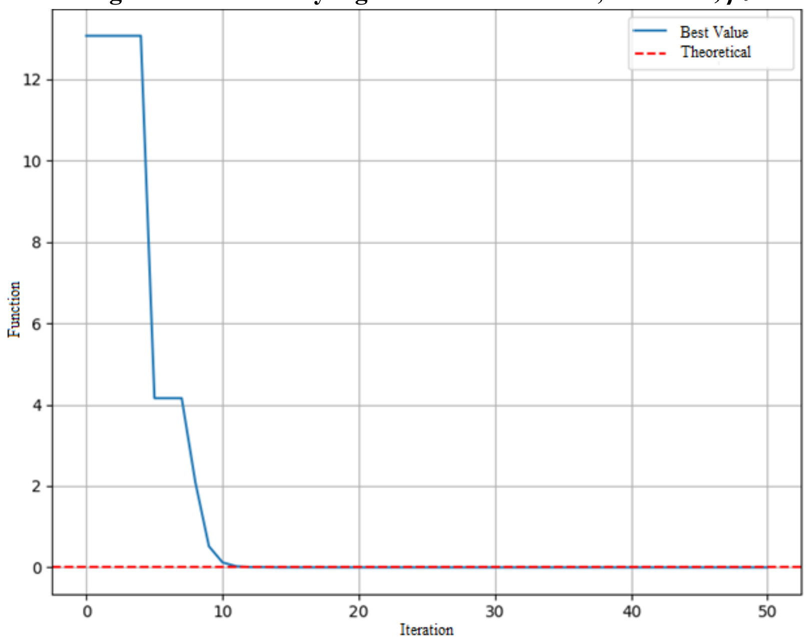
Table 8: Firefly results for function 2.

α	β_0	γ	x_1	x_2	$S(x_1, x_2)$	Real value	Abs. error
0.0	0.4	0.3	$8.1\text{x}10^{-5}$	$1.78\text{x}10^{-4}$	0.0	0.0	0.0
0.1	0.8	0.0	$-9.87\text{x}10^{-4}$	$3.95\text{x}10^{-4}$	$2.24\text{x}10^{-4}$	0.0	$2.24\text{x}10^{-4}$
0.2	0.9	0.3	$5.7\text{x}10^{-4}$	$-8.97\text{x}10^{-4}$	$2.24\text{x}10^{-4}$	0.0	$2.24\text{x}10^{-4}$
0.2	0.5	0.0	$7.11\text{x}10^{-4}$	$8.49\text{x}10^{-4}$	$2.43\text{x}10^{-4}$	0.0	$2.43\text{x}10^{-4}$
0.1	0.3	0.5	$-1.536\text{x}10^{-3}$	$-1.5\text{x}10^{-5}$	$4.68\text{x}10^{-4}$	0.0	$4.68\text{x}10^{-4}$

The parameter β_0, which defines the initial attraction intensity, had a direct impact on the speed at which fireflies gathered: high values accelerate convergence, while lower values allow for a more distributed search. The coefficient γ acted as a stability control factor, modulating the influence of distance on attractiveness, with moderate values of γ (around 0.1 – 0.2) proving ideal for balancing local exploration.

Figure 4 shows the convergence curve of the Firefly algorithm for function 2, highlighting the rapid reduction in the function's value in the first iterations and a gradual stabilization. The slight oscillation near the final convergence indicates the effect of residual randomness ($\alpha > 0$), which helps to avoid premature stagnation.

Figure 4: Convergence of the Firefly algorithm to function 2, for α=0.0, β_0=0.4 e γ=0,3.



In this way, the Firefly algorithm demonstrated solid performance and good overall accuracy, but with considerable sensitivity to the choice of control parameters. The analysis shows that the combination α=0.0, β0 =0.4 and γ=0.3 produced the best balance between stability and accuracy. Although it did not reach the global minimum in function 2 in the overall scenario, the method proved to be efficient in less multimodal search surfaces.

IV. CONCLUSION

Once the correct results were provided, a comparative analysis of the results of each method with the actual values was promptly performed. And here we can affirm that the code implementations were successful, as they all converged to the correct result.

The Luus-Jaakola method proved to be an excellent option due to its efficiency and speed in finding the result, even in "challenging" functions. The method does not present a great challenge in understanding and implementation.

The Differential Evolution method showed excellent accuracy in the results of all functions. The method requires more time compared to Luus-Jaakola and, like it, presents an easy understanding of its structure and operation, making it the "easiest" to implement.

Finally, the Firefly method was undoubtedly the most challenging method to implement. This method presented a longer execution time than the others, which was already expected due to its proposed operation. It demonstrated excellent accuracy, but it was the only one that was "fooled" by the Rastrigin function (Function 2), a fact that in no way diminishes the merit of this method. This problem is easily overcome by increasing the number of individuals and the number of iterations, as shown in Table 7.

Conflict of interest
There is no conflict to disclose.

ACKNOWLEDGEMENT

The authors are grateful to the "Coordenação de Aperfeiçoamento de Pessoal de Nível Superior – CAPES”

REFERENCES

[1]. Jamil, M., Yang, X-S. (2013). A literature survey of benchmark functions for global optimisation problems. Int. J. Mathematical Modelling and Numerical Optimisation, Vol. 4, No. 2, pp.150–194.
[2]. Liu, J., Hong, H., Fan, W., Chen, Y. (2024), A novel fast and efficient adaptive shuffled complex evolution algorithm for model parameter calibration. Frontiers in Environmental Science.
[3]. Silva, A. T. (2019). Análise dos Métodos Luus-Jaakola e Algoritmo de Colisão de Partículas na Estimativa de Parâmetros de um Problema de Transferência de Calor. Dissertação. Universidade Federal Fluminense. Volta Redonda.
[4]. Silva Neto, A. J.; Becceneri, J. C.; Campos Velho, H. F. (2016). Inteligência Computacional Aplicada a Problemas Inversos em Transferência Radiativa. EdUERJ, Rio de Janeiro.
[5]. Silva Neto, A. J., Moura Neto, F. D. (2005). Problemas Inversos - Conceitos e Aplicações. EdUERJ, Rio de Janeiro.
[6]. Storn, R., Price, K. (1997). Differential evolution - A simple and efficient heuristic for global optimization over continuous spaces. J. Global Optimization, v. 11, pp. 341–359.
[7]. Yang, X. (2009). Firefly algorithms for multimodal optimization, em: Stochastic Algorithms: Foundations and Applications, SAGA 2009, Lecture Notes in Computer Sciences, Vol. 5792, pp. 169-178 (2009).