Dicle University
Journal of Engineering
https://dergipark.org.tr/tr/pub/dumf
duje.dicle.edu.tr

Research Article

Investigation of Slime Mould Algorithm and Hybrid Slime Mould Algorithms' Performance in Global Optimization Problems

Osman ALTAY[1], Elif VAROL ALTAY[2*]

[1]Manisa Celal Bayar University, Department of Software Engineering, Manisa, osman.altay@cbu.edu.tr, ORCID: 0000-0003-3989-2432
[2]Manisa Celal Bayar University, Department of Software Engineering, Manisa, elif.altay@cbu.edu.tr, ORCID: 0000-0001-8087-2754

ARTICLE INFO

Article history:

Received 19 September 2022
Received in revised form 4 December 2022
Accepted 5 December 2022
Available online 31 December 2022

Keywords:

Slime mould algorithm, leader SMA, equilibrium optimizer SMA, CEC2020

Doi: 10.24012/dumf.1177288

* Corresponding author

ABSTRACT

The Slime mould algorithm (SMA) is a relatively new metaheuristic technique that was just presented. While the performance of the newly proposed algorithms gives satisfactory results in optimization problems, combining a recently proposed algorithm with the components of different algorithms improves the performance of SMAs. In recent years, leader SMA (LSMA) and equilibrium optimizer SMA (ESMA) methods, in which SMA is combined with different algorithms, have been proposed. The advantages of the two proposed methods over SMA in different problems are shown. In this study, in order to eliminate the disadvantages of SMA, such as slow convergence rate and local optimum, the performances of the CEC2020 test functions were investigated together with the LSMA and ESMA methods proposed in recent years. The results obtained are statistically analyzed and given in detail in the study.

Introduction

Metaheuristic algorithms have gained unexpectedly widespread popularity in recent years. Their proficiency in tackling several optimization challenges has resulted in this development [1]. Among the popular metaheuristic optimization algorithms in the literature are particle swarm optimization, genetic algorithms, differential evolution algorithms, and ant colony algorithms, as well as algorithms such as Grey Wolf Optimizer [2], Equilibrium Optimizer [3], Archimedes Optimization Algorithm [4], Spotted Hyena Optimizer [5], Aquila Optimizer [6], and Slime Mold Optimization Algorithm (SMA) [7], which were proposed in recent years. While each metaheuristic algorithm has distinct benefits, no method, according to the no-free lunch theorem, can handle all optimization problems. The performance of a metaheuristic algorithm is largely determined by its capacity for exploration and exploitation [8]. As a result, numerous scholars are continually proposing new algorithms and improving upon the original method. However, while having various appealing properties, it has been noted that these algorithms do not always perform as expected. The effectiveness of most metaheuristic optimization algorithms is dependent on the balance of two opposing aims, exploration and exploitation [9]. It is also called exploration and exploitation, diversification and intensification. Exploration guarantees that all areas of the solution domain are sufficiently investigated to provide an approximation of the global optimal solution. Exploitation directs the search effort toward the most effective solutions that have been found up to this point by exploring the environment for further options that are more effective. These two objectives are addressed by search algorithms that use local search techniques, global search approaches, or a combination of both local and global searches: these algorithms are frequently referred to as hybridization [10].

Hybridization may take place in a variety of ways, including the following:

- Starting the algorithm with one method and then applying the second technique to the final population generated with the first technique,
- Merging the approach's distinctive operators into the other technique,
- Using local search to enhance the answer identified by global search, and so on.

The main motivation for the paper is to examine the performance of SMA and different hybrid SMAs in global

optimization problems. For this purpose, the leader SMA (LSMA) [11] and equilibrium SMA (ESMA) [12] methods suggested by Naik et al. were chosen. The performance of these three different methods has been examined in the current and widely used CEC2020 test suite. The CEC2020 benchmark problems consist of 10 different optimization problems. These are unimodal functions, multimodal functions, hybrid functions, and composition functions. Furthermore, the performance of these methods was examined using different dimension values, and detailed analyses were carried out. Thus, the different capabilities of the methods obtained as a result of hybridization of a current optimization algorithm were compared with each other and with the original method in different types of problems, and a detailed examination was provided.

The remainder of the paper is organized as follows: To begin with, Section 2 provides an overview of SMA, LSMA, and ESMA. Section 3 describes ten distinct functions drawn from the CEC2020 test functions. Section 4 contains the experimental findings for the test functions. Finally, in Section 5, conclusions are stated and recommendations for further study are made.

Slime Mould Algorithm

In this section, SMA, and hybrid versions of SMA, LSMA, and ESMA, are explained and their mathematical expressions are given.

Original Slime Mould Algorithm

The mathematical notation of SMA consists of three steps. These are approach food, wrap food, and grabble food. In this section, the mathematical structure of SMA is briefly explained [13].

Approach Food: To describe slime mould's approaching behavior as a mathematical equation, the following contraction rule is proposed:

$$\overrightarrow{X(t+1)} = \begin{cases} \overrightarrow{X_b(t)} + \overrightarrow{vb} \cdot \left(\overrightarrow{W} \cdot \overrightarrow{X_A(t)} - \overrightarrow{X_B(t)}\right), r < p \\ \overrightarrow{vc} \cdot \overrightarrow{X(t)}, r \geq p \end{cases} \quad (1)$$

where $\overrightarrow{vb}$ is a $[-a, a]$ parameter, $\overrightarrow{vc}$ decreases linearly from 1 to 0. t indicates the current iteration, $\overrightarrow{X_b}$ denotes the region with the highest concentration of odor, $\vec{X}$ denotes slime mould position, $\overrightarrow{X_A}$ and $\overrightarrow{X_B}$ represent two randomly chosen swarm members, and $\overrightarrow{W}$ represents slime mould weight.

The following is the formula for the variable p:

$$p = \tanh|S(i) - DF| \quad (2)$$

where $i \in 1,2,\dots,n$, $S(i)$ is the $\vec{X}$'s fitness, and DF is the best fitness in all iterations.

$\overrightarrow{vb}$ is given below:

$$\overrightarrow{vb} = [-a, a] \quad (3)$$

$$a = \operatorname{arctanh}\left(-\left(\frac{t}{\max\_t}\right) + 1\right) \quad (4)$$

$\overrightarrow{W}$ formula is given below:

$$\overrightarrow{W(SmellIndex(\iota))} = \begin{cases} 1 + r \cdot log\left(\frac{bF - S(i)}{bF - wF} + 1\right), condition \\ 1 - r \cdot log\left(\frac{bF - S(i)}{bF - wF} + 1\right), \ others \end{cases} \quad (5)$$

$$SmellIndex = sort(S) \quad (6)$$

where $S(i)$ ranks in the top fifty percent of the population, r represents a random value in $[0,1]$, bF denotes the best fitness in the current iteration phase, wF means the worst fitness value, $SmellIndex$ specifies the series of sorted fitness values.

Wrap Food: The following equation may be used to update the position of slime mold:

$$\overrightarrow{X^*} = \begin{cases} rand \cdot (UB - LB) + LB, rand < z \\ \overrightarrow{X_b(t)} + \overrightarrow{vb} \cdot \left(W \cdot \overrightarrow{X_A(t)} - \overrightarrow{X_B(t)}\right), r < p \\ \overrightarrow{vc} \cdot \overrightarrow{X(t)}, r \geq p \end{cases} \quad (7)$$

where $rand$ and r stand for the random value in [0,1], and LB and UB stand for the lower and upper search range limits.

Grabble Food: As the number of iterations rises, the value of $\overrightarrow{vb}$ varies at random between $[-a, a]$ and eventually approaches zero. The value of $\overrightarrow{vc}$ varies between [-1,1] and finally goes to zero.

Leader Slime Mould Algorithm

SMA's primary reliance on the population's two slime molds and best leader leads to poor exploitation when more convergence iterations are performed. To eliminate this situation, LSMA has been proposed [1].

According to [2], the updating rule of the SMA concentration for the i-th slime mould $X_i(= \{x_i^1, x_i^2, \dots, x_i^k\})$ for a k dimensional issue from N slime mould is as follows:

$$\overrightarrow{X^*} = \begin{cases} rand \cdot (UB - LB) + LB, & r_1 < z \\ X_{GlobalBest}(t) + V_a.(W.X_{R1} - X_{R2}), & r_1 \geq z \ and \ r_2 < p \\ V_b \cdot X_i(t), & r_1 \geq z \ and \ r_2 \geq p \end{cases} \quad (8)$$

and

$$X_i(1) = r_1.(UB - LB) + LB \quad (9)$$

The r_1 and r_2 are random values in the range of 0 and 1; t is the current iteration, UB and LB upper and lower boundary of the search space, respectively, $X_{GlobalBest}$ is the global best concentration current iteration t, V_a

represents the velocity that is spread evenly throughout the interval, V_b represents the velocity that goes from 1 to 0 in a linear fashion, W represents the weight of the slime mould, X_{R1} and X_{R2} are the two types of slime mould that were chosen at random from the population of N, p is the probability to determine the slime mould trajectory, z is the elimination-and-dispersal rate which is fixed at 0:03 and $i \in 1, 2, \dots, N$.

The performance of the i-th slime mould is determined by its current fitness $f(X_i)$ and by the fitness of the world's best concentration $f(X_{L1})$, which is formulated as:

$$p = \tanh\left|f(X_i) - f(X_{L1})\right| \quad (10)$$

Both the velocity V_a and the velocity V_b are equally distributed in the $[-a, a]$ and $[-b, b]$ ranges, respectively. The values of a and b are as follows:

$$a = \operatorname{arctanh}\left(-\left(\frac{t}{tmax}\right) + 1\right) \quad (11)$$

and

$$b = 1 - \frac{t}{tmax} \quad (12)$$

The W is calculated using the slime mould's local fitness value. Let's rank the N slime mould's fitness value for the minimization issue in ascending order in iteration t.

$$\left[sorted_{fitness}, sort_{Index}\right] = sort(f) \quad (13)$$

where $f = (fX_1), f(X_2), \dots, f(X_N))$

The W is then calculated as follows:

$$W(\,sort_{Index}(l) = \begin{cases} 1 + r_3 \cdot log\left(\frac{f_{LocalBest} - sortf(l)}{f_{LocalBest} - f_{LocalWorst}} + 1\right), 1 \le l \le \frac{N}{2} \\ 1 - r_3 \cdot log\left(\frac{f_{LocalBest} - sortf(l)}{f_{LocalBest} - f_{LocalWorst}} + 1\right), \frac{N}{2} \le l \le N \end{cases} \quad (14)$$

$$f_{LocalBest = sortedFitness}(1) \quad (15)$$

$$f_{LocalWorst = sortedFitness}(N) \quad (16)$$

The best concentration globally is designated as leader1 (L1), while Leader2 (L2) and Leader3 (L3) stand for the second and third greatest concentrations, respectively. The model for the new updating rule of ith slime mould at iteration $(t + 1)$ in LSMA is:

$$X_i(t+1) = r_1.(UB - LB) + LB, when\ r_1 < z \quad (17.a)$$

$$X_i(t+1) = X_{L1}(t) + V_a.(W.X_{L2} - X_{R1}) + (W.X_{L3} - X_{R2}), \quad r_1 \ge z\ and\ r_2 < p \quad (17.b)$$

$$X_i(t+1) = V_b \cdot X_i(t),\ when\ r_1 \ge z\ and\ r_2 \ge p \quad (17.c)$$

Equilibrium Optimizer Slime Mould Algorithm

The search pattern of the SMA requires differential information between two random slime molds and the best slime mold, which may cause results to deviate from the optimum value. The equilibrium pools of the top potential solutions determine how EO searches.

In order to increase integrate the equilibrium pool and augment the SMA's properties, Naik et al. suggested the ESMA.

The air smell is how the slime mold finds the food. Assume there are N slime molds, each of whose location is given by the vector $X = [\vec{X}_1, \vec{X}_2, \dots, \vec{X}_N]'$. The ith slime mold's starting location vector is generated at random as Eq. (18):

$$\vec{X}_i(t=1) = r_1.(UB - LB) + LB, i = 1,2, \dots, N. \quad (18)$$

where t denotes the current iteration number, UB upper bound and LB lower bound. The new iteration in $t + 1$ is modeled as in Eq. (19).

$$\overrightarrow{X_i}(t+1) = \begin{cases} r_1.(UB - LB) + LB & r_1 < z \\ \vec{X}_{Gbest} + \overrightarrow{step_a}.\left(\vec{W}.\vec{X}_A - \vec{X}_B\right) & r_2 < p_i(t)\ and\ r_1 \ge z \\ \overrightarrow{step_b}.\vec{X}_i(t) & r_2 \ge p_i(t)\ and\ r_1 \ge z \end{cases} \quad (19)$$

Here $\vec{X}_{Gbest}$ is the global best value in the number of iterations. $\vec{X}_A$ and $\vec{X}_B$ are two randomly selected individuals in t iterations. The r_1 and r_2 values are random variables that take values between 0 and 1. The z value is 0.03, which is a constant. This number represents the likelihood that is used in the process of eradicating and dispersing the slime mold.

The weighting factor for the slime mold at iteration t is known as the $\vec{W}$ value, and it is determined using the local fitness value. The order of the fitness values in ascending order is done with $[sortf, sortIndex] = sort\ (f), where\ f = \{f_1, f_2, \dots, f_N\}$. Thus, the value of w is calculated as in Eq. (20).

$$\vec{W}\left(sortIndex(j)\right) = \begin{cases} 1 + r_3.\log\left(\frac{f_{Lbest} - sortf\ (j)}{f_{Lbest} - f_{Lworst}} + 1\right) & 1 \le j \le \frac{N}{2} \\ 1 - r_3.\log\left(\frac{f_{Lbest} - sortf\ (j)}{f_{Lbest} - f_{Lworst}} + 1\right) & \frac{N}{2} < j \le N \end{cases} \quad (20)$$

The r_3 value is random variables that take values between 0 and 1. f_{Lworst} and f_{Lbest} are the local worst ($f_{Lworst} = sortf(N)$) and best fitness ($f_{Lbest} = sortf(1)$ values, respectively, in the current iteration. The p_i value is calculated as in Eq. (21). p_i value with the help of other slime molds i. shows the decision probability of the trajectory of the slime mold.

$$p_i = tanh|f(X_i) - f_{Gbest}| \quad (21)$$

Here, the i value ranges from 1 to N, and X_i shows the position of the slime molds in the i'th iteration. f_{Gbest}, on the other hand, holds the best global best fitness value up to the current iteration.

The $\overrightarrow{step_a}$ and $\overrightarrow{step_b}$ represent a step size relative to the uniform distribution in the $[-a, a]$ and $[-b, b]$ ranges, respectively. a and b are calculated according to Eq. (22 and 23). And the T value indicates the maximum iteration.

$$a = \text{arctanh}\left(-\left(\frac{t}{T}\right) + 1\right) \quad (22)$$

$$b = 1 - \frac{t}{T} \quad (23)$$

$\vec{X}_A$ and $\vec{X}_B$ consist of two randomly selected individuals in the N slime mold. This can create the problem of falling to the local minimum. Here, the ESMA method, which replaces $\vec{X}_A$ with a position vector from the balance pool consisting of the best four position vectors and takes into account the average position, has been developed. The individual elements of the equilibrium pool are defined as in Eq. (24).

$$\vec{X}_{eq(1)} = X\big(sortIndex(1)\big)$$

$$\vec{X}_{eq(2)} = X(sortIndex(2))$$

$$\vec{X}_{eq(3)} = X(sortIndex(3)) \quad (24)$$

$$\vec{X}_{eq(4)} = X(sortIndex(4))$$

$$\vec{X}_{ave} = \frac{\vec{X}_{eq(1)} + \vec{X}_{eq(2)} + \vec{X}_{eq(3)} + \vec{X}_{eq(4)}}{4}$$

Equilibrium pool $\vec{X}_{eq,pool} = \{\vec{X}_{eq(1)}, \vec{X}_{eq(2)}, \vec{X}_{eq(3)}, \vec{X}_{eq(4)}, \vec{X}_{eq(ave)}\}$ is created using 5 different vectors in Eq. (24). In ESMA, the position vector of the next $X_i (j = 1,2,3, \dots, N$ is modelled as in Eq. (25).

$$\vec{X}_i(t+1) = r_1.(UB - LB) + LB, when\ r_1 < z \quad (25)$$

$\vec{X}_i(t+1) = \vec{X}_{Gbest} + \overrightarrow{step_a}\big(\vec{W}.\vec{X}_{eq} + \vec{X}_B\big), when\ r_2 < p_i(\text{t})$ and $r_1 \geq \text{z}$

$\vec{X}_i(t+1) = \overrightarrow{step_b}.\vec{X}_i(t), when\ \ r_2 \geq p_i(\text{t})$ and $r_1 \geq \text{z}$

Results and Discussion

In the study, IEEE Congress on Evolutionary Computation (CEC) 2020 test functions were selected to analyze the performance of SMA, LSMA, and ESMA methods [14]. The CEC2020 test functions consist of 10 different test functions. The first is the unimodal Shifted and Rotated Bent Cigar function. The second, third, and fourth functions are the multimodal functions Shifted and Rotated Schwefel's, Shifted and Rotated Lunacek bi-Rastrigin, and Expanded Rosenbrock's plus Griewangk's function, respectively [15]. In addition, there are 3 different hybrids and 3 different composition functions with N values of 3, 4, and 5, respectively. The names and equations of these functions are listed in Table 1. Unimodal functions play a decisive role in the convergence performance of algorithms. Multimodal functions are used to see if there are problems with early convergence and local optimization in an algorithm.

On the other hand, hybrid and composition functions, are used to determine the performance of algorithms' ability to avoid local optima and their balance between discovery and exploitation, as they have many local optima. Experiments in the study were carried out on a computer with the Windows 10 operating system, 32 GB RAM, and a CPU of Intel (R) core i9-10900k (3.7 GHz). In the study, the special parameters of the SMA, LSMA, and ESMA algorithms were taken exactly the same as in the original articles. In order to make a fair evaluation under equal conditions, the number of iterations was 1000 and all experiments were run 20 times. In addition, the performances of the algorithms in 3 different dimension values were compared by taking the dimension as 5, 10 and 20.

Hybrid Functions

$$F(x) = g_1(M_1 z_1) + g_2(M_2 z_2) + \cdots\ g_N(M_N z_N) + F^*(x)$$

$F(x)$:Hybrid function

$g_i(x)$: i^{th} basic function used to construct the hybrid function

N: Number of basic functions

$z = [z_1, z_2, \dots, z_N]$

$z_1 = [$

$y = x - o_i, S = randperm(1:D)$

p_i =Used to control the percentage of $g_i(x)$

$$n_i = Dimension\ for\ each\ basic\ function \sum_{i=1}^{N} n_i = D$$

$n_1 = \lceil p_1 D \rceil, n_2 = \lceil p_2 D \rceil, \dots, n_{N-1} = \lceil p_{N-1} D \rceil,$

$$n_N = D - \sum_{i=1}^{N-1} n_i$$

Composition Functions

$$F(x) = \sum_{i=1}^{N} \{w_i^* [\lambda_i g_i(x) + bias_i]\} + F^*$$

$F(x)$: Composition function

$g_i(x)$: i^{th} basic function used to construct the composition function

N: Number of basic functions

o_i: New shifted optimum position for each $g_i(x)$, define the global and local optima' s position

$bias_i$: defines which optimum is global optimum σ_i: used to control each $g_i(x)'$ s coverage range, a small σ_i gives a narrow range for that $g_i(x)$

λ_i: used to control each $g_i(x)$' s height

w_i: weight value for $g_i(x)$, calculated as below:

$$w_i = \frac{1}{\sqrt{\sum_{j=1}^{D}(x_j - o_j)^2}} exp\left(-\frac{\sum_{j=1}^{D}(x_j - o_j)^2}{2D\sigma_i^2}\right)$$

Then normalize the weight $\omega_i = w_i / \sum_{i=1}^{n} w_i$

So when $x = o_i$, $\omega_j = \begin{cases} 1, & j = i \\ 0, & j \neq i \end{cases}$ for $j = 1, 2, \dots, N, f(x) = bias_i + f^*$

Table 1. CEC'2020 test functions and equations

No	Function Name	Equation	Fi*
F1	Shifted and Rotated Bent Cigar Function	$F1 = x_1^2 + 10^6 \sum_{i=2}^{D} x_i^2$ $F1\big(M(x - o_1)\big) + F1^*$	100
F2	Shifted and Rotated Schwefel's Function	$f(x) = 418.9829 \times D - \sum_{i=1}^{D} g(z_i)$ $z_i = x_i + 4.209687462275036e + 002$ $g(z_i) = \begin{cases} z_i sin\left(\lvert z_i\rvert^{\frac{1}{2}}\right), & if\ \lvert z_i\rvert \le 500 \\ 500 - mod(z_i, 500))\sin\left(\sqrt{\lvert 500 - mod(z_i, 500)\rvert} - \frac{(z_i - 500)^2}{10000d}\right) & if\ z_i > 500 \\ (mod(\lvert z_i\rvert, 500) - 500)\sin\left(\sqrt{\lvert mod\lvert z_i\rvert, 500) - 500\rvert} - \frac{(z_i + 500)^2}{10000d}\right) & if\ z_i < -500 \end{cases}$ $F2(x) = f\left(M\left(\frac{1000(x - o_2)}{100}\right)\right) + F2^*$	1100
F3	Shifted and Rotated Lunacek bi-Rastrigin Function	$f(x) = min\left(\sum_{i=1}^{D}(\widehat{x_i} - \mu_0)^2, d\,D + s\sum_{i=1}^{D}(\widehat{x_i} - \mu_0)^2\right) + 10\left(D - \sum_{i=1}^{D}\cos(2\pi\widehat{z_i})\right)$ $\mu_0 = 2.5, \mu_1 = \sqrt{\frac{\mu_0^2 - d}{s}}, s = 1 - \frac{1}{2\sqrt{D + 20} - 8.2}, d = 1$ $y = \frac{10(x - o)}{100}, \frac{x_i}{x_i} = 2sign(x_i^*)y_i + \mu_0, for\ i = 1, 2, \dots, D$ $z = \bigwedge^{100}(\hat{x} - \mu_0)$ $F3(x) = f\left(M\left(\frac{600(x - o_3)}{100}\right)\right) + F3^*$	700
F4	Expanded Rosenbrock's plus Griewangk's Function	$f1(x) = \sum_{i=1}^{D-1}(100(x_i^2 - x_{i+1})^2 + (x_i - 1)^2)$ $f2(x) = \sum_{i=1}^{D}\frac{x_i^2}{4000} - \prod_{i=1}^{D} cos\left(\frac{x_i}{\sqrt{i}}\right) + 1$ $f4 = f2\big(f1(x_1, x_2)\big) + f2\big(f1(x_2, x_3)\big) + \cdots + f2\big(f1(x_{D-1}, x_D)\big) + f2\big(f1(x_D, x_1)\big) + f4^*$	1900
F5	Hybrid Function 1 (N = 3)	$N = 3, p = [0.3, 0.3, 0.4]$ $g1: Modified\ Schwefel's\ Function, g2: Rastrigin's\ Function$ $g3: High\ Conditioned\ Elliptic\ Function\ f3$	1700
F6	Hybrid Function 2 (N = 4)	$N = 4, p = [0.2, 0.2, 0.3, 0.3]$ $g1: Expanded\ Schaffer\ Function, g2: HGBat\ Function$ $g3: Rosenbrock's\ Function, g4: Modified\ Schwefel'sFunction$	1600
F7	Hybrid Function 3 (N = 5)	$N = 5, p = [0.1, 0.2, 0.2, 0.2, 0.3]$ $g1: Expanded\ Schaffer\ Function, g2: HGBat\ Function$ $g3: Rosenbrock's\ Function, g4: Modified\ Schwefel's\ Function$ $g5: High\ Conditioned\ Elliptic\ Function$	2100
F8	Composition Function 1 (N = 3)	$N = 3, \sigma = [10, 20, 30], \lambda = [1, 10, 1], bias = [0, 100, 200]$ $g1: Rastrigin's\ Function\ , g2: Griewank's\ Function, g3: Modified\ Schwefel's\ Function$	2200
F9	Composition Function 2 (N = 4)	$N = 4, \sigma = [10, 20, 30, 40], \lambda = [10, 1e-6, 10, 1], bias = [0, 100, 200, 300]$ $g1: Ackley's\ Function, g2: High\ Conditioned\ Elliptic\ Function$ $g3: Griewank's\ Function, g4: Rastrigin's\ Function$	2400
F10	Composition Function 3 (N = 5)	$N = 5, \sigma = [10, 20, 30, 40, 50], \lambda = [10, 1, 10, 1e-6, 1], bias = [0, 100, 200, 300, 400]$ $g1: Rastrigin's\ Function, g2: Happycat\ Function, g3: Ackley's\ Function$ $g4: Discus\ Function, g5: Rosenbrock's\ Function$	2500

Table 2, Table 3 and Table 4 show the results according to dimensions 5, 10, and 20, respectively. Average (Avg.), standard deviation (Std.) and minimum (Min.) values are given in the tables. In addition, for ease of reading, the best values found in each test function are made in bold.

Table 2. dim 5

Functions	Alg.	Metrics		
		Avg.	*Std.*	*Min.*
F1	SMA	**4.51E+03**	**5.33E+03**	1.23E+02
	LSMA	6.01E+03	5.59E+03	1.54E+02
	ESMA	5.20E+03	5.80E+03	**1.04E+02**
F2	SMA	1.22E+03	1.02E+02	1.13E+03
	LSMA	**1.21E+03**	**8.38E+01**	**1.11E+03**
	ESMA	1.24E+03	1.01E+02	1.13E+03
F3	SMA	7.08E+02	**1.79E+00**	7.05E+02
	LSMA	7.09E+02	2.16E+00	7.03E+02
	ESMA	**7.07E+02**	2.42E+00	**7.02E+02**
F4	SMA	**1.90E+03**	**1.31E-01**	**1.90E+03**
	LSMA	**1.90E+03**	1.38E-01	**1.90E+03**
	ESMA	**1.90E+03**	**1.31E-01**	**1.90E+03**
F5	SMA	1.72E+03	1.25E+01	**1.70E+03**
	LSMA	1.83E+03	6.83E+01	1.71E+03
	ESMA	**1.71E+03**	**9.52E+00**	**1.70E+03**
F6	SMA	**1.60E+03**	2.53E-01	**1.60E+03**
	LSMA	**1.60E+03**	3.14E-01	**1.60E+03**
	ESMA	**1.60E+03**	**2.27E-01**	**1.60E+03**
F7	SMA	**6.55E+04**	**0.00E+00**	**6.55E+04**
	LSMA	**6.55E+04**	**0.00E+00**	**6.55E+04**
	ESMA	**6.55E+04**	**0.00E+00**	**6.55E+04**
F8	SMA	2.21E+03	6.28E+00	**2.20E+03**
	LSMA	**2.20E+03**	**5.66E+00**	**2.20E+03**
	ESMA	2.23E+03	3.87E+01	**2.20E+03**
F9	SMA	2.58E+03	4.51E+01	**2.50E+03**
	LSMA	**2.56E+03**	5.14E+01	**2.50E+03**
	ESMA	2.58E+03	**4.17E+01**	**2.50E+03**
F10	SMA	**2.85E+03**	1.32E-02	2.85E+03
	LSMA	**2.85E+03**	1.06E+01	**2.80E+03**
	ESMA	**2.85E+03**	**1.13E-02**	2.85E+03

Table 3. dim 10

Functions	Alg.	Metrics		
		Avg.	*Std.*	*Min.*
F1	SMA	7.18E+03	4.57E+03	9.55E+02
	LSMA	7.03E+03	4.93E+03	**1.02E+02**
	ESMA	**6.37E+03**	4.72E+03	3.01E+02
F2	SMA	1.66E+03	2.38E+02	**1.23E+03**
	LSMA	1.69E+03	2.12E+02	1.24E+03
	ESMA	**1.64E+03**	**1.59E+02**	1.33E+03
F3	SMA	7.29E+02	**8.05E+00**	**7.15E+02**
	LSMA	7.30E+02	1.04E+01	7.16E+02
	ESMA	**7.28E+02**	8.23E+00	7.17E+02
F4	SMA	**1.90E+03**	**4.50E-01**	**1.90E+03**
	LSMA	**1.90E+03**	7.25E-01	**1.90E+03**
	ESMA	**1.90E+03**	5.14E-01	**1.90E+03**
F5	SMA	**7.98E+03**	**6.41E+03**	**1.86E+03**
	LSMA	1.01E+04	6.63E+03	2.50E+03
	ESMA	1.73E+04	1.82E+04	1.90E+03
F6	SMA	**1.60E+03**	2.70E-01	**1.60E+03**
	LSMA	**1.60E+03**	2.99E-01	**1.60E+03**
	ESMA	**1.60E+03**	**2.43E-01**	**1.60E+03**
F7	SMA	1.01E+04	8.46E+03	**2.28E+03**
	LSMA	5.93E+03	**4.41E+03**	2.65E+03
	ESMA	**5.35E+03**	4.81E+03	2.17E+03
F8	SMA	2.39E+03	3.13E+02	**2.20E+03**
	LSMA	**2.38E+03**	**2.46E+02**	2.24E+03
	ESMA	**2.38E+03**	2.48E+02	2.30E+03
F9	SMA	2.76E+03	9.43E+00	2.74E+03
	LSMA	**2.75E+03**	**8.60E+00**	2.74E+03
	ESMA	**2.75E+03**	5.82E+01	**2.50E+03**
F10	SMA	2.94E+03	3.16E+01	**2.90E+03**
	LSMA	**2.93E+03**	**2.62E+01**	**2.90E+03**
	ESMA	**2.93E+03**	2.67E+01	**2.90E+03**

Table 4. dim 20

Functions	Alg.	Metrics		
		Avg.	*Std.*	*Min.*
F1	SMA	6.31E+03	4.10E+03	**1.46E+02**
	LSMA	**5.07E+03**	4.12E+03	1.49E+02
	ESMA	7.85E+03	**3.81E+03**	2.91E+02
F2	SMA	1.94E+03	**2.74E+02**	1.54E+03
	LSMA	2.95E+03	6.14E+02	2.02E+03
	ESMA	**1.93E+03**	3.06E+02	**1.50E+03**
F3	SMA	7.52E+02	1.00E+01	7.38E+02
	LSMA	7.88E+02	1.96E+01	7.53E+02
	ESMA	**7.46E+02**	**9.71E+00**	**7.31E+02**
F4	SMA	**1.90E+03**	1.06E+00	**1.90E+03**
	LSMA	**1.90E+03**	1.42E+00	**1.90E+03**
	ESMA	**1.90E+03**	**1.01E+00**	**1.90E+03**
F5	SMA	4.22E+05	3.18E+05	1.81E+04
	LSMA	4.34E+05	2.64E+05	8.05E+04
	ESMA	**3.38E+05**	**2.32E+05**	**4.29E+04**
F6	SMA	**2.05E+03**	**0.00E+00**	**2.05E+03**
	LSMA	**2.05E+03**	**0.00E+00**	**2.05E+03**
	ESMA	**2.05E+03**	**0.00E+00**	**2.05E+03**
F7	SMA	4.20E+05	3.66E+05	5.40E+03
	LSMA	2.00E+05	2.35E+05	1.08E+04
	ESMA	**1.91E+05**	**2.06E+05**	**5.41E+03**
F8	SMA	3.24E+03	**1.12E+03**	**2.30E+03**
	LSMA	3.30E+03	1.28E+03	**2.30E+03**
	ESMA	**2.94E+03**	1.16E+03	**2.30E+03**
F9	SMA	2.86E+03	**1.62E+01**	2.84E+03
	LSMA	2.86E+03	1.93E+01	2.84E+03
	ESMA	**2.85E+03**	**1.62E+01**	**2.82E+03**
F10	SMA	2.93E+03	2.81E+01	**2.91E+03**
	LSMA	2.93E+03	3.06E+01	**2.91E+03**
	ESMA	**2.91E+03**	**1.10E+00**	**2.91E+03**

When Table 2 is examined, it is seen that the methods give the same average in 4 of the 10 test functions. While LSMA gave the best average in 3 functions, ESMA gave the best average in 2 functions. SMA, on the other hand, gave the best average in only one function. The convergence curve and boxplot graphics according to Dimension 5 are given in Figure 1 and Figure 2, respectively.

When Table 3 is examined, it is seen that the methods give the same average in 2 of the 10 test functions. It was seen that LSMA and ESMA gave the best average in 3 functions. While ESMA gave the best average in 4 functions, SMA gave the best average in 1 of them. The convergence curve and boxplot graphics according to Dimension 10 are given in Figure 3 and Figure 4, respectively.

When Table 4 is examined, it is seen that the methods give the same average in 2 of the 10 test functions. While ESMA gave the best average in 7 functions, LSMA gave the best result in only 1 of them. The convergence curve and boxplot graphics according to Dimension 20 are given in Figure 5 and Figure 6, respectively.

In Table 5, the algorithm or algorithms that give the best value for each test function in different dimensions according to the average value are given. When Table 5 is examined, it is seen that the performance of the methods varies according to the dimension in unimodal functions. In multimodal functions, it was seen that ESMA achieved a better mean value. It has been observed that ESMA gives relatively better results than other methods in hybrid functions. Considering the composite functions, LSMA gave the best average value when the dimension was taken as 5. When the dimension is taken as 10, it is seen that the performances of LSMA and ESMA are the same. Finally, it is seen that ESMA gives better performance when the dimension is taken as 20. In the light of these experimental results, it has been seen that the ESMA method outperforms the other methods, SMA and LSMA, in CEC2020 functions.

Table 5. Best algorithm or algorithms according to the average value

Functions	Dimension (5)	Dimension (10)	Dimension (20)
f1	SMA	ESMA	LSMA
f2	LSMA	ESMA	ESMA
f3	ESMA	ESMA	ESMA
f4	ALL	ALL	ALL
f5	ESMA	SMA	ESMA
f6	ALL	ALL	ALL
f7	ALL	ESMA	ESMA
f8	LSMA	LSMA and ESMA	ESMA
f9	LSMA	LSMA and ESMA	ESMA
f10	ALL	LSMA and ESMA	ESMA

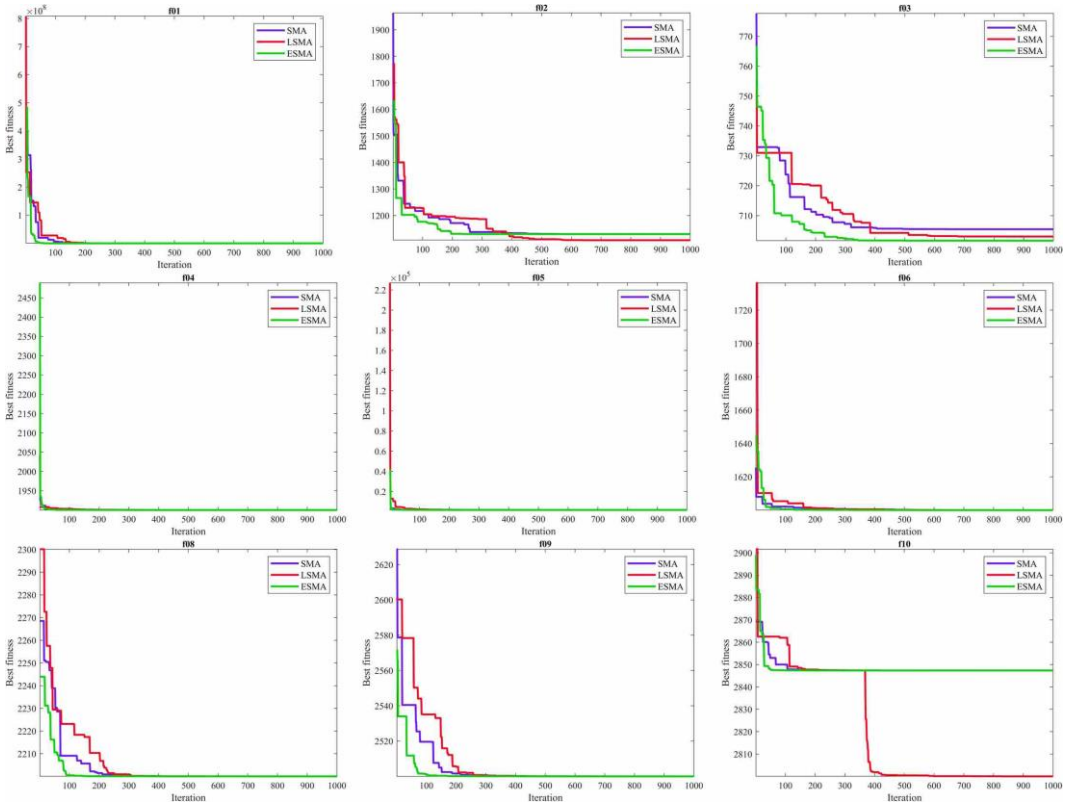

Figure 1. Convergence curve of the compared methods when dimension 5

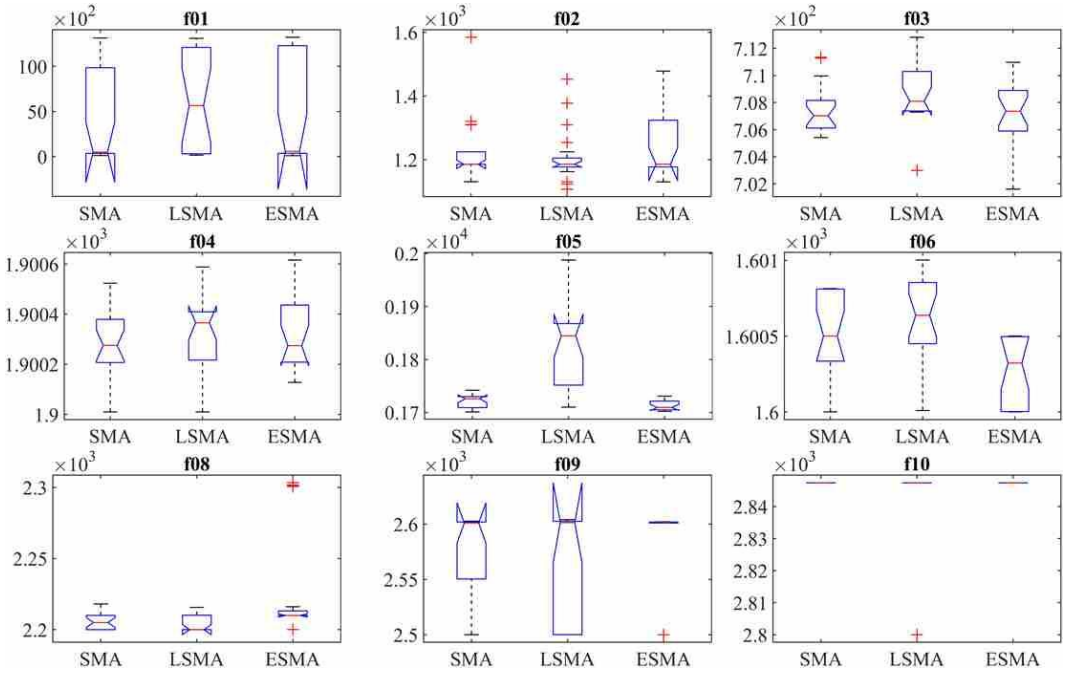

Figure 2. Boxplot of the compared methods when dimension 5

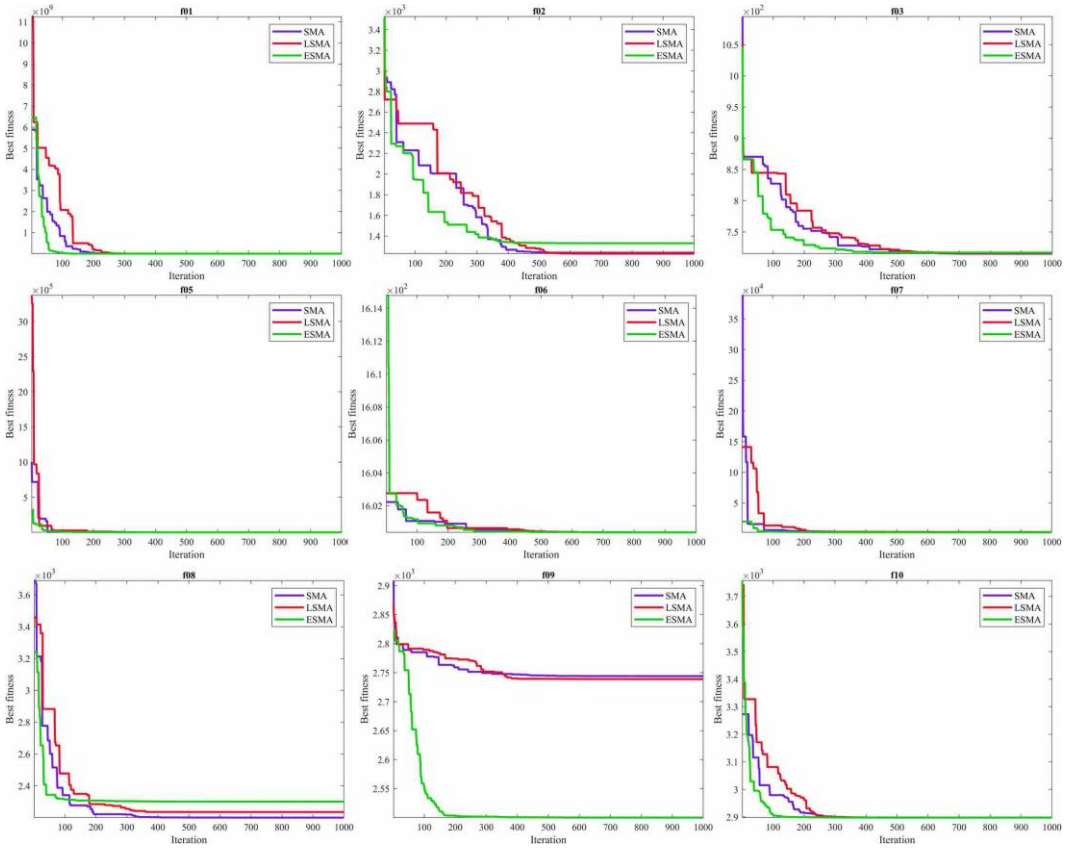

Figure 3. Convergence curve of the compared methods when dimension 10

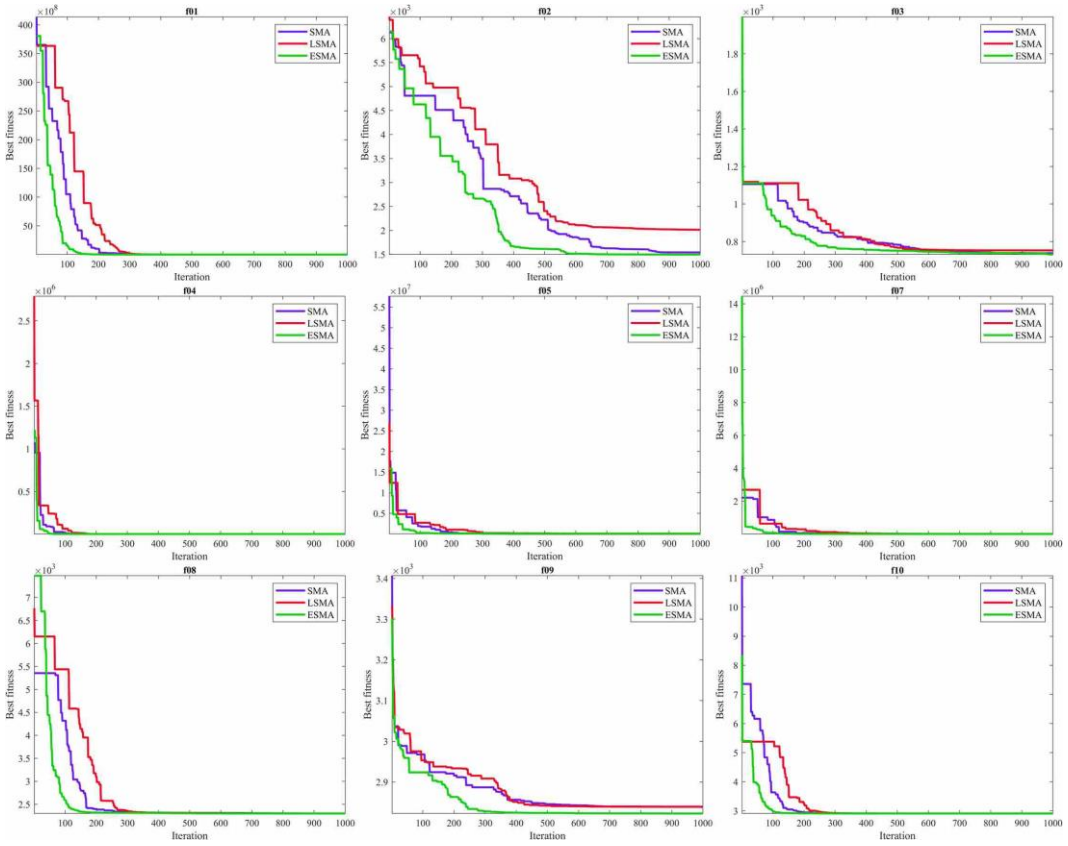

Figure 4. Boxplot of the compared methods when dimension 10

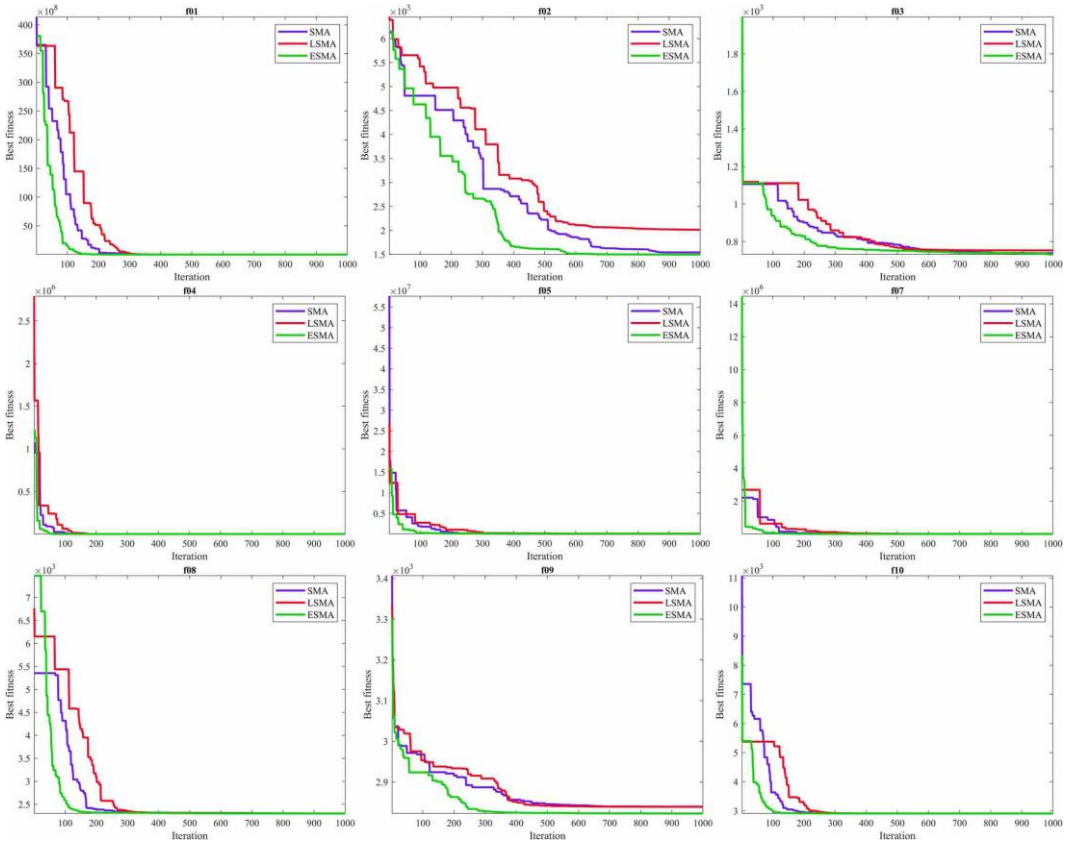

Figure 5. Convergence curve of the compared methods when dimension 10

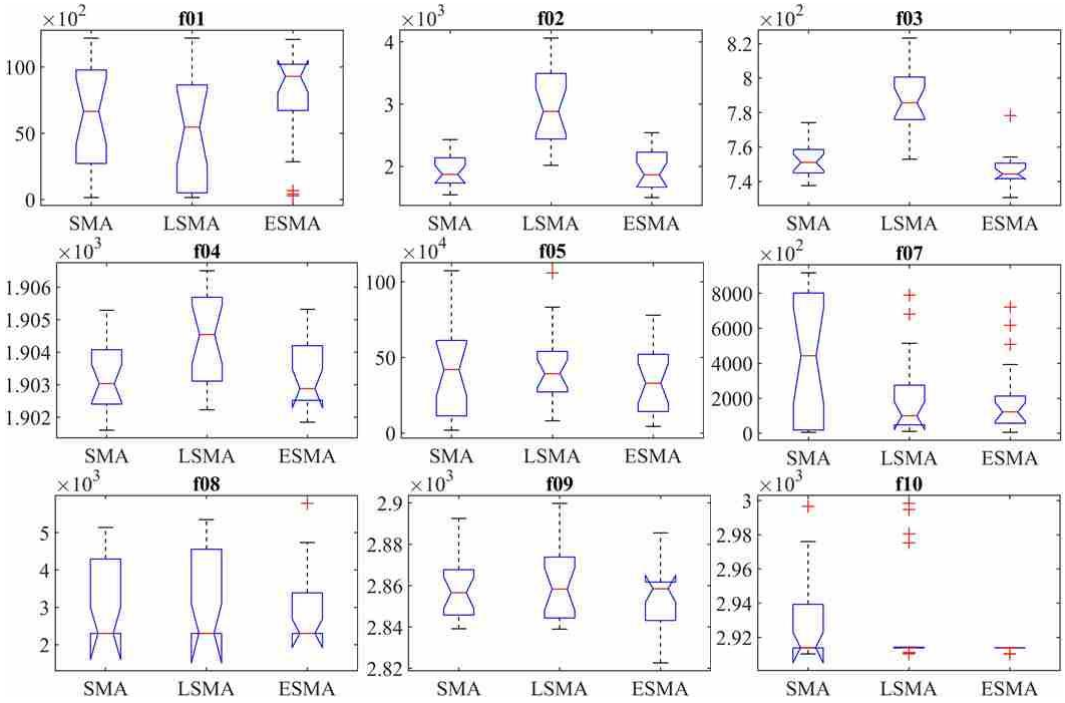

Figure 6. Boxplot of the compared methods when dimension 2

Conclusions

Metaheuristic methods have been used successfully in the literature for solving different problems. As the literature studies show, there is no method that gives the best performance for each problem. This increases the interest of the researchers in this subject. For this reason, it is aimed to find the best method by suggesting different hybrid versions of the newly introduced methods to the literature. In this study, performance analyses were made by running different hybrid versions of the SMA method, which has been proposed in recent years, in the CEC 2020 test functions under equal conditions. The experimental results showed that ESMA performed better than the standard SMA and LSMA. This study is significant both for making it easier for researchers to access one of the most recent metaheuristic optimization algorithms, SMA, and its variants, as well as for assisting them in selecting the best algorithm by providing a preliminary idea about the performance of metaheuristic algorithms that they can use in their studies.

References

[1] Sayed, G.I., Khoriba, G., Haggag, M.H.: A novel chaotic salp swarm algorithm for global optimization and feature selection. Appl. Intell. 48, 3462–3481 (2018). https://doi.org/10.1007/s10489-018-1158-6.

[2] Mirjalili, S., Mirjalili, S.M., Lewis, A.: Grey Wolf Optimizer. Adv. Eng. Softw. 69, 46–61 (2014). https://doi.org/10.1016/j.advengsoft.2013.12.007.

[3] Faramarzi, A., Heidarinejad, M., Stephens, B.,Mirjalili, S.: Equilibrium optimizer: A novel optimization algorithm. Knowledge-Based Syst. 191, 105190 (2020). https://doi.org/10.1016/j.knosys.2019.105190.

[4] Hashim, F.A., Hussain, K., Houssein, E.H., Mabrouk, M.S., Al-Atabany, W.: Archimedes optimization algorithm: a new metaheuristic algorithm for solving optimization problems. Appl. Intell. 51, 1531–1551 (2021). https://doi.org/10.1007/s10489-020-01893-z.

[5] Dhiman, G., Kaur, A.: Spotted Hyena Optimizer for Solving Engineering Design Problems. Proc. - 2017 Int. Conf. Mach. Learn. Data Sci. MLDS 2017. 2018-Janua, 114–119 (2018). https://doi.org/10.1109/MLDS.2017.5.

[6] Abualigah, L., Yousri, D., Abd Elaziz, M., Ewees, A.A., Al-qaness, M.A.A., Gandomi, A.H.: Aquila Optimizer: A novel meta-heuristic optimization algorithm. Comput. Ind. Eng. 157, 107250 (2021). https://doi.org/10.1016/j.cie.2021.107250.

[7] Li, S., Chen, H., Wang, M., Heidari, A.A., Mirjalili, S.: Slime mould algorithm: A new method for stochastic optimization. Futur. Gener. Comput. Syst. 111, 300–323 (2020). https://doi.org/10.1016/j.future.2020.03.055.

[8] Altay, E.V.: Gerçek dünya mühendislik tasarım problemlerinin çözümünde kullanılan metasezgisel optimizasyon algoritmalarının performanslarının incelenmesi. International Journal of Innovative Engineering Applications. 6, 65-74 (2022).

[9] Fagan, F., Vuuren, J.H. Van: A unification of the prevalent views on exploitation, exploration, intensification and diversification. Int. J. Metaheuristics. 2, 294 (2013).

https://doi.org/10.1504/ijmheur.2013.056407.

[10] Thangaraj, R., Pant, M., Abraham, A., Bouvry, P.: Particle swarm optimization: Hybridization perspectives and experimental illustrations. Appl. Math. Comput. 217, 5208–5226 (2011). https://doi.org/10.1016/j.amc.2010.12.053.

[11] Naik, M.K., Panda, R., Abraham, A.: Normalized square difference based multilevel thresholding technique for multispectral images using leader slime mould algorithm. J. King Saud Univ. - Comput. Inf. Sci. (2020). https://doi.org/10.1016/j.jksuci.2020.10.030.

[12]. Naik, M.K., Panda, R., Abraham, A.: An entropy minimization based multilevel colour thresholding technique for analysis of breast thermograms using equilibrium slime mould algorithm. Appl. Soft Comput. 113, 107955 (2021). https://doi.org/10.1016/j.asoc.2021.107955.

[13] Altay, O.: Chaotic slime mould optimization algorithm for global optimization. Springer Netherlands (2022). https://doi.org/10.1007/s10462-021-10100-5.

[14] Mohamed, A.W., Hadi, A.A., Mohamed, A.K., Awad, N.H.: Evaluating the Performance of Adaptive GainingSharing Knowledge Based Algorithm on CEC 2020 Benchmark Problems. 2020 IEEE Congr. Evol. Comput. CEC 2020 - Conf. Proc. (2020). https://doi.org/10.1109/CEC48606.2020.9185901.

[15] Varol Altay, E., Altay, O.: Güncel metasezgisel optimizasyon algoritmalarının CEC2020 test fonksiyonları ile karşılaştırılması. DÜMF Mühendislik Derg. 5, 729–741