International Journal of Industrial Engineering Computations 14 (2023) 309–322

Contents lists available at GrowingScience

International Journal of Industrial Engineering Computations

homepage: www.GrowingScience.com/ijiec

Hybrid algorithm proposal for optimizing benchmarking problems: Salp swarm algorithm enhanced by arithmetic optimization algorithm

Erkan Erdemir[a*]

[a]*Department of Information Technologies, Tokat Vocational and Technical Anatolian High School, Merkez/Tokat, 60030, Türkiye*

C H R O N I C L E

Article history:
Received October 2 2022
Accepted December 2 2022
Available online
January, 3 2023

Keywords:
Arithmetic
Benchmark
Optimization
Metaheuristic
Salp
Swarm

A B S T R A C T

Metaheuristic algorithms are easy, flexible and nature-inspired algorithms used to optimize functions. To make metaheuristic algorithms better, multiple algorithms are combined and hybridized. In this context, a hybrid algorithm (HSSAOA) was developed by adapting the exploration phase of the arithmetic optimization algorithm (AOA) to the position update part of the salp swarm algorithm (SSA) of the leader salps/salps. And also, there have also been a few new additions to the SSA. The proposed HSSAOA was tested in three different groups using 22 benchmark functions and compared with 7 well-known algorithms. HSSAOA optimized the best results in a total of 16 benchmark functions in each group. In addition, a statistically significant difference was obtained compared to other algorithms.

1. Introduction

Optimization is the process of bringing the problem or problems in the sciences or social sciences closest to the desired result..When optimizing problems, constraints and certain limits can be introduced. The result obtained in the optimization process is not always the lowest result. According to the type of problem, the best case may be the highest result (Tarik et al., 2016; Wright, 2022; Yang & Deb, 2015). In order to reach the best approach to the solution in optimization problems, algorithms are used in which operations are performed step by step. These algorithms are of two types, deterministic and stochastic algorithms. (Yang, 2010a; Britannica, 2022). When deterministic algorithms are used in the solution of problems, exact and same values are obtained as a solution. The process is short. However, there is a high probability of getting stuck in the local optimum. When the same problem is solved using a stochastic algorithm, different solutions are obtained each time. Even if it takes longer to reach the best solution in stochastic algorithms than deterministic algorithms, using randomness in the search process ensures that the local optimum is not caught and a better global solution is obtained. (Brownlee, 2021; Friedrich, 2022; Kochenderfer & Wheeler, 2019; Sergeyev et al., 2017; Yang, 2010a). Stochastic algorithms have many subcategories, and metaheuristic algorithms are at the top of these categories. Metaheuristic algorithms are simple, easy, adaptable to all kinds of problems, and non-derivative. Metaheuristic algorithms have been developed by researchers in categories such as physics, biology, swarm, evolutionary-based (Mirjalili et al., 2014; Sharma & Tripathi, 2022). Examples of today's popular metaheuristic algorithms are Arithmetic optimization algorithm (AOA), salp swarm algorithm (SSA), particle swarm optimization algorithm (PSO), genetic algorithm (GA), firefly algorithm (FFA), cuckoo search algorithm (CS) and jaya algorithm (JAYA) can be given (Abualigah et al., 2021; Dede et al., 2020; Holland, 1992; Kennedy & Eberhart, 2007; Mirjalili et al., 2017; Rao, 2016; Yang & Deb, 2009; Yang, 2010b).

* Corresponding author
E-mail: erdemirerkan@gmail.com (E. Erdemir)
ISSN 1923-2934 (Online) - ISSN 1923-2926 (Print)
2023 Growing Science Ltd.
doi: 10.5267/j.ijiec.2023.1.002

A hybrid algorithm is a type of algorithm created using a cooperative or integrative structure, which is one of the hybrid algorithm types of two or more algorithms. Algorithms included in the hybrid algorithm in a cooperative structure try to optimize the problem by working in parallel or sequentially. In the integrative structure, the structure is created by adding or editing the part or code of one algorithm to the structure of the other algorithm. The integrative algorithm created in the new situation is expected to optimize the problem in the best way (Talbi, 2002; Ting et al., 2015; Yılmaz et al., 2022a). Researchers can develop hybrid algorithms using one or more of the above models.

Some of the reasons why researchers developed hybrid algorithms are as follows:

Exploration and exploitation stages are of great importance in metaheuristic algorithms. Exploration is the stage in which solutions are determined so that the problem can reach the global optimum in the search space. Exploitation, on the other hand, is the process of choosing the best solution from the discovery. The process done in the exploitation stage is also more limited due to the discovery stage, that is, it is also expressed as the stage of reaching the local optimum. In principle, the exploration-exploitation pair should work in harmony in metaheuristic algorithms. Otherwise, this situation affects the performance of the algorithm. Therefore, it is necessary to reconsider the exploration and exploitation stages of the algorithm (Alba & Dorronsoro, 2005; Akyol, 2021; Črepinšek et al., 2013). For example, an algorithm that is weak in terms of exploration can become more efficient by using the exploration phase of a different algorithm, that is, by hybridizing it.

According to the No Free Lunch (NFL) theorem, not all problems can be solved by adhering to just one algorithm. While an algorithm may reach the closest results to the solution in some problems, it may not achieve the desired result in others. The method of the algorithm developed to solve a problem may lead to the opposite (undesirable) results in solving the other problem (Wolpert & Macready, 1997). However, using two or more algorithms as cooperative or integrative hybrids can contribute to better optimization of more problems.

Some of the current hybrid metaheuristic algorithm researches in the literature are as follows:

Şenel et al. (2019) implemented a new hybrid algorithm by placing the exploitation part of the PSO in the exploration phase of the grey wolf optimizer (GWO). The developed hybrid algorithm has been tested on benchmark functions, process flowsheeting problem, parameter estimation for frequency modulated sound waves and leather nesting problem. They found that the hybrid algorithm was successful compared to the algorithms they compared. Baş (2021) performed the hybrid process using the tree seed algorithm (TSA) in order for AOA to achieve more successful results. It used TSA's seed production method in the random walking phase of AOA and led to the generation of new candidate solutions. The performance of the hybrid algorithm has been compared first with AOA and then with well-known heuristic algorithms. 13 constrained optimization problems were used in the comparison process. The researcher stated that the hybrid algorithm was successful compared to other algorithms and obtained a statistically significant difference. Ramesh & Manavalan (2021) developed a hybrid algorithm (SSOAGOA) using SSA and grasshopper optimization algorithm (GOA). The developed algorithm was used as a feature selection method to determine the necessary properties to diagnose prostate cancer. SSA hybridized with PSO, GA, GOA, ant colony algorithm (ACO) and whale optimization algorithm (WOA) were used to compare the results. The properties of the features obtained from the hybrid algorithm were sent to the machine learning classifiers. According to the experimental results obtained from the classification algorithms, it has been determined by the researchers that feature selection with SSOAGOA is more successful than other hybrid algorithms. Wang et al. (2021), created a new hybrid algorithm using aquila optimizer (AO) and harris hawks optimizer (HHO). The hybrid algorithm used in solving 23 benchmarking problems and 4 industrial engineering design problems was compared with 5 algorithms. As a result, the researchers determined that the hybrid algorithm is more successful than other algorithms. Yılmaz et al. (2022b) trained the learning processes of multilayer perceptrons with a hybrid algorithm they prepared using multiverse optimization algorithm (MVO) and simulated annealing algorithm (SA). They used 12 data sets in their experimental studies. The researchers compared the experimental analysis results with the results obtained from the multilayer perceptron trained with 12 algorithms. As a result, they determined that the hybrid algorithm achieved superior and successful results compared to other algorithms.

Purpose of this study is to develop a new hybrid algorithm (HSSAOA) with high performance and accuracy, minimized error rate, and statistically proven success by using SSA and AOA. With HSSAOA, which is an integrative hybrid algorithm type, the exploration phase of AOA is adapted to a part of SSA. It is thought that HSSAOA, which is tested in optimizing benchmark functions, will be a more successful hybrid algorithm than SSA, AOA and other popular metaheuristic algorithms used in the study.

The contents of the article are explained in 5 sections and these contents are as follows:

In Section 1 (Introduction), information about optimization, algorithm, deterministic and stochastic algorithm, metaheuristic algorithm, hybrid algorithm and its types, current researches in the literature with hybrid algorithm and purpose of the study were explained.

In Section 2 (Overview), the SSA and AOA used in the development of the HSSAOA were described.

In Section 3 (Proposed Hybrid Algorithm) HSSAOA, which consists of hybridizing SSA and AOA, was mentioned. The flow diagram of the developed hybrid algorithm and its equations adapted according to the new situation were given.

In Section 4 (Experimental Results), information was given about which algorithms were used in comparison, benchmark (unimodal and multimodal) functions used in comparison analysis, number of search agents/iterations/independent runs and parameters of algorithms. Then, comparison results of algorithms, logarithmic convergence curves and statistical results were included.

In Section 5 (Conclusions and Future Studies), the success of the developed HSSAOA was discussed. In addition, information was given about the suggestions that will guide researchers who want to deal with HSSAOA in the future.

2. Overview

2.1. SSA

SSA was developed by Mirjalili et al. (2017), taking into account the lifestyle and food supply system of salp live's. Salps are transparent, like a jelly-fish and lead their lives in chains at deeps in the seas and oceans. There are leader and follower salps in the structure of each salp swarm. The leader is responsible for finding the food. The follower salps follow the leader salp and supply their own food. The leader salp updates its position each time it approaches the food. The follower salps update their position depending on the leader. In SSA, during the exploration process, it is tried to find the positions where the food is in the search space, while in the exploitation process, the existing positions are compared with the neighboring positions to determine the best position to reach the food from these (Bairathi & Gopalani, 2019; Faris et al., 2020; Mirjalili et al., 2017).

Mirjalili et al. (2017) explained the mathematical expressions of the salp swarm algorithm as follows;

The positions of the salps are held in an $SalpP$ matrix of size $N\text{x}d$ (search agent x search space). First, the positions of the salps are randomly assigned in the $lb - ub$ (lower bound-upper bound) range determined at the beginning. Eq. (1) contains the mathematical expression in which the leader's position is updated.

$$SalpP_j^1 = \begin{Bmatrix} FoodP_j + c_1 \text{ x } \left(\left(ub_j - lb_j \right) \text{x } c_2 + lb_j \right) & c_3 \geq 0{,}5 \\ FoodP_j - c_1 \text{x} \left(\left(ub_j - lb_j \right) \text{x } c_2 + lb_j \right) & c_3 < 0{,}5 \end{Bmatrix} \tag{1}$$

According to this equation, $SalpP_j^1$ is the leading salp position, $FoodP_j$ is the information of the food position in the j_{th} dimension, lb_j is the lower bound of the j_{th} dimension, ub_j is the upper bound of the j_{th} dimension and c_2-c_3 are a random number between 0 and 1. In the first article about SSA, the c_3 value is 0. In this case, a part of the equation cannot be used. Therefore, in a study in which Mirjalili is among them, the value of c_3 is given as 0.5 and it is included in the equation in this way (Faris et al., 2020; Mirjalili et al., 2017). The c_1 parameter is one of the main values for Eq.(1). It equilibrates the exploration and exploitation phase, which is calculated as in Eq.(2)

$$c_1 = 2e^{-\left(\frac{4itr}{Mitr}\right)^2} \tag{2}$$

According to this equation, the term e represents the euler number, itr equals the current iteration value and $Mitr$ equals the maximum iteration value. In Eq.(3), there is a position update of the follower salps. According to this equation, $SalpP_j^i$ represents the follower salp and $SalpP_j^{i-1}$ represents the previous follower salp.

$$SalpP_j^i = (SalpP_j^i + SalpP_j^{i-1})/2 \tag{3}$$

In the literature, there are SSA studies using multiple leader salp instead of a single leader salp. The use of multiple leader salp structure increased the randomness in the algorithm and positively affected the algorithm. However, this situation had a negative effect by increasing the instability of the algorithm. Some researchers have proposed an algorithm structure in which the leader (N/2) and follower (N/2) salps are accepted as half and half in order to benefit from the positive aspect of this situation and to be less affected by the negative aspects (Wang et al., 2018; Zhang et al., 2019). In addition, this situation is also seen in the codes of the researchers who developed the SSA (Faris et al., 2016a; Mirjalili, 2018). Fig.1 shows the flowchart of the SSA prepared according to multiple leader salp. In this study, multiple leader salp structures were used for SSA and HSSAOA.

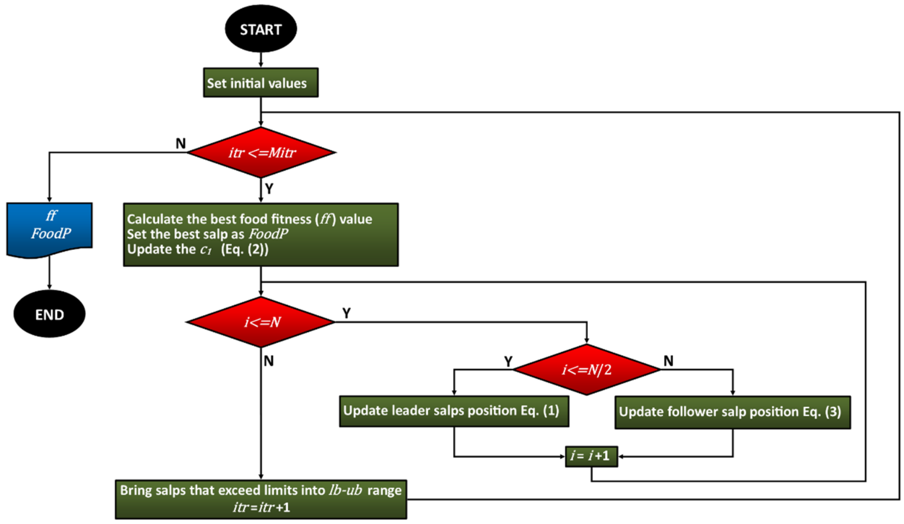

Fig. 1 Flowchart of SSA -based on multiple leader salps- (Faris et al., 2016a; Mirjalili, 2018; Wang et al., 2018; Zhang et al., 2019)

2.2. AOA

Developed by Abualigah et al. (2021), AOA is a population-based, structurally simple metaheuristic algorithm that updates its solutions through mathematical operators in the exploration and exploitation stages. In the exploration phase, using multiplication and division based equations, it searches for solutions in the search space of the problem by means of search agents without falling into the local optimum trap. In the exploitation phase, it provides to improve the solutions obtained in the exploration phase with addition and subtraction based equations.

Abualigah et al. (2021) explained the algorithmic steps of AOA as follows:

Initially, an x matrix of size Nxd (search agent x search space) is created in which the candidate solutions are held. Random values are assigned to this x matrix within the specified range $lb - ub$ (lower bound - upper bound). Values are assigned to the parameters required for the algorithm. The Math Optimizer Accelerated (MOA) function in Eq. (4) is used for the selection of the exploration or exploitation search phase of the algorithm. The value calculated from the MOA function is a value that increases incrementally from 0.2 to 0.9 during the iteration.

$$MOA\ (itr) = min + itr \text{ x } (max - min)/Mitr \tag{4}$$

In this equation $MOA(itr)$ represents the value of the current iteration of the MOA function, itr equals the current iteration value, $Mitr$ equals the maximum iteration value, min and max values represent the minimum and maximum values that the MOA function can take. The result obtained from this function should be calculated before the algorithm starts the processing process. The result is compared with a random value (r_1) from 0-1 range.

If r_1 is greater than the MOA function result, the exploration phase works. In the exploration phase, solutions are calculated using Eq. (5).

$$x_j^i(itr+1) = \begin{cases} D = best(x_j)\ /\ (MOP + \epsilon) \text{ x } \left((ub_j - lb_j) \text{ x } \mu + lb_j\right) & r_2 > 0{,}5 \\ M = best(x_j) \text{ x } MOP \text{ x } \left((ub_j - lb_j) \text{ x } \mu + lb_j\right) & r_2 \leq 0{,}5 \end{cases} \tag{5}$$

According to this equation, $x_j^i(itr+1)$ is the value of the next solution of the j_{th} position of the i_{th} solution, $best(x_j)$ is the best solution of x in the j_{th} position, lb_j and ub_j are the lower and upper bounds at the j_{th} position, ϵ parameter is a small integer value, parameter r_2 is a random number between 0-1 and parameter μ is a constant value used to control the search process. If r_2 is greater than 0.5, division arithmetic operator based equation (D) is used, otherwise multiplication based arithmetic operator based equation (M) is used. The Math Optimizer (MOP) value is calculated as in Eq. (6).

$$MOP = 1 - (itr^{1/\propto}/Mitr^{1/\propto}) \tag{6}$$

According to this equation, itr equals the current iteration value, $Mitr$ equals the maximum iteration value and $\propto$ represents an important and sensitive value.

If r_1 is less than or equal to the MOA function result, the exploitation phase runs. In the exploitation phase, solutions are calculated using Eq. (7).

$$x_j^i(itr+1) = \begin{cases} S = best(x_j) - MOP \text{ x } \left((ub_j - lb_j) \text{ x } \mu + lb_j\right) & r_3 > 0{,}5 \\ A = best(x_j) + MOP \text{ x } \left((ub_j - lb_j) \text{ x } \mu + lb_j\right) & r_3 \leq 0{,}5 \end{cases} \quad (7)$$

According to this equation, The r_3 parameter is a random number between 0 and 1. If r_3 is greater than 0.5, substraction arithmetic operator based equation (S) is used, otherwise addition based arithmetic operator based equation (A) is used. Other parameters ($x_j^i(itr+1)$, $best(x_j)$, MOP, ub_j, lb_j and µ) are used as in Eq. (5). Fig.2 shows the flowchart of the AOA.

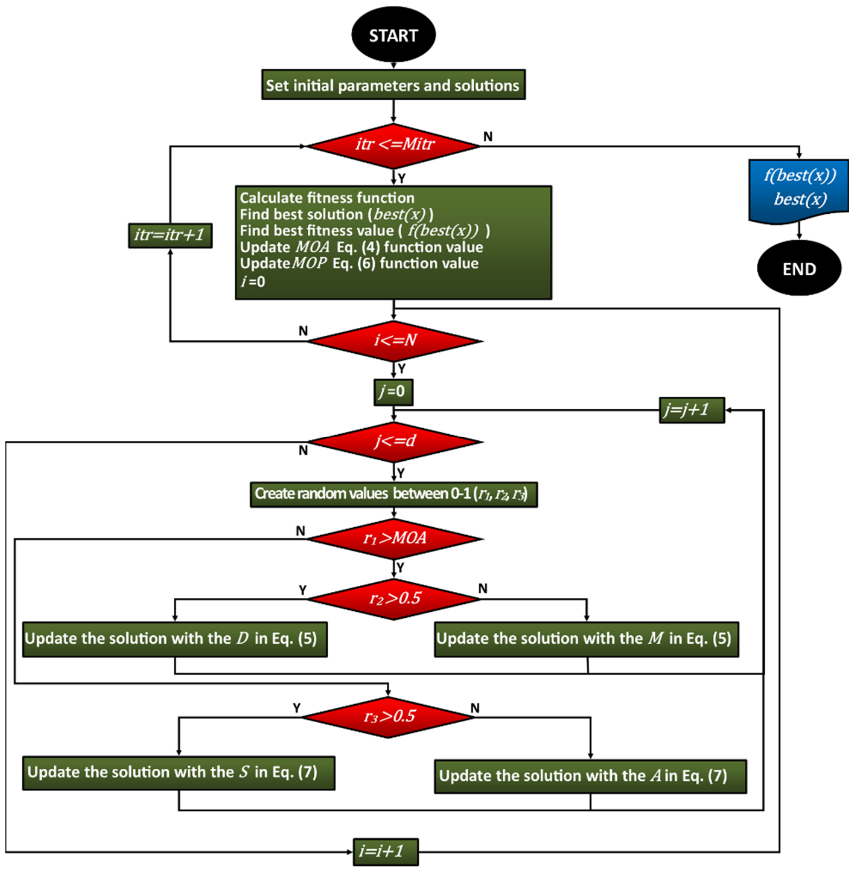

Fig. 2. Flowchart of AOA (Abualigah et al., 2021)

3. Proposed Hybrid Algorithm (HSSAOA)

In this developed algorithm, some changes have been made in the update method of the leader salps/salps. The division and multiplication arithmetic operator-based equations in the exploration phase of AOA have been adapted to replace the equations required for the leader update of the SSA. The MOA function, the max and min parameters of the MOA function, and the $\propto$ and μ parameters are used in the hybrid algorithm as they are used in the AOA. As in SSA and AOA, itr and $Mitr$ values are iteration and maximum iteration values, respectively. The way the algorithm works is as follows:

First of all, the positions of the salps are randomly assigned as in the SSA. Initial parameters of SSA and AOA are defined. In order to compare the values obtained in the later stages of the algorithm, the (best) food position and accordingly the fitness value are calculated from the randomly assigned positions. Position update of leader salp/salps is as in Eq. (8).

$$SalpP_j^i = \begin{cases} D = FoodP_j \;/\; \left((MOP_{new}/c_{1new}) \text{ x } \left((ub_j - lb_j) \text{ x } \mu + lb_j\right)\right) & c_2 < MOA \\ M = FoodP_j \text{ x } \left((MOP_{new}/c_{1new}) \text{ x } \left((ub_j - lb_j) \text{ x } \mu + lb_j\right)\right) & c_2 \geq MOA \end{cases} \quad (8)$$

According to this equation, $SalpP_j^i$ represents the position of the leader salps (if the leader is one, "1" is used instead of i), $FoodP_j$ represents the information of the food position in the j_{th} dimension, lb_j represents the lower bound of the j_{th} dimension, ub_j represents the upper bound of j_{th} dimension and c_2 represents a random number between 0 and 1. The use of μ parameter is the same as used in AOA. c_{1new} is an updated version of c_1 in SSA and MOP_{new} is an updated version of MOP in AOA. The equation for c_{1new} is in Eq. (9) and the equation for MOP_{new} is in Eq. (10).

$$c_{1new} = 2e^{-\left(\frac{4(itr-1)}{Mitr}\right)^2} \tag{9}$$

$$MOP_{new} = 1 - \left(\frac{(itr-1)^{\frac{1}{\propto}}}{Mitr^{\frac{1}{\propto}}}\right) \tag{10}$$

When the new equations are examined, iteration value (itr) has been updated as $itr - 1$, provided that it starts from 1. According to Eq. (8), if c_2 is less than the value obtained from the MOA function, the division arithmetic operator-based equation (D) is used. If c_2 is greater than and equal to the value obtained from the MOA function, the multiplication arithmetic operator-based equation (M) is used. The MOA function is calculated according to Eq. (4). Position update of follower salps is as used in SSA (Eq. (3)). Fig. 3 shows the flowchart of the HSSAOA prepared according to the multiple leader salp used in the study.

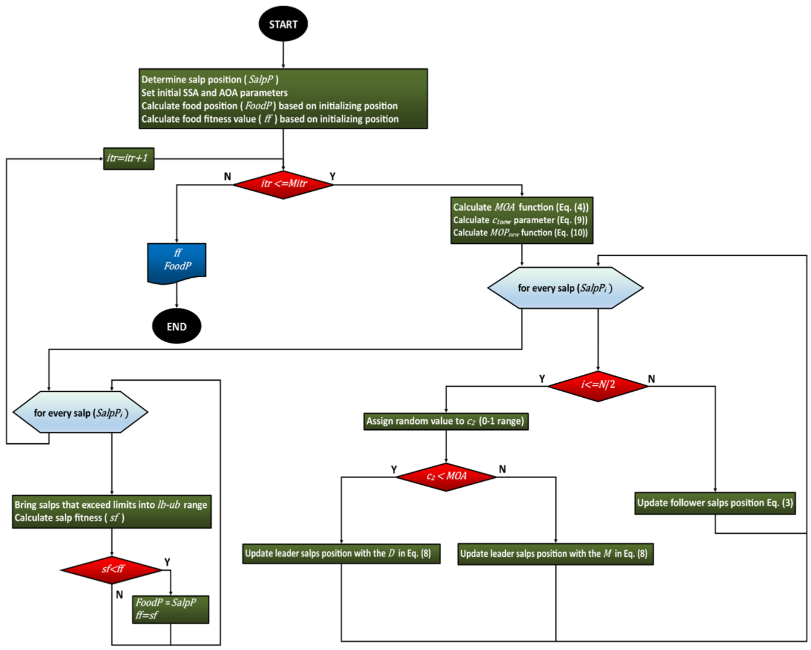

Fig. 3. Flowchart of HSSAOA

4. Experimental Results

In order to determine the success of HSSAOA, it should be run by the number of determined search agents, the number of iterations and the number of independent tries and the results should be compared with other popular metaheuristic algorithms. In order for the results to be confirmed as successful, it should be determined that there is a statistically significant difference. In this study, HSSAOA and other popular metaheuristic algorithms (AOA, SSA, PSO, GA, FFA, CS and JAYA) were compared using a total of 22 benchmark functions, 12 unimodal and 10 multimodal. Each algorithm was analyzed separately using the number of 30, 50 and 100 search agents. The number of iteration used in the analysis was 100. Each algorithm was run 30 times independently for the number of search agents and iterations, and the averages and standard deviations of the results were calculated. In addition, the fitness values obtained by some optimized benchmark functions during the iteration are shown in the logarithmic convergence curve. In order to better understand the results of the mean and standard deviation of the benchmark function in the tables (Table3, Table 5 and Table 7), the results were normalized between 0 and 1 (Mirjalili et al., 2017). In addition, the wilcoxon rank sum test, which is one of the non-parametric statistical tests, was applied to prove the statistical success of the successful algorithm compared to other algorithms (making a statistically significant difference). The algorithm that was successful as a result of each function was compared with other algorithms. In order for an algorithm to be statistically successful, that is, to express a statistically significant difference, it is required that the statistical result obtained (p) be less than 5E-02 (Derrac et al., 2011; Mirjalili et al., 2017). For the analysis of the algorithms, the EvoloPy framework, written in Python, which analyzes metaheuristic algorithms in areas such as optimization, clustering and artificial neural network, was used. AOA, which is not included in the EvoloPy framework, and the developed hybrid algorithm were adapted to this framework (Faris et al., 2016a; Faris et al., 2016b; Khurma et al., 2020; Qaddoura et al., 2020). In algorithms, there may be parameters calculated randomly or with equations, as well as parameters with constant values. The values of the parameters ($\propto$ and μ) with constant values in the AOA and HSSAOA, the values written in the code of the AOA was used (Abualigah, 2022). Constant values parameters of other algorithms are the same as in the EvoloPy framework (Faris et al., 2016a). Parameters with constant values according to the algorithms and their values are given in Table 1.

Table 1
Parameters, explanations and values of HSSAOA and other algorithms

Algorithm	Parameter	Explanation	Value
HSSAOA		The same as AOA	
AOA	$\propto$	Important and sensitive parameter affecting the MOP function	5
	μ	Parameter controlling the search process	0.499
SSA		No constant parameters	
PSO	$Vmax$	Maximum particle velocity	6
	$Vmin$	Minimum particle velocity	$-Vmax$
	$wMax$	Maximum value of inertia weight	0.9
	$wMin$	Minimum value of inertia weight	0.2
	c_1	1st acceleration coefficient	2
	c_2	2 nd acceleration coefficient	2
CS	β	Scale factor	1.5
	P_a	Discovery probability	0.25
FFA	β_{min}	Minimum parameter of attractiveness	0.2
	γ	Coefficent of absorption	1
	$\propto$	Parameter of randomization	0.5
	β_0	Initial attractiveness parameter	1
GA	$keep$	Elitism rate	2
	c_p	Possibility of crossover	1
	m_p	Possibility of mutation	0.01
JAYA		No constant parameters	

The name and equation of the benchmark functions used in this study, modal type, lower bound (lb), upper bound (ub), dimension (dim) and default minimum fitness values ($fmin$) are given in Table 2 (Fletcher & Powell, 1963; Gavana, 2013; Hussain et al., 2017; Jamil & Yang, 2013; Naik et al., 2016).

Table 2
Benchmark functions used in the study

Function Name	Equation	lb	ub	dim	$fmin$
Schwefel 1.2[U]	$F_1(x) = \sum_{i=1}^{d}\left(\sum_{j=1}^{i} x_j\right)^2$	-100	100	20	0
Schwefel 2.20[U]	$F_2(x) = \sum_{i=1}^{d} \lvert x_i \rvert$	-100	100	20	0
Schwefel 2.21[U]	$F_3(x) = max\lvert x_i \rvert. \quad 1 \le i \le d$	-100	100	20	0
Schwefel 2.22[U]	$F_4(x) = \sum_{i=1}^{d} \lvert x_i \rvert + \prod_{i=1}^{d} \lvert x_i \rvert$	-100	100	20	0
Schwefel 2.23[U]	$F_5(x) = \sum_{i=1}^{d} x_i^{10}$	-10	10	20	0
Step[U]	$F_6(x) = \sum_{i=1}^{d} (\lfloor \lvert x_i \rvert \rfloor)$	-100	100	20	0
Step 2[U]	$F_7(x) = \sum_{i=1}^{d} (\lfloor x_i + 0{,}5 \rfloor)^2$	-100	100	20	0
Matyas[U]	$F_8 = 0{,}26(x_1^2 + x_2^2) - 0{,}48x_1x_2$	-10	10	2	0
Wayburn Seader 1[U]	$F_9(x) = (x_1^6 + x_2^4 - 17)^2 + (2x_1 + x_2 - 4)^2$	-5	5	2	0
Sum Squares[U]	$F_{10}(x) = \sum_{i=1}^{d} ix_i^2$	-10	10	20	0
Sphere Model[U]	$F_{11}(x) = \sum_{i=1}^{d} x_i^2$	-5.12	5.12	20	0
Brent[U]	$F_{12}(x) = (x_1 + 10)^2 + (x_2 + 10)^2 + e^{-(x^2+y^2)}$	-20	0	2	0

Table 2

Benchmark functions used in the study (Continued)

Function Name	Equation	lb	ub	dim	$fmin$
Rastrigin[M]	$F_{13} = 10d + \sum_{i=1}^{d} [x_i^2 - 10\cos(2\pi x_i)]$	-5.12	5.12	20	0
Ackley[M]	$F_{14}(x) = -20exp\left(-0{,}2\sqrt{\frac{1}{d}\sum_{i=1}^{d} x_i^2}\right) - exp\left(\frac{1}{d}\sum_{i=1}^{d}\cos(2\pi x_i)\right) + 20 + exp(1)$	-32	32	20	0
Griewank[M]	$F_{15}(x) = \sum_{i=1}^{d}\frac{x_i^2}{4000} - \prod_{i=1}^{d} cos\left(\frac{x_i}{\sqrt{i}}\right) + 1$	-600	600	20	0
Pathological[M]	$F_{16}(x) = \sum_{i=1}^{d-1}\left(0{,}5 + \frac{sin^2\sqrt{100x_i^2 + x_{i+1}^2} - 0{,}5}{1 + 0{,}001(x_i^2 - 2x_i x_{i+1} + x_{i+1}^2)^2}\right)$	-100	100	20	0
Price 4[M]	$F_{17}(x) = (2x_1^3 x_2 - x_2^3)^2 + (6x_1 - x_2^2 + x_2)^2$	-500	500	2	0
Helical Valley[M]	$x_1 \geq 0 \Rightarrow \theta = \frac{\arctan\left(\frac{x_2}{x_1}\right)}{2\pi} \quad x_1 < 0 \Rightarrow \theta = \frac{\arctan\left(\frac{x_2}{x_1}\right) + \pi}{2\pi}$ $F_{18}(x) = 100\left[(x_3 - 10\theta)^2 + \left(\sqrt{x_1^2 + x_2^2} - 1\right)^2\right] + x_3^2$	-10	10	3	0
Csendes[M]	$F_{19}(x) = \sum_{i=1}^{d} x_i^6 (2 + sin\frac{1}{x_i})$	-1	1	20	0
Alpine 1[M]	$F_{20}(x) = \sum_{i=1}^{d} \lvert x_i sin(x_i + 0{,}1x_i)\rvert$	-10	10	20	0
Amgm[M]	$F_{21}(x) = \left[\frac{1}{n}\sum_{i=1}^{n}\lvert x_i\rvert - \left(\prod_{i=1}^{n}\lvert x_i\rvert\right)^{\frac{1}{n}}\right]^2$	0	10	20	0
Salomon[M]	$F_{22}(x) = 1 - cos\left(2\pi\sqrt{\sum_{i=1}^{d} x_i^2}\right) + 0{,}1\sqrt{\sum_{i=1}^{d} x_i^2}$	-100	100	20	0

U:Unimodal   M: Multimodal

The results of the algorithms analyzed with different search agents, the same iteration and the number of independent runs were grouped and named. Named Group 1 for 30 search agents/100 iterations/30 independent runs, Group 2 for 50 search agents/100 iterations/30 independent runs, and Group 3 for 100 search agents/100 iterations/30 independent runs. Table 3 contains the average and standard deviation results of the optimized benchmark functions according to Group 1.

Table 3

Normalized (0-1) average (Ave.) and standard deviation (Std.) results of optimized benchmark functions according to Group 1

F_x	HSSAOA		SSA		AOA		PSO		GA		FFA		CS		JAYA	
	Ave.	**Std.**	**Ave.**	**Std.**	**Ave.**	**Std.**	**Ave.**	**Std.**	**Ave.**	**Std.**	**Ave.**	**Std.**	**Ave.**	**Std.**	**Ave.**	**Std.**
F_1	**0.00E+00**	**0.00E+00**	1.76E-01	2.97E-01	3.35E-07	6.89E-07	1.19E-02	1.37E-02	1.00E+00	7.48E-01	3.65E-01	5.11E-01	5.78E-01	4.78E-01	8.50E-01	1.00E+00
F_2	**0.00E+00**	**0.00E+00**	2.83E-01	5.82E-01	2.13E-04	8.16E-04	8.18E-03	2.03E-02	1.00E+00	1.00E+00	8.77E-02	5.15E-01	6.97E-01	5.72E-01	8.42E-02	2.08E-01
F_3	**0.00E+00**	**0.00E+00**	3.00E-01	6.02E-01	5.20E-04	2.27E-03	3.96E-02	1.31E-01	1.00E+00	1.00E+00	2.07E-01	8.58E-01	6.45E-01	6.31E-01	6.47E-01	8.81E-01
F_4	**0.00E+00**	**0.00E+00**	4.76E-08	6.46E-08	1.93E-19	3.38E-20	1.93E-16	4.78E-17	1.00E+00	1.00E+00	6.81E-09	1.02E-08	4.22E-03	2.81E-03	4.51E-06	5.60E-06
F_5	**0.00E+00**	**0.00E+00**	4.66E-06	7.25E-06	4.73E-38	1.62E-37	3.75E-07	5.61E-07	1.00E+00	1.00E+00	3.31E-08	8.29E-08	4.65E-03	4.12E-03	7.03E-04	9.06E-04
F_6	**0.00E+00**	**0.00E+00**	2.56E-01	6.25E-01	2.36E-04	8.48E-04	8.52E-03	1.62E-02	1.00E+00	1.00E+00	6.66E-02	4.47E-01	7.36E-01	5.29E-01	9.33E-02	2.06E-01
F_7	7.23E-04	**0.00E+00**	2.44E-02	4.18E-02	6.10E-04	3.64E-05	**0.00E+00**	8.19E-05	1.00E+00	1.00E+00	3.93E-04	3.12E-03	3.97E-01	3.45E-01	1.96E-02	5.58E-02
F_8	2.89E-299	**0.00E+00**	9.17E-12	5.37E-12	**0.00E+00**	**0.00E+00**	1.36E-09	2.13E-09	1.00E+00	1.00E+00	4.10E-07	1.91E-07	5.28E-06	3.62E-06	2.37E-02	4.14E-02
F_9	1.86E-01	1.62E-01	1.48E-04	3.73E-04	1.00E+00	1.00E+00	**0.00E+00**	**0.00E+00**	4.62E-03	4.94E-03	3.93E-06	1.06E-05	4.31E-05	4.67E-05	2.11E-04	4.18E-04
F_{10}	**0.00E+00**	**0.00E+00**	5.10E-02	8.51E-02	1.41E-09	5.38E-09	1.07E-02	1.04E-01	1.00E+00	1.00E+00	3.19E-02	8.22E-02	3.43E-01	2.73E-01	1.60E-02	4.53E-02
F_{11}	**0.00E+00**	**0.00E+00**	3.43E-02	7.09E-02	1.37E-09	1.05E-08	2.13E-02	6.81E-02	1.00E+00	1.00E+00	5.08E-03	1.56E-02	3.74E-01	3.77E-01	2.10E-02	6.38E-02
F_{12}	1.27E-01	1.32E-01	3.52E-13	2.77E-13	1.00E+00	1.00E+00	**0.00E+00**	**0.00E+00**	2.57E-03	2.53E-03	3.29E-09	2.55E-09	1.49E-08	1.45E-08	1.22E-02	7.41E-03
F_{13}	**0.00E+00**	**0.00E+00**	3.14E-01	6.17E-01	1.35E-08	7.87E-08	7.81E-01	1.00E+00	8.05E-01	6.78E-01	4.95E-01	9.11E-01	1.00E+00	4.57E-01	8.66E-01	9.90E-01
F_{14}	**0.00E+00**	**0.00E+00**	3.90E-01	1.00E+00	1.93E-04	1.96E-03	1.13E-01	3.37E-01	1.00E+00	7.43E-01	1.29E-01	5.16E-01	9.09E-01	7.09E-01	3.47E-01	9.76E-01
F_{15}	**0.00E+00**	**0.00E+00**	4.42E-02	5.45E-02	1.81E-01	1.00E+00	1.64E-01	1.35E-01	1.00E+00	8.83E-01	3.75E-03	1.42E-02	3.68E-01	3.37E-01	3.86E-02	4.54E-02
F_{16}	**0.00E+00**	**0.00E+00**	8.18E-01	3.95E-01	2.84E-02	1.57E-01	8.88E-01	4.69E-01	7.88E-01	3.34E-01	9.36E-01	5.79E-01	1.00E+00	2.60E-01	6.50E-01	1.00E+00
F_{17}	**0.00E+00**	**0.00E+00**	1.32E-10	9.77E-11	7.06E-11	6.12E-11	1.21E-08	1.96E-08	1.00E+00	1.00E+00	6.87E-10	1.09E-09	5.66E-08	3.93E-08	6.54E-04	1.04E-03
F_{18}	1.00E+00	1.00E+00	2.23E-02	3.54E-02	8.80E-01	6.94E-01	**0.00E+00**	**0.00E+00**	1.28E-01	1.13E-01	1.02E-02	1.89E-02	3.25E-03	2.25E-03	1.11E-02	1.65E-02
F_{19}	**0.00E+00**	**0.00E+00**	4.80E-04	1.15E-03	1.63E-25	1.12E-24	1.98E-02	4.55E-02	1.00E+00	1.00E+00	9.58E-03	1.49E-02	7.42E-02	7.79E-02	6.05E-03	1.32E-02
F_{20}	9.62E-150	1.39E-149	1.92E-02	2.16E-02	**0.00E+00**	**0.00E+00**	5.32E-05	8.72E-05	1.00E+00	3.32E-01	2.44E-03	1.12E-03	7.29E-02	2.48E-02	6.04E-01	1.00E+00
F_{21}	**0.00E+00**	2.19E-33	1.71E-03	8.33E-04	4.33E-28	**0.00E+00**	3.04E-04	4.10E-04	6.25E-01	1.26E-01	1.00E+00	1.00E+00	2.30E-01	4.98E-02	2.17E-03	5.37E-04
F_{22}	**0.00E+00**	**0.00E+00**	4.44E-01	7.05E-01	5.89E-03	2.87E-02	6.64E-02	7.47E-02	1.00E+00	1.00E+00	3.18E-01	7.81E-01	8.26E-01	7.17E-01	3.75E-01	6.65E-01

According to the results in Table 3, the hybrid algorithm was successful in a total of 16 benchmark functions. 8 of them were unimodal benchmark functions and the other 8 were multimodal benchmark functions. In the benchmark functions that HSSAOA was successful, the second successful algorithm was AOA with 15 benchmark functions. In benchmark functions where HSSAOA could not reach the best results, AOA (in 2 benchmark functions) and PSO (in 4 benchmark functions) achieved the best results. Other algorithms did not achieve the best results in any benchmark function. Table 4 shows the wilcoxon rank sum test results (p values) of the analysis performed according to Group 1.

Table 4
According to Group 1, wilcoxon rank sum test p values (N/A = not applicable)

F_x	HSSAOA	SSA	AOA	PSO	GA	FFA	CS	JAYA
F_1	**N/A**	2.87E-11	2.87E-11	2.87E-11	2.87E-11	2.87E-11	2.87E-11	2.87E-11
F_2	**N/A**	2.87E-11	2.87E-11	2.87E-11	2.87E-11	2.87E-11	2.87E-11	2.87E-11
F_3	**N/A**	2.87E-11	2.87E-11	2.87E-11	2.87E-11	2.87E-11	2.87E-11	2.87E-11
F_4	**N/A**	2.87E-11	2.87E-11	2.87E-11	2.87E-11	2.87E-11	2.87E-11	2.87E-11
F_5	**N/A**	2.87E-11	2.87E-11	2.87E-11	2.87E-11	2.87E-11	2.87E-11	2.87E-11
F_6	**N/A**	2.87E-11	2.87E-11	2.87E-11	2.87E-11	2.87E-11	2.87E-11	2.87E-11
F_7	5.77E-11	2.87E-11	2.87E-11	**N/A**	2.87E-11	9.19E-06	2.87E-11	2.87E-11
F_8	2.87E-11	2.87E-11	**N/A**	2.87E-11	2.87E-11	2.87E-11	2.87E-11	2.87E-11
F_9	2.87E-11	8.34E-04	2.87E-11	**N/A**	2.87E-11	8.49E-10	2.87E-11	2.98E-06
F_{10}	**N/A**	2.87E-11	2.87E-11	2.87E-11	2.87E-11	2.87E-11	2.87E-11	2.87E-11
F_{11}	**N/A**	2.87E-11	2.87E-11	2.87E-11	2.87E-11	2.87E-11	2.87E-11	2.87E-11
F_{12}	2.87E-11	2.87E-11	2.87E-11	**N/A**	2.87E-11	2.87E-11	2.87E-11	2.87E-11
F_{13}	**N/A**	2.87E-11	2.87E-11	2.87E-11	2.87E-11	2.87E-11	2.87E-11	2.87E-11
F_{14}	**N/A**	2.87E-11	2.87E-11	2.87E-11	2.87E-11	2.87E-11	2.87E-11	2.87E-11
F_{15}	**N/A**	2.87E-11	2.87E-11	2.87E-11	2.87E-11	2.87E-11	2.87E-11	2.87E-11
F_{16}	**N/A**	2.87E-11	2.87E-11	2.87E-11	2.87E-11	2.87E-11	2.87E-11	2.87E-11
F_{17}	**N/A**	1.07E-06	2.87E-11	2.87E-11	2.87E-11	2.87E-11	2.87E-11	2.87E-11
F_{18}	2.87E-11	6.98E-05	3.26E-03	**N/A**	2.87E-11	2.56E-03	6.24E-09	1.63E-04
F_{19}	**N/A**	2.87E-11	2.87E-11	2.87E-11	2.87E-11	2.87E-11	2.87E-11	2.87E-11
F_{20}	2.87E-11	2.87E-11	**N/A**	2.87E-11	2.87E-11	2.87E-11	2.87E-11	2.87E-11
F_{21}	**N/A**	2.87E-11	5.32E-10	2.87E-11	2.87E-11	2.87E-11	2.87E-11	2.87E-11
F_{22}	**N/A**	2.87E-11	2.87E-11	2.87E-11	2.87E-11	2.87E-11	2.87E-11	2.87E-11

When Table 4 is examined, according to Table 3, HSSAOA, AOA and PSO algorithms have created a statistically significant difference compared to other algorithms with which they were statistically compared in benchmark functions they were successful. Table 5 contains the average and standard deviation results of the optimized benchmark functions according to Group 2.

Table 5
Normalized (0-1) average (Ave.) and standard deviation (Std.) results of optimized benchmark functions according to Group 2

F_x	HSSAOA		SSA		AOA		PSO		GA		FFA		CS		JAYA	
	Ave.	Std.	Ave.	Std.	Ave.	Std.	Ave.	Std.	Ave.	Std.	Ave.	Std.	Ave.	Std.	Ave.	Std.
F_1	**0.00E+00**	**0.00E+00**	1.04E-01	1.25E-01	2.34E-07	6.30E-07	6.61E-03	6.37E-03	9.89E-01	6.16E-01	4.06E-01	4.76E-01	7.67E-01	3.82E-01	1.00E+00	1.00E+00
F_2	**0.00E+00**	**0.00E+00**	2.16E-01	5.14E-01	1.93E-04	1.06E-03	5.12E-03	1.27E-02	1.00E+00	1.00E+00	5.06E-02	3.87E-01	8.73E-01	6.64E-01	7.45E-02	1.43E-01
F_3	**0.00E+00**	**0.00E+00**	2.01E-01	4.28E-01	3.61E-04	1.68E-03	2.74E-02	4.62E-02	1.00E+00	8.55E-01	1.98E-01	1.00E+00	8.26E-01	7.15E-01	6.13E-01	9.26E-01
F_4	**0.00E+00**	**0.00E+00**	1.51E-09	1.09E-09	9.32E-18	2.19E-18	4.33E-15	1.57E-15	1.00E+00	1.00E+00	1.38E-03	1.87E-03	3.11E-01	2.03E-01	3.88E-12	4.78E-12
F_5	**0.00E+00**	**0.00E+00**	1.68E-07	2.38E-07	1.18E-41	1.06E-41	4.59E-08	9.69E-08	1.00E+00	1.00E+00	2.86E-09	6.57E-09	4.59E-02	2.67E-02	5.32E-04	7.98E-04
F_6	**0.00E+00**	**0.00E+00**	2.01E-01	5.43E-01	1.83E-04	9.84E-04	5.01E-03	9.59E-03	1.00E+00	1.00E+00	5.50E-02	2.98E-01	9.13E-01	6.35E-01	7.75E-02	1.84E-01
F_7	8.88E-04	2.77E-05	7.02E-03	1.82E-02	7.42E-04	1.10E-04	1.98E-05	2.64E-05	1.00E+00	1.00E+00	**0.00E+00**	**0.00E+00**	6.23E-01	6.00E-01	1.07E-02	2.19E-02
F_8	1.74E-304	**0.00E+00**	1.01E-11	9.72E-12	**0.00E+00**	**0.00E+00**	1.79E-10	5.36E-10	1.00E+00	1.00E+00	1.06E-06	1.02E-06	7.46E-06	7.14E-06	3.57E-02	2.00E-01
F_9	4.59E-01	2.94E-01	7.78E-04	2.00E-03	1.00E+00	1.00E+00	**0.00E+00**	**0.00E+00**	1.93E-02	1.35E-02	3.25E-08	2.52E-08	5.11E-05	3.88E-05	1.24E-02	4.41E-02
F_{10}	**0.00E+00**	**0.00E+00**	1.87E-02	4.84E-02	5.87E-10	4.24E-09	1.83E-03	4.71E-03	1.00E+00	1.00E+00	1.54E-02	5.01E-02	4.78E-01	4.33E-01	1.40E-02	4.15E-02
F_{11}	**0.00E+00**	**0.00E+00**	5.87E-03	1.19E-02	2.02E-10	1.36E-09	1.41E-02	3.46E-02	1.00E+00	1.00E+00	2.73E-03	1.26E-02	6.29E-01	4.65E-01	1.24E-02	3.35E-02
F_{12}	1.21E-01	1.97E-01	3.19E-13	2.97E-13	1.00E+00	1.00E+00	**0.00E+00**	**0.00E+00**	3.30E-03	3.48E-03	5.63E-09	5.99E-09	1.65E-08	2.13E-08	1.49E-02	1.29E-02
F_{13}	**0.00E+00**	**0.00E+00**	2.19E-01	3.86E-01	1.00E-07	2.58E-06	7.47E-01	1.00E+00	7.44E-01	6.63E-01	4.78E-01	7.32E-01	1.00E+00	4.54E-01	9.53E-01	9.13E-01
F_{14}	**0.00E+00**	**0.00E+00**	2.29E-01	5.03E-01	1.10E-04	1.13E-03	6.27E-02	3.76E-01	9.88E-01	8.81E-01	8.47E-02	3.79E-01	1.00E+00	6.32E-01	3.47E-01	1.00E+00
F_{15}	**0.00E+00**	**0.00E+00**	3.17E-02	3.14E-02	9.89E-02	5.74E-01	1.57E-01	1.67E-01	1.00E+00	1.00E+00	2.49E-03	6.63E-03	6.70E-01	7.77E-01	4.27E-02	5.95E-02
F_{16}	**0.00E+00**	**0.00E+00**	7.78E-01	8.32E-01	1.28E-02	8.45E-02	8.51E-01	5.91E-01	7.49E-01	4.42E-01	9.73E-01	4.86E-01	1.00E+00	1.81E-01	6.46E-01	1.00E+00
F_{17}	**0.00E+00**	**0.00E+00**	6.26E-12	3.35E-12	7.83E-11	9.88E-11	1.55E-13	8.07E-14	1.00E+00	1.00E+00	2.57E-12	1.22E-12	2.23E-09	1.14E-09	5.98E-05	7.63E-05
F_{18}	1.00E+00	1.00E+00	1.15E-02	2.01E-02	3.34E-01	5.03E-01	**0.00E+00**	**0.00E+00**	8.54E-02	3.79E-02	4.92E-03	9.18E-03	2.14E-03	1.14E-03	1.25E-02	2.20E-02
F_{19}	**0.00E+00**	**0.00E+00**	5.70E-05	1.55E-04	2.84E-28	1.89E-27	3.15E-03	6.48E-03	1.00E+00	1.00E+00	4.18E-03	6.09E-03	1.88E-01	1.72E-01	5.23E-03	1.31E-02
F_{20}	2.61E-151	5.35E-151	3.57E-02	1.41E-01	**0.00E+00**	**0.00E+00**	3.87E-06	1.01E-05	1.00E+00	1.00E+00	5.97E-03	9.35E-03	6.17E-02	6.15E-02	2.76E-02	3.58E-02
F_{21}	**0.00E+00**	2.72E-34	1.25E-04	7.53E-05	7.63E-28	**0.00E+00**	3.60E-03	5.51E-03	1.00E+00	2.70E-01	4.92E-01	1.00E+00	3.83E-01	8.05E-02	9.61E-04	3.74E-04
F_{22}	**0.00E+00**	**0.00E+00**	3.13E-01	3.35E-01	4.30E-03	2.40E-02	5.64E-02	5.50E-02	1.00E+00	1.00E+00	2.37E-01	6.28E-01	9.44E-01	7.42E-01	3.29E-01	6.44E-01

When Table 5 is examined, HSSAOA was successful in 16 benchmark functions, as in Group 1. 8 of them were unimodal and the other 8 were multimodal benchmark functions. In the benchmark functions that HSSAOA was successful, the second successful algorithm was AOA with 14 benchmark functions. Other algorithms with successful results were AOA (in 2 benchmark functions), PSO (in 3 benchmark functions) and FFA (in 1 benchmark function). As a result of the analysis, other algorithms could not achieve the best results in optimizing the benchmark functions. Table 6 shows the results of the wilcoxon rank sum test of the optimized benchmark functions according to Group 2.

Table 6
According to Group 2, wilcoxon rank sum test *p* values (N/A = not applicable)

F_x	HSSAOA	SSA	AOA	PSO	GA	FFA	CS	JAYA
F_1	**N/A**	2.87E-11	2.87E-11	2.87E-11	2.87E-11	2.87E-11	2.87E-11	2.87E-11
F_2	**N/A**	2.87E-11	2.87E-11	2.87E-11	2.87E-11	2.87E-11	2.87E-11	2.87E-11
F_3	**N/A**	2.87E-11	2.87E-11	2.87E-11	2.87E-11	2.87E-11	2.87E-11	2.87E-11
F_4	**N/A**	2.87E-11	2.87E-11	2.87E-11	2.87E-11	2.87E-11	2.87E-11	2.87E-11
F_5	**N/A**	2.87E-11	2.87E-11	2.87E-11	2.87E-11	2.87E-11	2.87E-11	2.87E-11
F_6	**N/A**	2.87E-11	2.87E-11	2.87E-11	2.87E-11	2.87E-11	2.87E-11	2.87E-11
F_7	2.87E-11	2.87E-11	2.87E-11	7.10E-04	2.87E-11	**N/A**	2.87E-11	2.87E-11
F_8	3.39E-07	2.87E-11	**N/A**	2.87E-11	2.87E-11	2.87E-11	2.87E-11	2.87E-11
F_9	2.87E-11	1.15E-02	2.87E-11	**N/A**	2.87E-11	7.73E-10	2.87E-11	2.20E-05
F_{10}	**N/A**	2.87E-11	2.87E-11	2.87E-11	2.87E-11	2.87E-11	2.87E-11	2.87E-11
F_{11}	**N/A**	2.87E-11	2.87E-11	2.87E-11	2.87E-11	2.87E-11	2.87E-11	2.87E-11
F_{12}	2.87E-11	3.18E-11	2.87E-11	**N/A**	2.87E-11	2.87E-11	2.87E-11	2.87E-11
F_{13}	**N/A**	2.87E-11	2.13E-09	2.87E-11	2.87E-11	2.87E-11	2.87E-11	2.87E-11
F_{14}	**N/A**	2.87E-11	2.87E-11	2.87E-11	2.87E-11	2.87E-11	2.87E-11	2.87E-11
F_{15}	**N/A**	2.87E-11	2.87E-11	2.87E-11	2.87E-11	2.87E-11	2.87E-11	2.87E-11
F_{16}	**N/A**	2.87E-11	2.87E-11	2.87E-11	2.87E-11	2.87E-11	2.87E-11	2.87E-11
F_{17}	**N/A**	2.87E-11	2.87E-11	2.87E-11	2.87E-11	2.87E-11	2.87E-11	2.87E-11
F_{18}	2.87E-11	4.49E-08	8.14E-03	**N/A**	2.87E-11	3.42E-03	5.23E-11	7.90E-05
F_{19}	**N/A**	2.87E-11	3.18E-11	2.87E-11	2.87E-11	2.87E-11	2.87E-11	2.87E-11
F_{20}	2.87E-11	2.87E-11	**N/A**	2.87E-11	2.87E-11	2.87E-11	2.87E-11	2.87E-11
F_{21}	**N/A**	5.32E-10	2.87E-11	2.87E-11	2.87E-11	2.87E-11	2.87E-11	2.87E-11
F_{22}	**N/A**	2.87E-11	2.87E-11	2.87E-11	2.87E-11	2.87E-11	2.87E-11	2.87E-11

According to Table 5, statistical analysis was performed between successful algorithms and other algorithms. According to the results in Table 6, these algorithms obtained statistically significant differences compared to the algorithms they were compared. Table 7 contains the average and standard deviation results of the optimized benchmark functions according to Group 3.

Table 7
Normalized (0-1) average (Ave.) and standard deviation (Std.) results of optimized benchmark functions according to Group 3

F_x	HSSAOA		SSA		AOA		PSO		GA		FFA		CS		JAYA	
	Ave.	Std.	Ave.	Std.	Ave.	Std.	Ave.	Std.	Ave.	Std.	Ave.	Std.	Ave.	Std.	Ave.	Std.
F_1	**0.00E+00**	**0.00E+00**	4.22E-02	5.36E-02	1.37E-07	5.68E-07	4.12E-03	3.67E-03	9.47E-01	7.59E-01	3.23E-01	3.94E-01	1.00E+00	5.12E-01	9.77E-01	1.00E+00
F_2	**0.00E+00**	**0.00E+00**	7.99E-02	3.05E-01	7.77E-05	5.93E-04	2.22E-03	6.64E-03	8.16E-01	8.68E-01	4.21E-02	3.76E-01	1.00E+00	1.00E+00	5.60E-02	1.67E-01
F_3	**0.00E+00**	**0.00E+00**	1.16E-01	2.70E-01	2.14E-04	1.82E-03	2.09E-02	3.21E-02	1.00E+00	1.00E+00	1.16E-01	5.77E-01	9.71E-01	6.89E-01	5.68E-01	8.51E-01
F_4	**0.00E+00**	**0.00E+00**	9.12E-09	1.43E-08	6.24E-18	2.45E-18	4.02E-15	1.93E-15	1.00E+00	1.00E+00	4.34E-09	5.22E-09	2.43E-01	1.44E-01	5.48E-07	8.03E-07
F_5	**0.00E+00**	**0.00E+00**	6.03E-10	2.79E-09	3.75E-42	9.91E-42	9.69E-09	3.89E-08	1.00E+00	1.00E+00	1.49E-10	4.97E-10	3.05E-01	2.89E-01	1.75E-04	5.17E-04
F_6	**0.00E+00**	**0.00E+00**	7.53E-02	3.02E-01	4.78E-05	5.18E-04	2.16E-03	6.51E-03	7.85E-01	1.00E+00	4.27E-02	4.56E-01	1.00E+00	6.14E-01	5.49E-02	1.46E-01
F_7	1.19E-03	1.03E-04	3.29E-04	1.03E-03	9.83E-04	2.67E-04	**0.00E+00**	**0.00E+00**	9.45E-01	1.00E+00	1.02E-05	1.21E-05	1.00E+00	9.15E-01	6.67E-03	1.63E-02
F_8	2.29E-305	**0.00E+00**	1.05E-11	9.07E-12	**0.00E+00**	**0.00E+00**	1.94E-11	3.17E-11	1.00E+00	1.00E+00	9.49E-07	6.42E-07	4.38E-06	4.46E-06	1.76E-02	7.47E-02
F_9	1.71E-01	1.24E-01	7.36E-06	1.94E-05	1.00E+00	1.00E+00	**0.00E+00**	**0.00E+00**	7.98E-03	5.43E-03	5.11E-08	8.95E-08	1.17E-05	8.84E-06	1.10E-04	2.88E-04
F_{10}	**0.00E+00**	**0.00E+00**	4.08E-03	8.22E-03	7.36E-11	1.15E-09	2.38E-02	2.49E-01	1.00E+00	1.00E+00	2.37E-02	7.39E-02	8.40E-01	5.70E-01	1.09E-02	2.88E-02
F_{11}	**0.00E+00**	**0.00E+00**	4.45E-04	1.73E-03	2.20E-10	1.60E-09	3.09E-03	7.62E-03	9.31E-01	8.67E-01	2.25E-03	8.03E-03	1.00E+00	1.00E+00	1.19E-02	5.66E-02
F_{12}	3.47E-02	3.54E-02	4.16E-13	3.41E-13	1.00E+00	1.00E+00	**0.00E+00**	**0.00E+00**	2.65E-03	2.74E-03	1.08E-08	8.91E-09	4.52E-08	6.30E-08	1.36E-02	1.05E-02
F_{13}	**0.00E+00**	**0.00E+00**	1.82E-01	3.10E-01	3.05E-09	2.18E-08	6.05E-01	6.18E-01	6.06E-01	2.94E-01	4.49E-01	4.73E-01	1.00E+00	2.50E-01	8.77E-01	1.00E+00
F_{14}	**0.00E+00**	**0.00E+00**	1.66E-01	3.94E-01	3.80E-05	7.02E-04	2.97E-02	2.70E-01	9.02E-01	1.00E+00	7.42E-02	4.31E-01	1.00E+00	6.85E-01	2.43E-01	4.36E-01
F_{15}	**0.00E+00**	**0.00E+00**	1.77E-02	2.41E-02	9.94E-02	6.39E-01	8.73E-02	8.28E-02	9.48E-01	1.00E+00	2.39E-03	4.61E-03	1.00E+00	7.68E-01	3.91E-02	3.47E-02
F_{16}	**0.00E+00**	**0.00E+00**	6.81E-01	1.00E+00	4.82E-03	7.67E-02	8.26E-01	6.13E-01	6.95E-01	4.81E-01	1.00E+00	5.63E-01	9.73E-01	2.31E-01	5.48E-01	8.52E-01
F_{17}	**0.00E+00**	**0.00E+00**	5.52E-10	3.44E-10	4.42E-10	7.70E-10	1.66E-11	9.25E-12	1.00E+00	1.00E+00	1.81E-09	1.04E-09	1.01E-06	1.09E-06	7.55E-05	1.21E-04
F_{18}	6.84E-01	4.11E-01	1.42E-03	7.72E-04	1.00E+00	1.00E+00	**0.00E+00**	**0.00E+00**	1.39E-01	4.86E-02	9.77E-04	8.60E-04	3.15E-03	1.58E-03	1.42E-02	1.83E-02
F_{19}	**0.00E+00**	**0.00E+00**	1.06E-06	3.58E-06	9.08E-32	3.88E-31	1.68E-04	2.70E-04	1.00E+00	1.00E+00	2.42E-03	5.01E-03	4.82E-01	4.12E-01	5.96E-04	1.87E-03
F_{20}	1.58E-151	1.79E-151	5.20E-04	4.84E-03	**0.00E+00**	**0.00E+00**	9.17E-07	1.93E-06	1.00E+00	1.00E+00	1.06E-02	4.14E-02	4.84E-02	8.95E-02	2.83E-02	1.20E-01
F_{21}	**0.00E+00**	7.28E-36	1.55E-06	5.77E-06	1.33E-27	**0.00E+00**	1.81E-05	1.01E-04	8.89E-01	5.73E-01	1.45E-01	1.00E+00	1.00E+00	4.78E-01	4.06E-04	5.33E-04
F_{22}	**0.00E+00**	**0.00E+00**	1.87E-01	3.73E-01	2.05E-03	1.63E-02	4.31E-02	5.23E-02	7.91E-01	1.00E+00	1.25E-01	2.86E-01	1.00E+00	7.32E-01	2.95E-01	5.65E-01

According to the data in Table 7, HSSAOA achieved the best results in a total of 16 benchmark functions, 8 of which are unimodal and 8 of which were multimodal, as in previous analyzes (Table 3 and Table 5). In the benchmark functions that HSSAOA was successful, the second successful algorithm was AOA with 14 benchmark functions. AOA (in 2 benchmark functions) and PSO (in 4 benchmark functions) were the other two algorithms that achieve the best results. Table 8 shows the results of the wilcoxon rank sum test of the optimized benchmark functions according to Group 3.

Table 8
According to Group 3, wilcoxon rank sum test p values (N/A = not applicable)

F_x	HSSAOA	SSA	AOA	PSO	GA	FFA	CS	JAYA
F_1	**N/A**	2.87E-11	2.87E-11	2.87E-11	2.87E-11	2.87E-11	2.87E-11	2.87E-11
F_2	**N/A**	2.87E-11	2.87E-11	2.87E-11	2.87E-11	2.87E-11	2.87E-11	2.87E-11
F_3	**N/A**	2.87E-11	2.87E-11	2.87E-11	2.87E-11	2.87E-11	2.87E-11	2.87E-11
F_4	**N/A**	2.87E-11	2.87E-11	2.87E-11	2.87E-11	2.87E-11	2.87E-11	2.87E-11
F_5	**N/A**	2.87E-11	2.87E-11	2.87E-11	2.87E-11	2.87E-11	2.87E-11	2.87E-11
F_6	**N/A**	2.87E-11	2.87E-11	2.87E-11	2.87E-11	2.87E-11	2.87E-11	2.87E-11
F_7	2.87E-11	1.04E-10	2.87E-11	**N/A**	2.87E-11	2.60E-04	2.87E-11	2.87E-11
F_8	3.88E-04	2.87E-11	**N/A**	2.87E-11	2.87E-11	2.87E-11	2.87E-11	2.87E-11
F_9	2.87E-11	1.95E-02	2.87E-11	**N/A**	2.87E-11	3.18E-11	2.87E-11	1.09E-03
F_{10}	**N/A**	2.87E-11	2.87E-11	2.87E-11	2.87E-11	2.87E-11	2.87E-11	2.87E-11
F_{11}	**N/A**	2.87E-11	2.87E-11	2.87E-11	2.87E-11	2.87E-11	2.87E-11	2.87E-11
F_{12}	2.87E-11	2.87E-11	2.87E-11	**N/A**	2.87E-11	2.87E-11	2.87E-11	2.87E-11
F_{13}	**N/A**	2.87E-11	2.51E-05	2.87E-11	2.87E-11	2.87E-11	2.87E-11	2.87E-11
F_{14}	**N/A**	2.87E-11	2.87E-11	2.87E-11	2.87E-11	2.87E-11	2.87E-11	2.87E-11
F_{15}	**N/A**	2.87E-11	2.87E-11	2.87E-11	2.87E-11	2.87E-11	2.87E-11	2.87E-11
F_{16}	**N/A**	2.87E-11	2.87E-11	2.87E-11	2.87E-11	2.87E-11	2.87E-11	2.87E-11
F_{17}	**N/A**	2.87E-11	9.19E-06	2.87E-11	2.87E-11	2.87E-11	2.87E-11	2.87E-11
F_{18}	2.87E-11	2.58E-06	1.63E-08	**N/A**	2.87E-11	7.91E-04	2.87E-11	4.91E-06
F_{19}	**N/A**	2.87E-11	7.03E-11	2.87E-11	2.87E-11	2.87E-11	2.87E-11	2.87E-11
F_{20}	2.87E-11	2.87E-11	**N/A**	2.87E-11	2.87E-11	2.87E-11	2.87E-11	2.87E-11
F_{21}	**N/A**	2.87E-11	5.32E-10	2.87E-11	2.87E-11	2.87E-11	2.87E-11	2.87E-11
F_{22}	**N/A**	2.87E-11	2.87E-11	2.87E-11	2.87E-11	2.87E-11	2.87E-11	2.87E-11

When Table 8 is examined, according to Table 7, HSSAOA, AOA and PSO algorithms have created a statistically significant difference compared to other algorithms with which they were statistically compared in benchmark functions they were successful. Fig. 4-6 shows the logarithmic convergence curves of the fitness values of some functions optimized by the algorithms run according to Group 1-3, respectively.

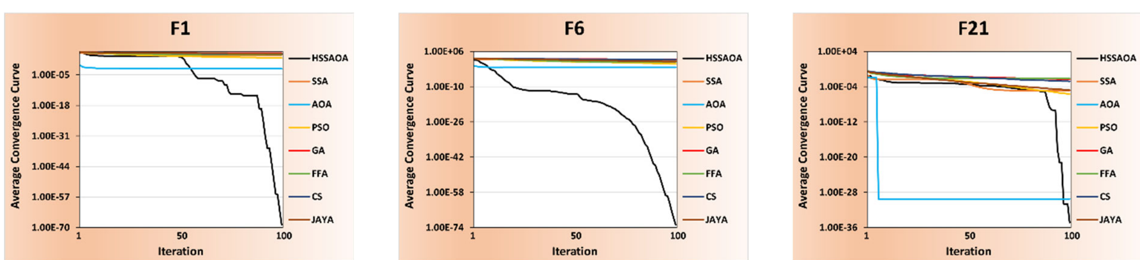

Fig. 4 Logarithmic convergence curve of some functions according to Group 1

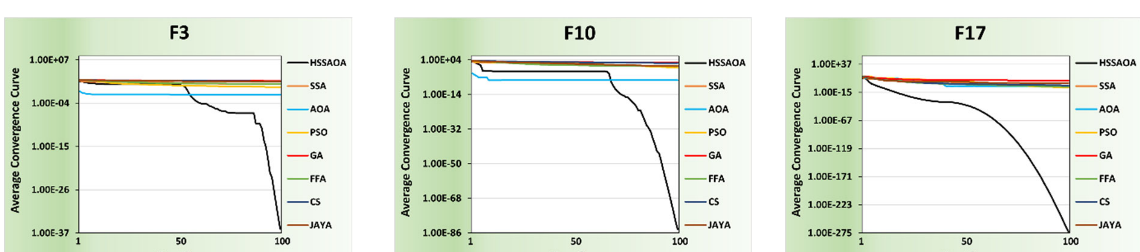

Fig. 5 Logarithmic convergence curve of some functions according to Group 2

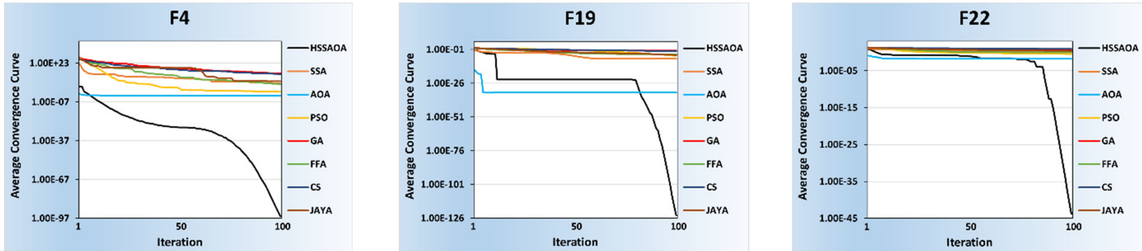

Fig. 6 Logarithmic convergence curve of some functions according to Group 3

When the convergence curves of the functions in Fig. 4-6 are examined, it is seen that HSSAOA has better optimization. While the fitness values of other algorithms become stable after a certain iteration value, the hybrid algorithm continued to improve fitness values until the last iteration value.

5. Conclusions and Future Works

HSSAOA is a hybrid metaheuristic algorithm developed by adapting the exploration phase equations of AOA to replace the SSA's equations for the position update of the leader salp/salps and introducing new updates to the existing structure. The hybrid algorithm achieved the best result in 16 of 22 benchmark functions in each group compared to 7 algorithms compared. According to these results, the algorithms were subjected to the wilcoxon rank sum statistical test. The algorithm that was successful in each group was compared with other algorithms. As a result, HSSAOA obtained a statistically significant difference compared to the other 7 algorithms. This indicates that the algorithm was successful in both stages. This is proof that different metaheuristic algorithms can be hybridized by changing the exploration and/or exploitation stages and better results can be achieved. The developed hybrid algorithm is generally successful. However, there are also benchmark functions where it cannot achieve the best results. In this case, the algorithm can be tried to be improved without changing the structure of the hybrid algorithm. For example, by increasing the number of search agents and the number of iterations or by updating the constant values of the algorithm with different values, it can be observed whether the algorithm is successful in terms of performance without changing the structure of the algorithm. In cases where the hybrid algorithm cannot be successful without changing its structure, it can be determined whether the algorithm is successful in terms of performance by changing the structure of the algorithm. For example, hybridizing the hybrid algorithm with a third algorithm or adding/changing the part taken from the AOA to a different or same part of the SSA can be given as examples of structural change. In short, with the changes made, it is thought that HSSAOA will be a better hybrid metaheuristic algorithm.

In addition, it is thought that the algorithm will be successful in areas such as optimizing different functions, training artificial neural network, clustering and feature selection by using this hybrid algorithm in future studies.

References

Akyol, S. (2021). Global optimizasyon için yeni bir hibrit yöntem: kaya kartalı optimizasyonu-tanjant arama algoritması. *Fırat Üniversitesi Mühendislik Bilimleri Dergisi*, *33*(2), 721-733.

Alba, E., & Dorronsoro, B. (2005). The exploration/exploitation tradeoff in dynamic cellular genetic algorithms. *IEEE Transactions on Evolutionary Computation*, *9*(2), 126-142.

Abualigah, L., Diabat, A., Mirjalili, S., Abd Elaziz, M., & Gandomi, A.H. (2021). The arithmetic optimization algorithm. *Computer Methods in Applied Mechanics and Engineering, 376, 113609*, pages 38.

Abualigah, L. (2022). *The Arithmetic Optimization Algorithm (AOA). MATLAB Central File Exchange*. https://www.mathworks.com/matlabcentral/fileexchange/84742-the-arithmetic-optimization-algorithm-aoa

Bairathi, D., & Gopalani, D. (2019). *Salp swarm algorithm (SSA) for training feed-forward neural networks*, in: Bansal, J., Das, K., Nagar, A., Deep, K. & Ojha, A. (Eds.), Soft Computing for Problem Solving. Advances in Intelligent Systems and Computing, Springer, Singapore, *816*, pp. 521-534.

Baş, E. (2021). Hybrid the arithmetic optimization algorithm for constrained optimization problems. *Konya Mühendislik Bilimleri Dergisi*, *9*(3), 713-734.

Britannica, T. Editors of Encyclopaedia (2022). *Algorithm. Encyclopedia Britannica.* https://www.britannica.com/science/algorithm

Brownlee, J.A. (2021). *Gentle introduction to stochastic optimization algorithms. Machine Learning Mastery.* https://machinelearningmastery.com/stochastic-optimization-for-machine-learning/.

Črepinšek, M., Liu, S.-H., & Mernik, M. (2013). Exploration and exploitation in evolutionary algorithms: A survey. *ACM Comput. Surv*, *45*(3), Article 35, 33 pages.

Dede, T., Grzywiński, M., & Rao, R.V. (2020). *Jaya: A new meta-heuristic algorithm for the optimization of braced dome structures*, in: Rao, R.V., & Taler, J. (Eds.), Advanced Engineering Optimization Through Intelligent Techniques. Advances in Intelligent Systems and Computing, Springer, Singapure, *949*, pp. 13-20.

Derrac, J., García, S., Molina, D., & Herrera, F. (2011). A practical tutorial on the use of nonparametric statistical tests as a methodology for comparing evolutionary and swarm intelligence algorithms. *Swarm and Evolutionary Computation*, *1*(1), 3-18.

Faris, H., Qaddoura, R., Aljarah, I., Bae, J.W., Fouad, M.M., Floden, L., Wolz, D., Bawagan, P., & Merelo-Guervós, J.J. (2016a). *EvoloPy. Github*. https://github.com/7ossam81/EvoloPy

Faris, H., Aljarah, I., Mirjalili, S., Castillo, P.A., & Merelo, J.J. (2016b). *Evolopy: an open-source nature-inspired optimization framework in python*. in: Proceedings Of The 8th International Joint Conference On Computational Intelligence-ECTA (IJCCI 2016). SciTePress, Porto, *1*, pp. 171-177.

Faris H., Mirjalili S., Aljarah I., Mafarja M., & Heidari A.A. (2020). *Salp swarm algorithm: theory, literature review, and application in extreme learning machines*. in: Mirjalili, S., Song Dong, J., & Lewis, A. (Eds.), Studies in Computational Intelligence. Nature-Inspired Optimizers. Springer, Cham, *811,* pp. 185-199.

Fletcher, R., & Powell, M.J.D. (1963). A rapidly convergent descent method for minimization. *The Computer Journal*, *6*(2), 163–168.

Friedrich, T., Kötzing, T., Krejca, M.S., & Rajabi, A. (2022). *Escaping local optima with local search: a theory-driven discussion,* in: International Conference on Parallel Problem Solving from Nature. Springer, Cham, pp. 442-455.

Gavana, A. (2013). *Global Optimization Benchmarks and AMPGO. Test Functions Index.* http://infinity77.net/global_optimization/test_functions.html

Holland, J.H. (1992). Genetic algorithms. *Scientific American*, *267*(1), 66-72.

Hussain, K., Salleh, M.N.M., Cheng, S., & Naseem, R. (2017). Common benchmark functions for metaheuristic evaluation: a review. *JOIV: International Journal on Informatics Visualization*, *1*(4-2), 218-223.

Jamil, M., & Yang, X.-S. (2013). A literature survey of benchmark functions for global optimization problems. *International Journal of Mathematical Modelling and Numerical Optimisation (IJMMNO)*, *4*(2), 150-194.

Kennedy, J. & Eberhart, R. (1995). *Particle swarm optimization*, in: Proceedings of ICNN'95 - International Conference on Neural Networks. Perth, WA, Australia, *4*, pp. 1942-1948.

Khurma, R.A., Aljarah, I., Sharieh, A., & Mirjalili, S. (2020). *Evolopy-FS: an open-source nature-inspired optimization framework in python for feature selection*, in: Mirjalili, S., Faris, H. & Aljarah, I. (Eds.), Evolutionary Machine Learning Techniques. Algorithms for Intelligent Systems. Springer, Singapore, pp. 131-173.

Kochenderfer, M.J., & Wheeler, T.A. (2019). *Algorithms for optimization.* Mit Press.

Mirjalili, S., Mirjalili, S.M., & Lewis, A. (2014). Grey wolf optimizer. *Advances in Engineering Software*, *69*, 46-61.

Mirjalili, S., Gandomi, A.H., Mirjalili, S.Z., Saremi, S., Faris, H., & Mirjalili, S.M. (2017). Salp swarm algorithm: A bio-inspired optimizer for engineering design problems. *Advances in Engineering Software*, *114*, 163-191.

Mirjalili, S. (2018). *SSA: Salp swarm algorithm. MATLAB Central File Exchange.* https://www.mathworks.com/matlabcentral/fileexchange/63745-ssa-salp-swarm-algorithm.

Naik, M.K., Samantaray, L., & Panda, R. (2016). *A hybrid CS–GSA algorithm for optimization*, in: Bhattacharyya, S., Dutta, P., & Chakraborty, S. (Eds.), Hybrid Soft Computing Approaches. Studies in Computational Intelligence. Springer, New Delhi, *611*, pp. 3-35.

Qaddoura, R., Faris, H., Aljarah, I., & Castillo, P.A. (2020). *EvoCluster: an open-source nature-inspired optimization clustering framework in python*. in: International Conference on the Applications of Evolutionary Computation (Part of EvoStar). Spain, Seville, pp. 20-36.

Ramesh, J., & Manavalan, R. (2021). Hybrid of salp swarm optimization algorithm and grasshopper optimization algorithm (SSOAGOA) for feature selection. *International Journal of Grid and Distributed Computing*, *14*(1), 1350–1366.

Rao, R.V. (2016). Jaya: A simple and new optimization algorithm for solving constrained and unconstrained optimization problems. *International Journal of Industrial Engineering Computations*, *7*(1), 19-34.

Sergeyev, Y.D., Kvasov, D.E., & Mukhametzhanov, M.S. (2017). Operational zones for comparing metaheuristic and deterministic one-dimensional global optimization algorithms. *Mathematics and Computers in Simulation*, *141*, 96-109.

Sharma, V., & Tripathi, A.K. (2022). A systematic review of meta-heuristic algorithms in IoT based application. *Array*, *14*, *100164*, pages 8.

Şenel, F.A., Gökçe, F., Yüksel, A.S., & Yiğit, T. (2019). A novel hybrid PSO–GWO algorithm for optimization problems. *Engineering with Computers*, *35*, 1359–1373.

Talbi, E.G. (2002). A taxonomy of hybrid metaheuristics. *Journal of Heuristics*, *8*(5), 541-564.

Tarik, B.M., Zerhouni, F.Z., Stambouli, A.B., Tioursi, M., & M'harer, A. (2016). *Parameter optimization of photovoltaic solar cell and panel using genetic algorithms strategy*, in: Vasant, P., Weber, G., & Dieu, V. (Eds.), Handbook of Research on Modern Optimization Algorithms and Applications in Engineering and Economics. IGI Global, pp. 581-600.

Ting, T.O., Yang, X.-S., Cheng, S., & Huang, K. (2015). *Hybrid metaheuristic algorithms: past, present, and future*, in: Yang, X.-S. (Ed.), Recent Advances in Swarm Intelligence and Evolutionary Computation. Studies in Computational Intelligence. Springer, Cham, *585*, 71-83.

Wang, D., Zhou, Y., Jiang, S., & Liu, X. (2018). *A simplex method-based salp swarm algorithm for numerical and engineering optimization*, in: Shi, Z., Mercier-Laurent, E. & Li, J. (Eds.), Intelligent Information Processing IX. IIP 2018. IFIP Advances in Information and Communication Technology. Springer, Cham, *538*, pp.150-159.

Wang, S., Jia, H., Abualigah, L., Liu, Q., & Zheng, R. (2021). An improved hybrid aquila optimizer and harris hawks algorithm for solving industrial engineering optimization problems. *Processes*, *9*(9), article 1551, pages 28.

Wolpert D.H., & Macready, W.G. (1997). No free lunch theorems for optimization. *IEEE Transactions on Evolutionary Computation*, *1*(1), 67-82.

Wright, S. J. (2022). *Optimization. Encyclopedia Britannica.* https://www.britannica.com/science/optimization

Yang, X.-S., & Deb, S. (2009). *Cuckoo search via lévy flights*, in: 2009 World Congress on Nature & Biologically Inspired Computing (NaBIC). Coimbatore, India, pp. 210-214.

Yang, X.-S. (2010a). *Engineering optimization,* in: Engineering Optimization: An Introduction with Metaheuristic Applications. Wiley, Hoboken, NJ, pp. 15-28

Yang, X.-S. (2010b). Firefly algorithm, stochastic test functions and design optimisation. *International Journal of Bio-Inspired Computation*, *2* (2), 78-84.

Yang, X., & Deb, S. (2015). *Cuckoo search for optimization and computational intelligence*, in: Khosrow-Pour, M. (Ed.), Encyclopedia of Information Science and Technology Third Edition. IGI Global, pp. 133-142.

Yılmaz, Ö., Altun, A.A., & Köklü, M. (2022a). A new hybrid algorithm based on MVO and SA for function optimization. *International Journal of Industrial Engineering Computations*, *13*(2), 237-254.

Yılmaz, Ö., Altun, A.A., & Köklü, M. (2022b). Optimizing the learning process of multi-layer perceptrons using a hybrid algorithm based on MVO and SA. *International Journal of Industrial Engineering Computations*, *13*(4), 617-640.

Zhang, Q., Chen, H., Heidari, A.A., Zhao, X., Xu, Y., Wang, P., Li, Y., & Li, C. (2019). Chaos-induced and mutation-driven schemes boosting salp chains-inspired optimizers. *IEEE Access*, *7*, 31243-31261.