Expert Systems With Applications 188 (2022) 116001

Contents lists available at ScienceDirect

Expert Systems With Applications

journal homepage: www.elsevier.com/locate/eswa

Joint Opposite Selection (JOS): A premiere joint of selective leading opposition and dynamic opposite enhanced Harris' hawks optimization for solving single-objective problems

Florentina Yuni Arini [2], Sirapat Chiewchanwattana [*,1], Chitsutha Soomlek [3], Khamron Sunat [4]

Department of Computer Science, Faculty of Science, Khon Kaen University, and Advanced Smart Computing Lab, AIRC, Khon Kaen University, Thailand

ARTICLE INFO

Keywords:
Harris' Hawks Optimization (HHO)
Joint Opposite Selection (JOS)
Selective Opposition (SO)
Dynamic Opposite (DO)
Selection Leading Opposition (SLO)
Nature-Inspired Optimization Algorithm

ABSTRACT

In this paper, we proposed Joint Opposite Selection (JOS) operator that is a joint of two opposition learning techniques: the Selective Leading Opposition (SLO) and the Dynamic Opposite (DO). SLO uses a linearly decreasing threshold value to determine the close distance dimension of the search agents. DO provides the search agents chances to expand their abilities in the search space. We applied JOS to the Harris Hawks Optimization (HHO), the performance is increased because JOS balances the capability of exploration phase by using SLO and exploitation phase by using DO. The new algorithm, named Harris' Hawks Optimization-Joint Opposite Selection (HHO-JOS), is also proposed in this research as an enhanced version of HHO to solve single-objective problems. When the hawks deploy JOS, SLO assists the hawks to succeed in exploitation phase by changing their close distance dimension and DO tries to diverse the search space range of the hawks in the exploration phase using a Random Jump Strategy (RJS). The sufficient Jumping rate (Jr) of DO in HHO-JOS is 0.25, according to our experimental results. The proposed algorithm was included in a competition conducted on 30 benchmark functions of CEC 2014 and 29 benchmark functions of CEC 2017. Both benchmarks contain collections of single-objective problems for real parameter numerical optimization. The problems were employed to evaluate and compare the proposed HHO-JOS to the original HHO, three variations of OBLs embedded in the original HHO, and 31 nature-inspired algorithms by using a scoring metric. The results of the competition showed that the premiere JOS on HHO consistently achieves robustness performance on CEC 2014 and CEC 2017. Comprehensive statistical analysis also demonstrated that HHO-JOS can compete with many leading optimization algorithms. Therefore, we can conclude that the proposed joint opposite selection is well-matched to HHO and succeeds in elevating HHO-JOS.

1. Introduction

Based on technical reports by Awad et al. (2016) and Liang et al. (2013), single-objective function of the real parameter, numerical, and optimization is a primary complex task of the optimization problem. Many researchers used single-objective function to realize a novel optimization algorithm's strength. Scientific reviews confirm that a metaheuristic optimization algorithm (Boussaïd et al., 2013; Halim et al., 2020; Hussain et al., 2019) is a promising and sophisticated optimization approach and is purposed to minimize or maximize the objective of the problems. The minimization problem is commonly found in cost and energy for wireless sensor networks (WSNs) localization (Sharma and Gupta, 2020), tracking errors for portfolio management (Derigs & Nickel, 2003), the mass of truss (Pholdee and Bureerat, 2014), and test case minimization for software testing (Ahmed, 2016). The maximization attains to various works, such as increasing the lifetime route of a wireless sensor network (Wang et al., 2020), finding high potential investment in the market (Tsai & Liu, 2019), boosting the

* Corresponding author.
E-mail addresses: floyuna@yahoo.com (F.Y. Arini), sunkra@kku.ac.th (S. Chiewchanwattana), chitsutha@kku.ac.th (C. Soomlek), khamron_sunat@yahoo.com (K. Sunat).
[1] ORCID: 0000-0003-4473-2206.
[2] ORCID: 0000-0001-5850-8585.
[3] ORCID: 0000-0002-9063-0705.
[4] ORCID: 0000-0002-2042-3284.

https://doi.org/10.1016/j.eswa.2021.116001
Received 27 February 2021; Received in revised form 19 July 2021; Accepted 28 September 2021
Available online 2 October 2021
0957-4174/© 2021 Elsevier Ltd. All rights reserved.

seasonal allocation of hectares (Chetty & Adewumi, 2014), enhancing the manufacturing profit (Sathish, 2019), and increasing the reliability predictions of the identification of transcription factor binding sites (TFBSs) (Yang et al., 2011).

The minimization or maximization of the optimization algorithm influences the decisions of expert systems and their accomplishment. Nadimi-Shahraki et al. (2021) proposed an Improved Grey Wolf Optimizer (IGWO) by utilizing a dimensional learning-based hunting search strategy (DLH) to build a new way of communication among the wolf for sharing the information. The performance of IGWO was evaluated on the 29 benchmark functions of CEC 2018 and solved engineering problems. Wu et al. (2020) offered an adaptive logarithmic spiral-Levy Firefly Algorithm (AD-IFA) to strengthen the local exploitation and accelerate the convergence of the former LF-FA evaluated on 29 well-known benchmark functions. Elaziz et al. (2017) used an opposition-based learning technique to improved SCA. The improvement of SCA is to enhance the exploration ability in the search space by generating more accurate candidate solutions. Chen et al. (2017) employed a robust ant colony optimization (RACO) which applied the self-adaptive approach by considering domain adjustment, domain division, pheromone increment, and ant size of ACO for continuous functions.

The Harris' Hawks Optimization (HHO), which is a fairly span-new aviate optimization algorithm in Swarm Intelligence (SI), adopts the cooperative hunting behavior of Harris' Hawks proposed by Heidari et al. (2019). The major characteristic of HHO is the ability to represent the harmonization of its exploration and exploitation in the progressive searching selection with the randomized decrement escaping energy and in the rapid attack influenced by randomizing jump strength and short-dive using Lévy flight. It is noticed that the original HHO is evaluated and competes in real parameter optimization Congress Evolutionary Computation (CEC) 2005 (Suganthan et al., 2005). However, there is a lack of scientific evidence on whether the original HHO itself can work with more serious complex problems than the CEC 2005 benchmark functions, such as CEC 2014 and CEC 2017. Both benchmarks contain not only unimodal and multimodal, which show the promising capability of exploration and exploitation, but also hybrid functions and composite functions. Both benchmarks confirm the performance of the algorithm in terms of the ability to escape from the local optima.

Recently, to solve more complex optimization problems, the HHO has already been modified and developed. Many researchers claimed that blending HHO into the conventional algorithm produces a superior solution in real-world problems and complex problems. Rodríguez-Esparza et al. (2020) applied HHO to image segmentation and achieved more efficient and reliable performance in quality, consistency, and accuracy. Liu et al. (2020) proposed CCNMHHO as an improved version of HHO, which combines a simplified crisscross optimizer and the Nelder-Mead simplex algorithm. CCNMHHO is an efficient and reliable method to solve unknown parameters for photovoltaic and solar cell models. Houssein et al. (2020) claimed that the hybridizes Harris' Hawks Optimization-Support Vector Machines (HHO-SVM) and Harris' Hawks Optimization-k-Nearest Neighbors (HHO-kNN) achieved high classification accuracies for drug design and discovery prediction in chemoinformatics. Houssein et al. (2020) successfully employed HHO to determine the sink node location in a large-scale wireless sensor network (LSWSN) to measure and sense biosphere conditions such as temperature, wind, pressure, humidity, and pollution stages.

Although the development of HHO variants show superiority and efficiency in various applications mentioned above, there are still spaces for HHO to alleviate the local optima's stagnate issue and the premature convergence, such as using the opposition learning technique. The tendency of oppositional-based learning strategy's success influences the search strategies. Scientific reviews on the OBL's ability to enhance algorithms and employ them in applications are undoubted (Mahdavi et al., 2018; Rojas-Morales et al., 2017; Xu et al., 2014). Scientific researches show that the OBL-based techniques could be applied to enhance the performance of several existing evolutionary algorithms. Tubishat et al. (2020) utilized OBL and a local search algorithm to improve the Salp Swarm Algorithm (ISSA) for feature selection purposes. Gupta and Deep (2019) proposed a modification of SCA (m-SCA) by generating the opposite number according to the perturbation rate to escape from trapping at the local optima, then applied self-adaptive SCA to exploit the promising search area. Wu et al. (2017) applied Elite Opposition (EO) on Water Wave Optimization (WWO) for solving engineering problems and function optimization, then used Local Neighborhood Search (LNS) to enhance the local search of WWO. Li et al. (2016) expanded Cuckoo Search's search area (CS) by using elite opposition and chaotic disturbance. Yang and Huang (2012) embedded the opposition and dynamic Cauchy mutation into an Artificial Bee Colony to solve function optimizations.

In this paper, we simulated a group of Harris' hawks' behavior when they are hunting their prey, i.e., a jackrabbit. Our premier brand is a joint of OBL, named Joint Opposite Selection (JOS), which improves the artificial Harris' hawks to have promising search strategies. The search strategies (Olorunda & Engelbrecht, 2008; Xu & Zhang (2014); Morales-Castañeda et al., 2020) are defined as the exploration and exploitation phases. The joint search strategies used in the proposed JOS are Selective Leading Opposition (SLO) and Dynamic Opposition (DO). JOS assists the Harris' hawks in the leading group (SLO) which have close positions to the solution or the jackrabbit in this case. This leading group can accelerate their exploitation ability by moving to the opposite positions; if the hunting has not been success for a while, all the hawks have a chance to diverse asymmetrically and dynamically using DO (Xu et al., 2020). JOS on HHO succeeds in elevating the former HHO by balancing the search space, i.e., exploration and exploitation phases. The HHO's search action is scheduled by a decreasing escaping energy. The escaping energy indicates which search strategies are being used by the hawks and it is restrained by E. The E in HHO is set as a threshold value of SLO. If the hawks decide to maintain the exploration phase, the parameter Jr is employed as the jumping rate of DO. In other words, JOS is deployed by the threshold value used in SLO and the jumping rate is used in DO. In addition, the threshold is adjusted as a linearly decreasing parameter over time and it is used as the threshold value for SLO and the jumping rate for DO. More details about the search strategies will be explained in the later sections.

The performance of JOS embedded in HHO, also called HHO-JOS, is determined on a collection of 30 benchmark functions of CEC 2014 and 29 benchmark functions of CEC 2017. The main contributions of this research are highlighted as follows:

- A new opposition learning technique, Selective Leading Opposition, is invented.
- JOS is proposed and embedded in HHO to increase the hunting's performance of the HHO algorithm.
- The adjustment of linearly decreasing threshold value for SLO and jumping rate for DO succeed in elevating the former HHO.
- The reasonable probability jumping rate of DO in HHO-JOS is determined.
- The behavior of HHO with SLO, HHO with DO, and HHO with JOS are analyzed.
- The effectiveness of HHO-JOS performance is evaluated by using a scoring metric and statistical analysis, i.e., Wilcoxon sign rank test, Kruskal-Wallis test, and Friedman mean rank.
- Convergence graphs presenting nine algorithms on benchmark functions of CEC 2017 exhibit elite solution of HHO-JOS comparing to the original version of HHO.

The rest of the paper is organized as follows: the related work of HHO and opposition-based techniques and their inheritance is briefly presented in Section 2. In Section 3, JOS and HHO-JOS are proposed. Section 4 discusses experiment setup and experimental result analysis. Finally, the conclusions and future work are pointed out in the last section.

2. Related work

In classification, metaheuristic optimization algorithms can be categorized into four categories (Mirjalili & Lewis, 2016): evolutionary algorithm, physical-based algorithm, human-based algorithm, and swarm-based algorithm. Surprisingly, the quantity of recently proposed novel algorithms in the group of swarm-based algorithms, also known as Swarm Intelligent (SI) (Yang et al., 2016), is increased. As shown in Table 1, 27 novel SI optimization algorithms were proposed in the last five years. Those algorithms mimic the living organisms' intelligent behavior in the nature (Yang et al., 2018), which are influenced by their interactions in the community and environment. Each living organism has its patterns of surviving. These intelligences exhibit their behaviors of hunting, foraging, breeding, and mating. Scientists, then, recognized the patterns as Collective Intelligence (CI) and formulated them into mathematical formulas for computational decision purposes (Blomkvist et al., 2012; Henson & Hayward, 2010; Tron & Margaliot, 2004; Tsibulsky & Norman, 2007).

Table 1
Novel Swarm Intelligent Optimization Algorithms.

Animal Inspiration	Algorithms	Behavior	Year
Mayfly	Mayfly Algorithm (MA) (Zervoudakis & Tsafarakis, 2020)	Flight behavior and the mating process	2020
Sparrow	Sparrow Search Algorithm (SSA) (Xue & Shen, 2020)	Foraging and anti-predation	2020
Rat	Rat Swarm Optimizer (RSO) (Dhiman et al., 2020)	Chasing and attacking	2020
Hawks	Harris' Hawks Optimization (HHO) (Heidari et al., 2019)	Cooperative hunting	2019
Penguins	Emperor Penguins Colony (EPC) (Harifi et al., 2019)	Emperor penguins	2019
Butterfly	Butterfly Optimization Algorithm (BOA) (Arora & Singh, 2019)	Food search and mating	2019
Barnacles	Barnacles mating optimizer (BMO) (Sulaiman et al., 2019)	Mating	2018
Coyote	Coyote Optimization Algorithm (COA) (Pierezan & Dos Santos Coelho, 2018)	Canis latrans species	2018
Owl	Owl Search Algorithm (OSA) (Jain et al., 2018)	Hunting	2018
Grasshopper	Grasshopper Optimization Algorithm (GOA) (Saremi et al., 2017)	Behavior	2017
Mouth brooding fish	Mouth Brooding Fish (MBF) (Jahani & Chizari, 2018)	Movement, dispersion, and protection	2017
Spotted hyena	Spotted Hyena Optimizer (SHO) (Dhiman & Kumar, 2017)	Hunting	2017
Lion	Lion Optimization Algorithm (LOA) (Yazdani & Jolai, 2016)	Cooperation characteristics	2016
Dolphin	Dolphin Swarm Optimization Algorithm (DSOA) (Yong et al., 2016)	Hunting	2016
Dragonfly	Dragonfly Algorithm (DA) (Mirjalili, 2016b)	Navigation, foods search, enemies avoidance	2016
Moth-flame	Moth-flame Optimization (MFO) (Mirjalili, 2015)	Navigation	2015
Elephant	Elephant Herding Optimization (EHO) (Wang et al., 2016)	Herding	2015
Grey Wolf	Grey Wolf Optimizer (GWO) (Mirjalili et al., 2014)	Hunting	2014
Swallow	Swallow Swarm Optimization (SSO) (Neshat et al., 2013)	Movement	2013
Krill	Krill Herd (KH) (Gandomi & Alavi, 2012)	Herding	2012
Cuckoo	Cuckoo Optimization Algorithm (COA) (Rajabioun, 2011)	Breeding	2011
Bat	BAT (Yang, 2010)	Echolocation	2010
Cuckoo	Cuckoo Search (CS) (Yang & Deb, 2009)	Brood parasitic CS of some birds	2009
Bees	Bee Collecting Pollen Algorithm (BCPA) (Lu & Zhou, 2008)	Honeybees collecting pollen	2008
Monkey	Monkey Search (MS) (Mucherino & Seref, 2007)	Food Search	2007
Ant	Ant Colony Optimization (ACO) (Dorigo et al., 2006)	Food Search	2006
Bird Flock	Particle Swarm Optimization (PSO) (Kiranyaz, 2014)	Movement	1995

Based on No Free Lunch (NFL) theorem (Wolpert and Macready, 1997), there is no constrain or rule to define an idea based on the mimicking of nature-inspired behavior. The ideas can be behavior that presents the original mimic based on the nature-inspired as shown in Table 1. Nevertheless, the ideas can also be some kinds of strategies to elevate an algorithm. For example, Li et al. (2020) offered an idea of a learning model that utilizes communication and potential knowledge and among the individuals and the population. Li et al. (2019) proposed dynamic Cuckoo Search using Taguchi opposition-based search to solve multi-dimension problems. Li and Wang (2021) improved the elephant herding optimization by adapting the biogeography based learning named BLEHO to ensure the continuity of the population. Li et al. (2020) presented an improvement of elephant herding optimization (IMEHO) that employs the strategy of learning, velocity, and elitism to meet the best individual for preserving the lifecycle. Wang et al. (2016) elevated the original Krill herd by utilizing the opposition based learning, Cauchy mutation and position clamping. Li et al. (2020) used the genetic operator and Q-Learning step size to enhance the performance of cuckoo search, called DMQL-CS.

This section is divided into two parts. The first part explains the Harris' Hawks Optimization (Heidari et al., 2019). The second part describes the Opposition Based Learning technique (Tizhoosh, 2005) and its extended techniques relative to the proposed Joint Opposite Selection (JOS), namely Selective Opposition (Dhargupta et al., 2020) and Dynamic Opposite (Xu et al., 2020).

2.1. Harris' hawks optimization (HHO)

HHO (Heidari et al., 2019) adopts the nature hunting "surprise pounce" strategy of Harris' hawks in searching, tagging, enclosing, and finally attacking their prey, a jackrabbit (Bednarz, 1988). A hawk in Fig. 1 illustrates the role of the hunting strategy of each hawk. The hunting strategy of HHO is divided into two phases, i.e., exploration and exploitation. HHO also takes the escaping energy $|E_S|$ of a jackrabbit into consideration, as shown in Fig. 2. The hawks' exploration phase is an extensively promising global area for the hawks as a probability search space randomly explores for a jackrabbit. The hawks' exploitation phase is a hopeful local search guided by the hawks' exploration phase. The escaping energy $|E_S|$ is linearly decreased from two to zero as depicted by a yellow arrow pointing to the jackrabbit in Fig. 1. The magnitude of escaping energy $|E_S|$ is linearly decreased from two to zero as depicted by a yellow arrow pointing to the jackrabbit in Fig. 1. The decrement of $|E_S|$ occurs as the iteration is decreasing by t/T which can be defined as $E = 2^*(1-(t/T))$, where t is the initial iteration and T is the maximum iteration.

Generally, in the exploration phase, the hawks perform two strategies, either perch based on the other hawks, $q < 0.5$, or perch based on their random locations, $q \geq 0.5$. In the exploitation phase, there are two considerations: the strength of the prey and the strategy of the hawks. The strength of the prey is represented by the decrement of escaping energy from one to zero and is divided into two parts. When $0.5 \leqslant |E_S| < 1$, the jackrabbit still has enough energy to escape. On the other hand, when $0 \leqslant |E_S| < 0.5$, the jackrabbit is running out of energy. Meanwhile, the strategy of the Harris' hawks is influenced by a random number (r), where $r = [0,1]$. If r is less than 0.5, the hawks will perform rapid dive on soft besiege and hard besiege. When r is equal to or greater than 0.5, the hawks will perform only soft besiege and hard besiege without the rapid dive. Therefore, the hawks will strike a rapid dive only when the jackrabbit has limited energy to escape.

The activities of the Harris' hawks on monitoring and perching the

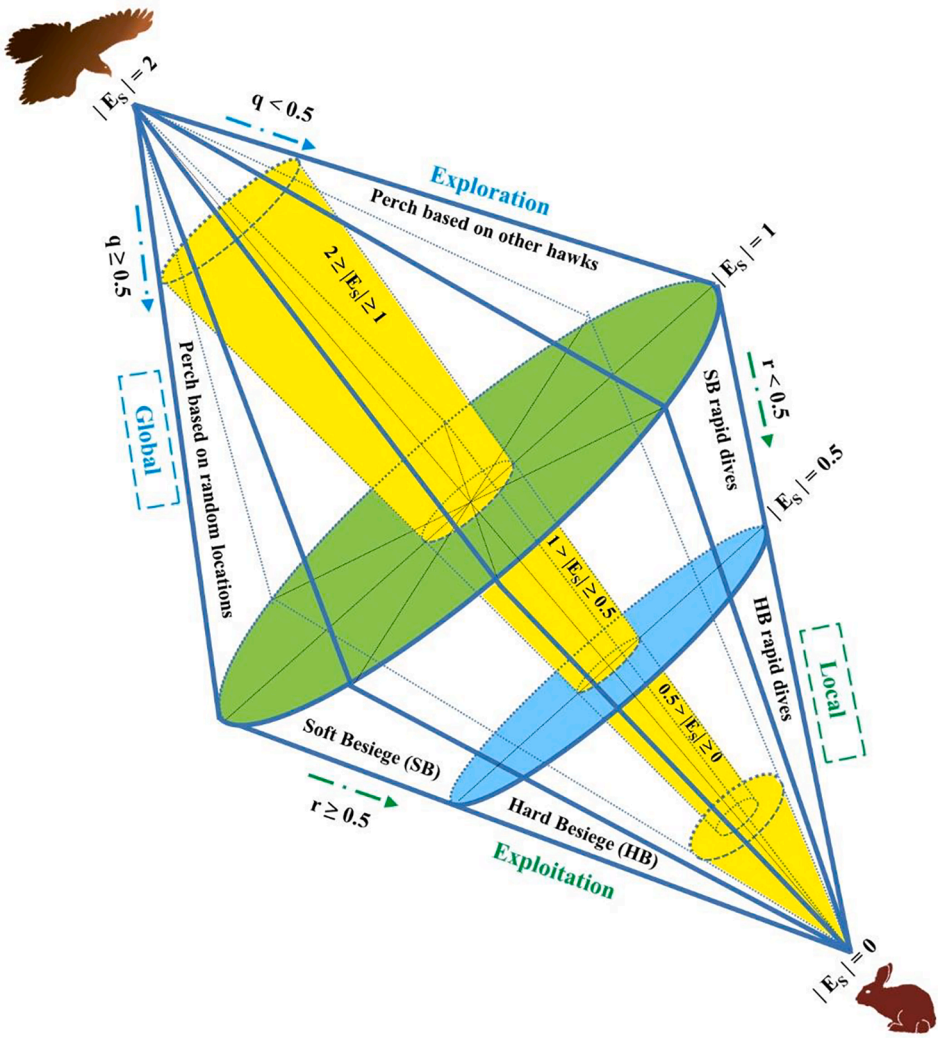


Fig. 1. Harris' Hawks Optimization (HHO) Exploration and Exploitation Phase.

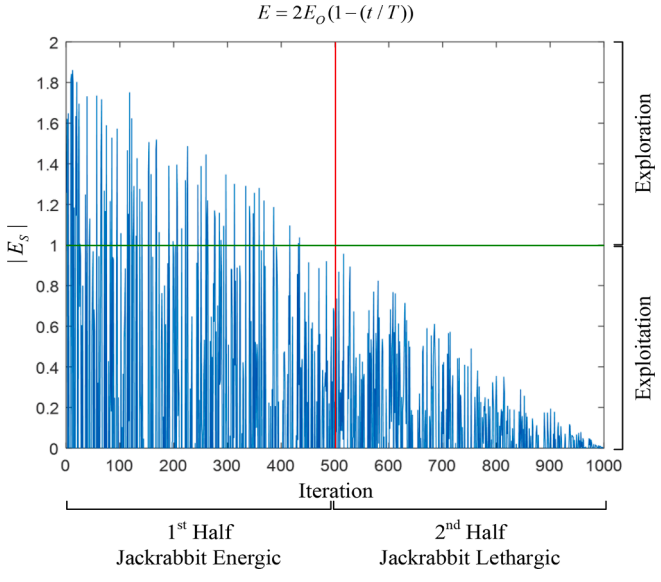


Fig. 2. The evolution of absolute escaping energy ($|E_S|$) of a jackrabbit ran for 1000 iterations globally.

jackrabbit are formulated in Eq. (1) regarding the number of hawks in a partition group. From Eq. (1), when $q < 0.5$, the partition is based on the other hawks' positions. It means the current hawks move toward the selected hawks when the positions are close enough to the jackrabbit and it is time to strike. If $q \geqslant 0.5$, the partition group of the average hawks randomly monitors the jackrabbit from the height of the trees within the range inside their hunting ground, which is represented by *LB* and *UB*.

$$\boldsymbol{X}_{h+1} = \begin{cases} \boldsymbol{X}_{hs} - r_1 * |\boldsymbol{X}_{hs} - 2 * r_2 * \boldsymbol{X}_{hc}| & q \geqslant 0.5 \\ \left(\boldsymbol{X}_{jrabbit} - \boldsymbol{X}_{hm}\right) - r_3(LB + r_4(UB - LB)) & q < 0.5 \end{cases} \quad (1)$$

$\boldsymbol{X}_{h+1}$ is the new position vector of the hawks in the next generation, $\boldsymbol{X}_{hs}$ is randomly selected hawks, $\boldsymbol{X}_{hc}$ is the current position vector of the hawks, and $\boldsymbol{X}_{jrabbit}$ is the position of the jackrabbit where r_1, r_2, r_3, r_4, and q are uniform random numbers between [0,1]. The boundary variables are *LB* (Lower Bound) and *UB* (Upper Bound). $\boldsymbol{X}_{hm}$ is the mean of the current positions of the hawks $\boldsymbol{X}_{hc}$ in each d dimension on D dimension as indicated in Eq. (2):

$$\boldsymbol{X}_{hm,d} = \frac{1}{N} \sum_{i=1}^{N} \boldsymbol{X}_{hc,d}, \ \forall d \ \text{ such that } 1 \leqslant d \leqslant D \quad (2)$$

The change of hawks' hunting search space from the exploration phase to the exploitation phase is influenced by the escaping energy of the jackrabbit as stated in Eq. (3):

$$E_S = 2^{*}E_0{}^{*}\left(1-\frac{t}{T}\right) \tag{3}$$

where E_S represents the escaping energy of the jackrabbit, and T is the maximum number of iterations and E_0 represents initial energy which is influenced by $2^{*}rand-1$.

Fig. 2 illustrates the changes of the magnitude of jackrabbit escaping energy $|E_S|$ at the phases of exploration and exploitation as shown on the $y-$ axis. The changes control the linear decrement of $2^{*}(1-(t/T))$. The $x-$axis shows the changes of the $|E_S|$ on each iteration starting from zero until it reaches the maximum value (T). $|E_S|$ is influenced by the initial energy E_0, which is varied between $[-1, 1]$ in every iteration. We assume that in the first half of the iteration, the jackrabbit is still energized, and in the second half, the jackrabbit is lethargic. As shown in Fig. 1, the hawks' exploitation phase performs four tactics: soft besiege, soft besiege with rapid dive, hard besiege, and hard besiege with rapid dive. The tactics are influenced by the instinct of the jackrabbit to avoid the predators, the Harris' Hawks. We assume that r is the opportunity of a jackrabbit for escaping from the hawks.

The first potential tactic of a group of hawks is soft besiege $(r \geqslant 0.5)$. In this case, the jackrabbit still has enough energy to escape $(0.5 \leqslant E_S < 1)$. The soft besiege is formulated in Eq. (4). The escaping energy E_S is influenced by the movement of the current positions of the hawks $\boldsymbol{X}_{hc}$ toward the jackrabbit's position with a certain Jumping rate J_S approaching the delta position of current hawks. $\Delta\boldsymbol{X}_{hc}$ represents the movement from the current positions of the hawks toward the jackrabbit. The delta position describes in Eq. (5) and J_S is a random jump strength of the jackrabbit to simulate the jackrabbit's escaping movement. The J_S modeled in $J_S = 2(1-r_5)$, where r_5 is a random number between [0,1].

$$\boldsymbol{X}_{h+1} = \Delta\boldsymbol{X}_{hc} - E_S\left|J_S\boldsymbol{X}_{jrabbit} - \boldsymbol{X}_{hc}\right| \tag{4}$$

$$\Delta\boldsymbol{X}_{hc} = \boldsymbol{X}_{jrabbit} - \boldsymbol{X}_{hc} \tag{5}$$

The second potential tactic is the hawks decide to perform hard besiege $(r_5 < 0.5)$ and the jackrabbit is lacking of energy to escape $(0 \leqslant |E_S| < 0.5)$ as modeled in Eq. (6). In this situation, the hawks attack hard. Eq. (6) calculates the new positions of the hawks by taking the escaping energy of the jackrabbit into consideration and the delta movement of the current positions vector of the hawks toward the jackrabbit's position.

$$\boldsymbol{X}_{h+1} = \boldsymbol{X}_{jrabbit} - E_S|\Delta\boldsymbol{X}_{hc}| \tag{6}$$

In the third and fourth potential tactics, the hawks perform rapid dive together with soft besiege and hard besiege to assassinate the jackrabbit before the hawks perform a surprise pounce strategy. For soft besiege with rapid dive, we still use the original soft besiege modeled in Eq. (7). The difference is the escaping energy. The escaping energy is influenced by the movement of the current hawks toward the jackrabbit with a jumping rate approaching to the jackrabbit's position. If they fail from using the first strategy $(\boldsymbol{X}_{h1s})$ on the soft besiege, they will perform the second strategy $(\boldsymbol{X}_{h2s})$ in Eq. (10) by using the rapid dive Lévy flight as shown in Eq. (8,9).

$$\boldsymbol{X}_{h1s} = \boldsymbol{X}_{jrabbit} - E_S\left|J_S\boldsymbol{X}_{jrabbit} - \boldsymbol{X}_{hc}\right| \tag{7}$$

$$\boldsymbol{X}_{h2s} = \boldsymbol{X}_{h1s} + S \times LF(d) \tag{8}$$

where d is the dimensions.

$$LF(d) = 0.01 \times \frac{u \times \sigma}{|v|^{\frac{1}{\beta}}}, \sigma = \left(\frac{\Gamma(1+\beta) \times sin\left(\frac{\pi\beta}{2}\right)}{\Gamma\left(\frac{1+\beta}{2}\right) \times \beta \times 2^{\left(\frac{\beta-1}{2}\right)}}\right)^{\frac{1}{\beta}} \tag{9}$$

$$\boldsymbol{X}_{h+1} = \begin{cases} \boldsymbol{X}_{h1s} & if\ f(\boldsymbol{X}_{h1s}) < f(\boldsymbol{X}_{hc}) \\ \boldsymbol{X}_{h2s} & if\ f(\boldsymbol{X}_{h2s}) < f(\boldsymbol{X}_{hc}) \end{cases} \tag{10}$$

Paul Lévy presented the details of the Lévy flight formula (Chawla & Duhan, 2018; Jamil & Zepernick, 2013; Kamaruzaman et al., 2013) modeled in Eq. (9) by using a simple power-law formula, where u and v are random numbers between [0,1]. β is a constant value equal to 1.5. Eq. (10) updates the new position of the hawks.

$$\boldsymbol{X}_{h1s} = \boldsymbol{X}_{jrabbit} - E_s\left|J_s\boldsymbol{X}_{jrabbit} - \boldsymbol{X}_{hm}\right| \tag{11}$$

The first attack strategy on hard besiege with rapid dive is formulated in Eq. (11). The only difference from the soft besiege with rapid dive in Eq. (10) is the escaping energy. This time, the escaping energy is influenced by the average hawks' positions $(\boldsymbol{X}_m)$ that approach the jackrabbit with a jumping rate. If the first strategy on hard besiege fails, it will employ the rapid dive using Lévy flight. To execute the second strategy of hard besiege with rapid dive, Eq. (8, 9, 10) are run in sequence.

2.2. Opposition based learning (OBL)

OBL was originally proposed by Tizhoosh (2005). OBL is an opposite learning technique for optimization algorithms, especially in metaheuristic optimization algorithms, to improve their quality performance at the initialization stage and the generation stage by using a jumping rate (Rahnamayan et al., 2007). Scientists (Mahdavi et al., 2018; Rojas-Morales et al., 2017; Xu et al., 2014) summarized the effectiveness of this technique. The use of the opposite solution for a given problem provides opportunities to find candidate solutions randomly closer to the global solutions (Wang et al., 2011). Rahnamayan (2007) defines the opposite number and opposite point as follows:

Definition of Opposite Number: Let x be a real number in an interval $[LB, UB]; x \in [LB, UB]$. LB is the lower bound and UB is the upper bound. The opposite number of x is denoted by $\overline{x}$ in Eq. (12).

$$\overline{x} = LB + UB - x \tag{12}$$

Definition of Opposite Point: Let $\boldsymbol{X} = (x_1, x_2, ..., x_d)$. Each element in $\boldsymbol{X}$ is a point in a $d$dimensional space, $x_j \in \Re$ and $x_j \in [\mathrm{LB_j}, \mathrm{UB_j}]$, where $j = 1, 2, ..., d$. The opposite point of $\boldsymbol{X}$ is denoted by $\overline{\boldsymbol{X}} = (\overline{x}_1, \overline{x}_2, ..., \overline{x}_d)$ in Eq. (13).

$$\overline{\boldsymbol{X}} = LB + UB - \boldsymbol{X} \tag{13}$$

The opposite point is compared to the $\boldsymbol{X}$ position, then, it is sorted by its fitness value. If $\boldsymbol{X}$ fitness value is better than $\overline{\boldsymbol{X}}$ fitness value, then the position's population remains the same. However, if $\overline{\boldsymbol{X}}$ fitness value is better than $\boldsymbol{X}$ fitness value, the position population is updated based on better $\overline{\boldsymbol{X}}$ fitness value. Recently, OBL is extended its idea to achieve higher performance. The Selective Opposition (SO) and Dynamic Opposite (DO) are the improved versions of OBL adopted by the proposed Joint Opposite Selection (JOS).

2.2.1. Selective Opposition (SO)

SO is an extended idea of OBL introduced by Dhargupta et al. (2020). The idea of SO (Pseudocode 1) is to change the faraway dimension of the search agents closer to the solution using the opposition strategy which is influenced by a linearly decreasing threshold.

Pseudocode 1: Selective Opposition (SO)

1: **Input**: initial generation t, maximum generation T, population size N, dimension$d$
2: **Output**: $\overline{X}_{d_t}$: new position based on selective opposition
3: threshold $= 2 - \left[t^{*}\left(\frac{2}{T}\right)\right]$
4: **for** $i = 1 : N$
5: **if** $\boldsymbol{X}_i$ is not equal to $\boldsymbol{X}_{ibest}$
6: **for** $j = 1 : d$

(continued on next page)

(*continued*)

```
    ddj = |x_ibest,j − x_i,j| {ddj = difference distance for each dimension}
    if ddj > threshold
      identify the close distance dimensions (df)
      count the number of close distance dimensions (df)
    else
      identify the faraway distance dimensions (dc)
      count the number of far distance dimensions (dc)
    end
  end
  sum all ddj
  src = 1 − 6*Σ_{j=1}(ddj)^2 / (ddj*(ddj^2 − 1)), {src = Spearman's Rank Correlation Coefficient}
  if src <= 0 and df > dc
    perform X̄_df = LB_df + UB_df − X_df
  end
 end
end
```

The first step to perform SO is defining a threshold. The threshold will decrease until it reaches the maximum generation. For all search agents, SO examines the distance for each of them from the current search agent dimension comparing to the best position of the search agent, as denoted in Eq. (14).

$$dd_j = |x_{ibest,j} - x_{i,j}| \tag{14}$$

where dd_j is the difference distance of each dimension. When the difference distance (dd_j) is greater than the threshold, identify and calculate for the faraway and close search agents. Then, enumerate all the difference distances of the agents and the solution.

$$src = 1 - \frac{6*\sum_{j=1}\left(dd_j\right)^2}{dd_j*\left(dd_j^2 - 1\right)} \tag{15}$$

The enumeration influences the Spearman's Rank Correlation Coefficient (src) value as indicated in Eq. (15). The src measures the associativity of the current hawks and the jackrabbit. If $src$is less than zero and the number of the faraway distance dimensions (d_f) is greater than the number of close distance dimensions (d_c), the search agent will be moved by Eq. (16).

$$\overline{X}_{d_f} = LB_{d_f} + UB_{d_f} - X_{d_f} \tag{16}$$

2.2.2. Dynamic opposite (DO)

Xu et al. (2020) merged the idea of Quasi Opposition (Rahnamayan et al., 2007) and Quasi-Reflection (Ergezer et al., 2009) to create Dynamic Opposite (DO). The excellence of DO is the capability to dynamically expand the search space. Therefore, the expansion of the search space using the opposite strategy contributes to the probability of approaching the global solution. The movement of DO is illustrated in Pseudocode 2.

Pseudocode 2: Dynamic Opposite (DO)

```
Input: Jumping rate Jr, population size N, position X
Output: X̄_do: new position based on DO
if rand(0,1) < Jr
  for i = 1 : N
    X̄ = LB + UB − X (17)
    X_r = rand(X̄), rand ∈ [0,1] (18)
    X̄_do = X + rand(X_r − X) (19)
  end
end
```

The Jumping rate (Jr) defines the probability of performing DO, $\overline{X}$ is the opposite point, $X_r$is the random value for the opposite point, and $\overline{X}_{do}$ is the dynamic opposite. If the random value is less than the nominated Jr, the DO will perform for all populations. This condition is named Random Jump Strategy (RJS). The random jump tries to effectively restart the exploration ability and increase highly optimized performance (Hu et al., 1994; Selvam and Narayanan, 2019; Zhang et al., 2021). The initial position of DO is denoted in Eq (17), the current position of X makes a move to the opposite position $\overline{X}$ in the search space. The move of opposite position $\overline{X}$ is influenced by a certain random number between [0,1] which provides a new position X_r as shown in Eq. (18). Then, the DO will deploy Eq. (19) when the random number between [0,1] and the current position X approaching to X_r moves in a different direction from the current position.

3. HHO-JoS

Our new idea is the artificial Harris' hawks should have some strategies that are diverging from real behaviors and is illustrated in this section. Section 3.1 exhibits the proposed JOS and Section 3.2 describes the workflow of JOS embedded in HHO. Section 3.3 illustrates HHO-JOS in the search space.

3.1. The proposed JOS

Opposition Based Learning (OBL) is a learning technique that offers opportunities to find better solutions that imminent to the global optimal using the opposite move. In general, OBL is a mechanism to enhance optimization algorithms. In this paper, we proposed a new opposition-based learning technique named Joint Opposite Selection (JOS) to improve the performance and efficiency of the later HHO in the search space.

Fig. 3 exhibits the hierarchy development of JOS in OBL. JOS is a joint opposition-based learning technique that combine the benefits of Dynamic Opposite (DO) and Selective Leading Opposition (SLO). The DO is a merged opposition of quasi and quasi-reflection (Xu et al, 2020). SLO is expanded from the original idea of Selective Opposition (SO), which is embedded in the Grey Wolf Optimizer (GWO) (Dhargupta et al., 2020).

In former GWO, Fig. 4 shows the locations of the search agents based on the competence (fittest value) which is classified based on its social hierarchy. First, alpha (α) is the fittest solution for the decision-maker. Beta (β) and delta (δ), which are the second and third solutions respectively, have the role to inform prospective hunting location. The last one, omega (ω), plays an assistant role of those three wolves classifier. Because the task of omega (ω) is to assist those three wolves, its position can be either faraway (ω_1) or close (ω_2) to the fittest solution (α). In the SO strategy, the intention is to manage the exploration performance by changing the far away (ω_1) search agent by moving to the opposite position to find a better solution. This strategy is to produce a faster convergence rate. However, the swift convergence rate might

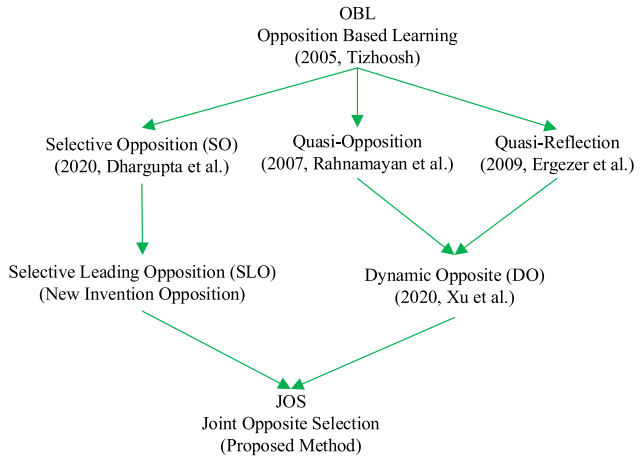


Fig. 3. A Hierarchical Development of JOS in OBL.

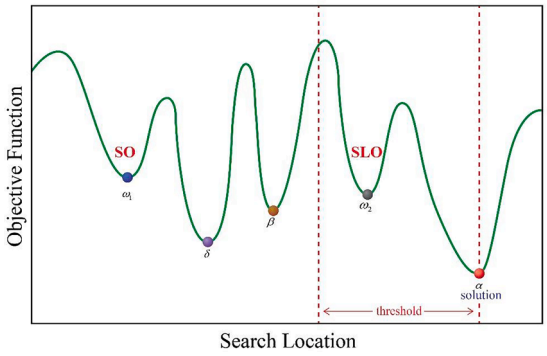


Fig. 4. The difference between the locations selected by SO (faraway) and selected by SLO (close) to the solution.

quickly lead to being trapped in the local optima, causing its performance to be unstable as exhibits in the experiment in Section 4. Then, we consider changing the strategy by selecting the leading search agent which has a close (ω_2) position to the fittest solution (α). The move of the close agent is influenced by the threshold that is linearly decreased over time. The idea is the closer position can effectively reach the global optima. We named our new invention Selective Leading Opposition (SLO). In our proposed, we used SLO to support the capability of JOS on HHO.

The opposite move of SLO is illustrated Fig. 5. Suppose that the leading point of a search agent (x,y,z) in Fig. 5 has the exact opposite point (x',y',z'); the distance between those two points to the solution are C_1 and C_3, successively. Further proof, suppose the last two dimensions of (x,y,z) are opposed to (x,y',z') with the distance C_2. Therefore, the distance comparison results in $C_2 < C_1 < C_3$ shows that C_2 has the nearest distance to the solution. The detailed explanation of the close distance dimension change of the SLO is shown in Fig. 6.

Based on Fig. 6, the coordinate x and y represent the first and second dimension, respectively. Together, those coordinates represent the coordinate position of hawks (x_h, y_h) and the coordinate position of a jackrabbit (x_r, y_r). From the examples in Fig. 6, the position of the jackrabbit stays on the same coordinate R(5,4), which means the first dimension $x_r = 5$ and the second dimension$y_r = 4$. Considering coordinate A as the first possible case, the coordinate of the hawk's position is (4,1), where the first dimension is $x_h = 4$ and the second dimension is

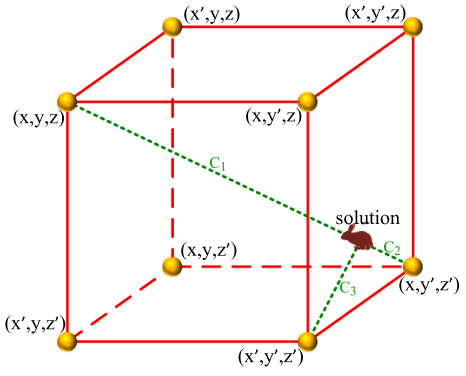


Fig. 5. The opposite movement of SLO.

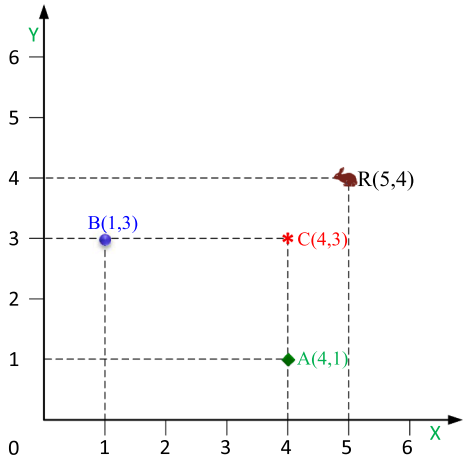


Fig. 6. The change of close dimension on SLO. Coordinate A, B, and C are the positions of hawks (x_h, y_h) and coordinate R is the position of a jackrabbit (x_r, y_r)in two-dimensions. For the close distance dimension, coordinate A is the x, coordinate B is the y, and coordinate C are the x and y.

$y_h = 1$. Then, we can calculate the difference distance between x_r and x_h based on coordinate A, which is 1. If we assumed that the threshold is 2, the difference distance between x_r and x_h is less than 2. That means the coordinate x of this hawk is closer to the jackrabbit than the other hawks. Then, this closer hawk should be the selected leader for hunting. At this condition, the strategy SLO will apply at the first dimension. For the second dimension, the difference is greater than 2; it means they do not need to apply SLO because the coordinate y_h is far away from y_r. For the second case, considering point B, the SLO is only applied to the second dimension. As for the last case, considering coordinate C, the SLO is applied to both dimensions. Therefore, from the examples in Fig. 6 above, we can conclude that the opposition strategy on the SLO will move the hawks closer to the rabbit in the opposite directions by considering their current positions and their eligible dimensions.

Pseudocode 3: Selective Leading Opposition (SLO)

Input: initial generation t, maximum generation T, population size N, dimension d
Output: $\overline{X}_{d_c}$: new position based on selective opposition
threshold $= 2 - \left[t^* \left(\frac{2}{T} \right) \right]$
for $i = 1 : N$
  if X_i is not equal to X_{ibest}
    for $j = 1 : d$
      $dd_j = |x_{ibest,j} - x_{i,j}|$ {dd_j = difference distance for each dimension}
      if $dd_j <$ threshold
        identify the close distance dimensions (d_c)
        count the number of close distance dimensions (d_c)
      else
        identify the faraway distance dimensions (d_f)
        count the number of far distance dimensions (d_f)
      end
    end
    sum all dd_j
    $src = 1 - \frac{6 * \sum_{j=1} (dd_j)^2}{dd_j * \left(dd_j^2 - 1 \right)}$, {$src$ = Spearman's Rank Correlation Coefficient}
    if $src <= 0$and $d_c > d_f$
      perform $\overline{X}_{d_c} = LB_{d_c} + UB_{d_c} - X_{d_c}$
    end
  end
end

The flow of SLO is described in Pseudocode 3. SLO sets a threshold value to classify the faraway and close distance dimensions by considering the difference distance for each dimension, if the current position is not equal to the solution position. The threshold is linearly decreased in the new generation. After evaluating the position of each agent, SLO calculates the correlation of the selected agent's position to the best solution's position using the Spearman's Rank Correlation Coefficient. Spearman's Rank Correlation Coefficient measures the association strength of the selected agent and the best solution. If the calculation result of the correlation coefficient is less than zero or negative, then the position of the eligible search agent is opposed. Therefore, the Spearman coefficient assists efficiently by selecting the eligible search agent and dimensions that perform the opposite move.

Based on the experimental result in Section 4, the SLO in HHO is better than SO. However, the improvement is not much because of its ability to converge fast. Therefore, the DO strategy tries to overcome this drawback by enlarging the opportunities of all search agents in the search space to find a better solution. The DO moves dynamically and asymmetrically in the search space. The opportunity is influenced by the jumping rate. The sufficient DO jumping rate of JOS is 0.25 (for further experimental evidence, see Section 4).

3.2. The proposed HHO-JOS

Fig. 7 shows how we embed JOS (SLO and DO) in HHO-JOS. The strategy of JOS in HHO initially starts with a random set domain of hawks, which produces DO initial population within the range boundary of lower bound and upper bound based on the dimension. While the number of function evaluations (*nFE*) is less than the maximum function evaluations (*maxFE*), it will perform the main HHO-JOS. The JOS strategy is deployed after the verification of the hawks' boundaries and evaluation of the hawks' fitness value. Each time the hawks evaluate its fitness, the *nFE* is updated. The best fitness value of hawks is the jackrabbit's position.

The linearly decreasing escaping energy of the jackrabbit controls the search action of HHO in the search space and it is restrained by E (Pseudocode 4 line 9). Then, the E of HHO is set as the threshold value of SLO. To perform the movement, the selected boundary for SLO is also defined. The details of the movement in SLO strategy is exhibited based on Pseudocode 3. Afterward, the hawks perform the "surprise pounce" strategy in the exploration and exploitation phases sequentially as shown in Pseudocode 4. To balance the search space, the DO is performed with the Random Jump Strategy (RJS). RJS uses a proper jumping rate of 0.25. Therefore, all hawks have 25 percent chances to expand the search space to escape from trapping at the local optima.

At the end of the optimizing process, HHO-JOS produces single best fitness value known as one best objective function. This term is a definite solution of single-objective optimization (Mirjalili, 2016a) for minimization problem shown in Eq. 20.

$$\text{minimization } f(x_1, x_2, x_3, \ldots, x_{d-1}, x_d) \quad \text{in range of } LB_j \leqslant x_j \leqslant UB_j,\ j = 1, 2, 3, \ldots, d \tag{20}$$

where d is the number of dimensions, *LB* is the lower bound of the *j-th* dimension and *UB* is the upper bound of the *j-th* dimension. To confirm the best fitness of HHO-JOS, the hawks in optimization keep updating their best fitness value until they meet the final criteria *maxFE* and continues until they complete 51 runs, which is the maximum number of runs in our experiments. We can conclude its average best fitness on 51 runs. According to the experiments

in Section 4, JOS on HHO shows the effectiveness of solving the single-objective problems. The JOS makes a major impact and improvement on the minimization of single-objective problems especially when it is integrated to the original HHO, resulting in HHO-JOS.

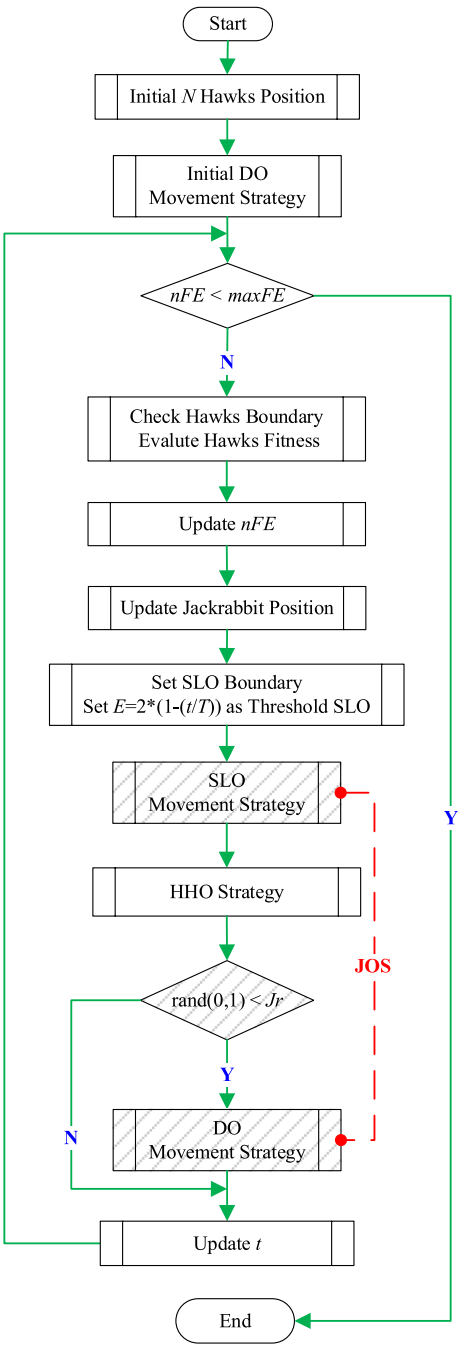


Fig. 7. How JOS is embedded in HHO

It is noticeable that the majority of real-world problems are multi-objective problems. A multi-objective problem is referring to a problem that has more than one objective function (Coello Coello, 2009). Due to the lack of the fittest solution, a multi-objective problem could be converted to single-objective problems (Deb and Deb, 2014; Neumann & Wegener, 2007). For example, to balance the performance of economic

and environmental, eco-design converts multi-objective to single-objective (Lim et al., 2013). The resulting experiment showed the eco-efficiency in the variety of systems, processes, and products. Tan et al. (2009) experienced converting multi-objective PSO to single-objective to find a competitive solution for solving the engineering problem.

Pseudocode 4: HHO-JOS

1: Initialize individual randomly in population of X_h in size of N
2: Initialize individual randomly in population of DO in size of N //**Pseudocode 2**
3: $nFE = N$
4: **while** $nFE < maxFE$ **do**
5: Update population of X_h
6: Calculate the fitness values of X_h
7: Best fitness value of $X_h$is the best location of jackrabbit
8: Set selective boundary for SLO
9: Set $E = 2^*(1-(t/T))$ as threshold for SLO //SLO Threshold
10: Perform SLO //**Pseudocode 3**
11: **for** $i = 1 : N$
12: Set Escaping Energy$E_S = E^*(2^*rand() - 1)$ Eq. (3)
13: **if** $|E_S| \geq 1$ **then** //Exploration Phase
14: update the position of X_h Eq. (1)
15: **elseif** $|E_S| < 1$**then** //Exploitation Phase
16: **if** ($r \geq 0.5$ and $|E_S| < 0.5$) **then** //Soft Besiege
17: update the position of X_h Eq. (4,5)
18: **elseif** ($r < 0.5$and $|E_S| < 0.5$) **then** //Hard Besiege
19: update the position of X_h Eq. (6)
20: **elseif** ($r \geq 0.5$and $|E_S| \geq 0.5$) **then** //Soft Besiege rapid dive
21: update the position of X_h Eq. (7,8,9)
22: **elseif** ($r < 0.5$and $|E_S| \geq 0.5$) **then** //Hard Besiege rapid dive
23: update the position of X_h Eq. (9,10,11)
24: **end**
25: **end**
26: **end**
27: **if** $rand(0,1) < Jr$
28: Perform DO //**Pseudocode 2**
29: **end**
30: **end**

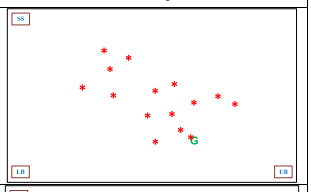
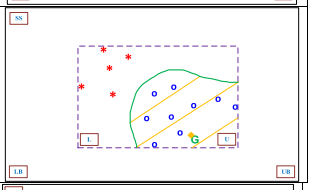
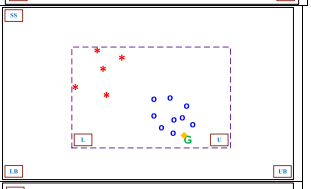
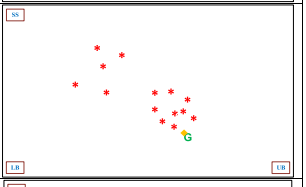
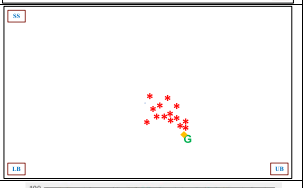
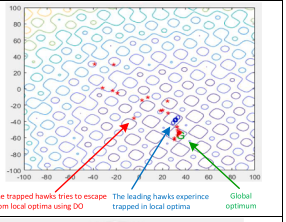
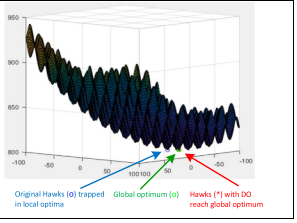
3.3. HHO-JOS in search space

In the search space, the movement strategy of the hawks with the JOS technique is illustrated in Table 2. Part no. I illustrates the diversity of the N hawks ($N = 15$) in the original boundary's search space $[LB, UB]$ LB is the original lower boundary and UB is the original upper boundary. The position of a hawk (red star ∗), near the global optimum G, represents the best position of the hawk. In part no. II, we can assume that the best position of the hawk is the jackrabbit location (brown diamond ◆). For SLO preparation, a boundary is set based on the current hawks' positions and a threshold value is determined. A dashed rectangle -- indicates the boundary of SLO, where L is the lower boundary and U is the upper boundary. The green curve line ⌒ defines the threshold. The hawks in blue circles o and the shading area ⁄ present the hawks with the difference distance dimensions less than the threshold value. It means that the hawks are closer to the jackrabbit. The red stars ∗ presents the hawks with difference distance dimensions greater than the threshold value. In this case, the hawks are faraway from the jackrabbit.

The SLO strategy is performed in part no. III of Table 2. The closer hawks, i.e., the blue circles o, approach the jackrabbit position. The faraway hawks stay still as watchers. For part no. IV, the normal boundary $[LB, UB]$ of the search space for the hawks is set for exploration and exploitation phases. In part no. V, the hawks perform the "surprise pounce" strategy of exploration and exploitation phases as illustrated in Fig. 1 of Section 2. In the exploitation phase, the hawks perform four conditional strategies: soft besiege, hard besiege, soft besiege with rapid dive, and hard besiege with rapid dive.

In part no. VI, the 2D contour of benchmark function F8 of CEC 2017 illustrates the moves of hawks' population ($N = 30$) by using random restart. When the hawks (i.e., the blue circles o) are trapped, by using the DO strategy, the trapped hawks in red stars ∗ try to escape by jumping out of the local optima to reach the global optimum, which is represented by a capital G G. In the last part, the effect of random restart is depicted on the 3D graph of F8, CEC 2017. The hawks in the original HHO, as presented by the blue circles o, are trapped in the local optima. At the same time, the hawks of the proposed HHO-JOS (red stars ∗) succeed in reaching the global optimum location, which is represented by a green circle o.

Table 2
The illustration of HHO-JOS in search space.

No.	Objective	Search Space
I	**Perform random hawks** SS is Search Space *LB*: original lower boundary *UB*: original upper boundary [G] Global optimum [∗] Red Stars represent *N* hawks Example *N*=15	
II	**Identification The Leading Hawks (SLO)** [⌒] Green Curve: threshold [--] Dashed Rectangle: current hawks boundary (*L* and *U*) [◆] Brown Diamond: jackrabbit position [∗] Red Stars: hawks position ≥ threshold [o] Blue Circles: hawks position < threshold [⁄] Shaded Area: area hawks position < threshold	
III	**Perform SLO** Hawk with less threshold [o] approach to rabbit position [◆] in *L* and *U* boundary. [∗] Red Stars hawks position > threshold defines as the **watchers**. [o] Blue Circles hawks position < threshold define as the **selective leaders**.	
IV	**Set Normal Search Space for exploration and exploitation** Hawks will perform exploration and exploitation using normal boundary *LB* and *UB*	
V	**Perform exploration and exploitation** Hawks perform a "surprise pounce" strategy on exploration and exploitation. On the exploitation phase, hawks perform four conditional strategies : ▪ Soft Besiege (SB) ▪ Hard Besiege (HB) ▪ SB rapid dive ▪ HB rapid dive	
VI	**Perform Random Restart in 2D of F8 CEC 2017** Comparison of original HHO and HHO-DO on *N*=30 [G] Green G: Global optimum F8 in 2D [o] Blue Circles: original HHO [∗] Red Stars: the trapped hawks try to escape from local optima using DO	
VII	**Perform Random Restart in 3D of F8 CEC 2017** [o] Green Circle: global optimum F8 in 3D [o] Blue Circles: original HHO trapped in local optima [∗] Red stars: the DO in HHO-JOS succeed in reaching the global optimum location	

4. Experiments and discussion

This section is divided into two parts: experimental setup and experimental results. The experimental results include required comprehensive statistical analysis such as Wilcoxon and comparison assessment with other algorithms, globally.

Table 3
CEC 2014 competition on single-objective real parameter numerical optimization.

Typology	No.	Functions	Optima, F_i^*
Unimodal Functions	1	Rotated High Conditioned Elliptic Function	100
	2	Rotated Bent Cigar Function	200
	3	Rotated Discus Function	300
Simple Multimodal Functions	4	Shifted and Rotated Rosenbrock's Function	400
	5	Shifted and Rotated Ackley's Function	500
	6	Shifted and Rotated Weierstrass Function	600
	7	Shifted and Rotated Griewank's Function	700
	8	Shifted Rastrigin's Function	800
	9	Shifted and Rotated Rastrigin's Function	900
	10	Shifted Schwefel's Function	1000
	11	Shifted and Rotated Schwefel's Function	1100
	12	Shifted and Rotated Katsuura Function	1200
	13	Shifted and Rotated HappyCat Function	1300
	14	Shifted and Rotated HGBat Function	1400
	15	Shifted and Rotated Expanded Griewank's plus Rosenbrock's Function	1500
	16	Shifted and Rotated Expanded Scaffer's F6 Function	1600
Hybrid Functions	17	Hybrid Function 1 (N = 3) Modified Schwefel's, Rastrigin's, High-Conditioned Elliptic	1700
	18	Hybrid Function 2 (N = 3) Bent Cigar, HGBat, Rastrigin's	1800
	19	Hybrid Function 3 (N = 4) Griewank's, Weierstrass, Rosenbrock's, Scaffer's F6	1900
	20	Hybrid Function 4 (N = 4) HGBat, Discus, Expanded Griewank's plus Rosenbrock's, Rastrigin's	2000
	21	Hybrid Function 5 (N = 5) Scaffer's F6, HGBat, Rosenbrock's, Modified Schwefel's, High-Conditioned Elliptic	2100
	22	Hybrid Function 6 (N = 5) Katsuura, HappyCat, Expanded Griewank's plus Rosenbrock's, Modified Schwefel's, Ackley's Function	2200
Composition Functions	23	Composition Function 1 (N = 5) Rotated Rosenbrock's, High Conditioned Elliptic, Rotated Bent Cigar, Rotated Discus	2300
	24	Composition Function 2 (N = 3) Schwefel's, Rotated Rastrigin's, Rotated HGBat	2400
	25	Composition Function 3 (N = 3) Rotated Schwefel's, Rotated Rastrigin's, Rotated High Conditioned Elliptic	2500
	26	Composition Function 4 (N = 5) Rotated Schwefel's, Rotated HappyCat, Rotated High Conditioned Elliptic, Rotated Weierstrass, Rotated Griewank's	2600
	27	Composition Function 5 (N = 5) Rotated HGBat, Rotated Rastrigin's, Rotated Schwefel's, Rotated Weierstrass, Rotated High Conditioned Elliptic	2700
	28	Composition Function 6 (N = 5) Rotated Expanded Griewank's plus Rosenbrock's, Rotated HappyCat Function, Rotated Schwefel's Function, Rotated Expanded Scaffer's F6, Rotated High Conditioned Elliptic	2800
	29	Composition Function 7 (N = 3) F17, F18, F19	2900
	30	Composition Function 8 (N = 3) F20, F21, F22	3000
Search Range: $[-100, 100]^D$			

Table 4
CEC 2017 competition on single-objective real parameter numerical optimization.

Typology	No.	Functions	Optima, F_i^*
Unimodal Functions	1	Shifted and Rotated Bent Cigar Function	100
	2	Shifted and Rotated sum of Differential Power Function	200
	3	Shifted and Rotated Zakhrov Function	300
Simple Multimodal Functions	4	Shifted and Rotated Rosenbrock's Function	400
	5	Shifted and Rotated Rastrigin's Function	500
	6	Shifted and Rotated Expanded Scaffer's F6 Function	600
	7	Shifted and Rotated Lunacek Bi_Rastringin's Function	700
	8	Shifted and Rotated Non-Continuous Rastringin's Function	800
	9	Shifted and Rotated Levy Function	900
	10	Shifted and Rotated Schwefel's Function	1000
Hybrid Functions	11	Hybrid Function 1 (N = 3) Zakhrov, Rosenbrock's, Rastrigin's	1100
	12	Hybrid Function 2 (N = 3) High-Conditioned Elliptic, Modified Schwefel's, Ben Cigar	1200
	13	Hybrid Function 3 (N = 3) Ben Cigar, Rosenbrock's, Lunacek Bi_Rastringin's	1300
	14	Hybrid Function 4 (N = 4) High-Conditioned Elliptic, Ackley, Schaffer's F7, Rastringin's	1400
	15	Hybrid Function 5 (N = 4) Ben Cigar, HGBat, Rastringin's, Rosenbrock's	1500
	16	Hybrid Function 6 (N = 4) Expanded Schaffer's F6, HGBat, Rosenbrock's, Modified Schwefel's	1600
	17	Hybrid Function 6 (N = 5) Katsuura, Ackley, Expanded Griewank's plus, Rosenbrock's, Schwefel's, Rastringin's	1700
	18	Hybrid Function 6 (N = 5) High-Conditioned Elliptic, Ackley, Rastringin's, HGBat, Discus	1800
	19	Hybrid Function 6 (N = 5) Bent Cigar, Rastringin's, Griewank's plus Rosenbrock's, Weierstrass, Expanded Schaffer's F6	1900
	20	Hybrid Function 6 (N = 5) HappyCat, Katsuura, Ackley, Rastringin's, Modified Schwefel's, Schaffer F7	2000
Composition Functions	21	Composition Function 1 (N = 3) Rosenbrock's, High Conditioned Elliptic, Rastringin's	2100
	22	Composition Function 1 (N = 3) Rastringin's, Griewank's, Modified Schwefel's	2200
	23	Composition Function 1 (N = 4) Rosenbrock's, Ackley, Modified Schwefel's, Rastringin's	2300
	24	Composition Function 2 (N = 4) Ackley, High-Conditioned Elliptic, Griewank's, Rastringin's	2400
	25	Composition Function 3 (N = 5) Rastringin's, HappyCat, Ackley Discus, Rosenbrock's	2500
	26	Composition Function 4 (N = 5) Expanded Schaffer's F6, Modified Schwefel's, Griewank's, Rosenbrock's, Rastringin's	2600
	27	Composition Function 5 (N = 6) HGBat, Rastringin's, Modified Schewel's, Bent Cigar, High-Conditioned Elliptic, Expanded Schaffer's F6	2700
	28	Composition Function 6 (N = 6) Ackley, Griewank, Discus, Rosenbrock, HappyCat, Expanded Schaffer's F6	2800
	29	Composition Function 7 (N = 3) F15, F16, F17	2900
	30	Composition Function 8 (N = 3) F15, F18, F19	3000
Search Range: $[-100, 100]^D$			

4.1. Experimental setup

The experiments were conducted on a Windows 10 with a CPU Intel® Core™ i9-7980XE CPU @ 2.60 GHz processor and RAM 64 GB. The algorithms were written in MATLAB. The experiments were exhibited to solve single-objective real parameter numerical optimization of CEC 2014 and CEC 2017. The CECs are the preferable standard benchmark problem set on single-objective real parameters and required a specific standard value of the parameters to run the experiment as follows:

a. The population size (N) equals to 30 ($N = 30$). Note: The population size is fixed.
b. On each experiment, the maximum number of runs (*maxRun*) is 51 runs.
c. We tested on 30 benchmark functions of CEC 2014 and 29 benchmark functions of CEC 2017. It is noted that only F2 function on CEC 2017 was deleted due to an unconfirmed result on the experiment.
d. Each benchmark function was tested on four numbers of variables (dimension = D): 10D, 30D, 50D and 100D.
e. The maximum number of function evaluations (*maxFE*) is set up based on 10,000 multiply by the dimensions of 10D, 30D, 50D, and 100D.
f. For each run, the maximum number of iteration (T) is defined by dividing *maxFE* with N.
g. The searching space is in the range of $[-100, 100]^D$, where the lower bound (LB) is -100, the upper bound (UB) is 100.

Detailed explanation of benchmark functions CEC 2014 (Table 3) and CEC 2017 (Table 4) are available in the technical reports of Liang et al., 2013 and Awad et al., 2016, respectively. Both CECs are classified into four categories: unimodal functions, multimodal functions, hybrid functions, and composition functions. Although the CECs are classified by the same categories, some benchmark functions are defined by the names and the numbers differently. The numbers of benchmark functions on CEC 2014 are specified as follows: unimodal (F1-F3), multimodal function (F4-F16), hybrid functions (F17-F22), and composition functions (F23-F30). In CEC 2017, the specification of unimodal functions (F1-F3) remains the same. Still, the basic function of each benchmark is different. There is a reduction function on the multimodal function (F4-F10), six functions less compared to CEC 2014. Additionally, four numbers of benchmark functions are included in the hybrid functions (F11 − F20) and some of the functions are composed differently in CEC 2014. In composition functions (F20 − F30) of CEC 2017, three additions benchmark functions and the functions are composed differently from CEC 2014.

The benchmark functions named in CEC 2014 are determined by rotated and non-separable on unimodal functions, whereas multimodal functions are either shifted or rotated and separable or non-separable. Hybrid functions are constructed from different subcomponent functions randomly. Composite functions are created by a combination of two or more hybrid functions. Meanwhile, CEC 2017 also used these criteria; the differences are on some of the basic functions that composed in each category on each benchmark function. Regardless of those differences, both CEC 2014 and CEC 2017 are applied on real parameter single-objective numerical optimization.

The performance of HHO-JOS and the comparison were evaluated by a scoring metric [55] on CEC 2014 and CEC 2017, as shown in Table 10. The scoring metric is comprised of the sum of the error (SE) and the sum of the rank (SR). The highest score for each part is 50. Therefore, the highest total score of SE and SR is 100. Each score is influenced by the number of the function F. Meanwhile, ef represents the best fitness value of each function in each algorithm. On the summation of SE and SR the best fitness value of every algorithm in 10D, 30D, 50D, and 100D; then, is multiplied by the weights of 0.1, 0.2, 0.3, and 0.4, respectively.

The formulas of SE and the total score of SE are presented in Eq. (21) and Eq. (22), respectively.

$$SE = 0.1*\left(\sum_{i=1}^{F} ef_{10D}\right) + 0.2*\left(\sum_{i=1}^{F} ef_{30D}\right) + 0.3*\left(\sum_{i=1}^{F} ef_{50D}\right) + 0.4*\left(\sum_{i=1}^{F} ef_{100D}\right) \tag{21}$$

$$\text{Total Score } SE = \left(1 - \frac{SE - SE_{min}}{SE}\right)*50 \tag{22}$$

where SE_{min} is the minimal value of the sum mean error for the 10, 30, 50, and 100 dimensions. The formulas of SR and the total score of SR are stated in Eq. (23) and Eq. (24), respectively.

$$SR = 0.1*\left(\sum_{i=1}^{F} ef_{10D}\right) + 0.2*\left(\sum_{i=1}^{F} ef_{30D}\right) + 0.3*\left(\sum_{i=1}^{F} ef_{50D}\right) + 0.4*\left(\sum_{i=1}^{F} ef_{100D}\right) \tag{23}$$

$$\text{Total Score } SR = \left(1 - \frac{SR - SR_{min}}{SR}\right)*50 \tag{24}$$

where, SR_{min} is the minimal value of the sum rank for the 10, 30, 50, and 100 dimensions. Thus, the overall total score is concluded in Eq. (25)

$$\text{Total Score} = \text{Total Score } SE + \text{Total Score } SR \tag{25}$$

4.2. Experimental results

The experiments were conducted with five considerations as follows. Section 4.2.1 discusses the Random Jump Strategy nomination of jumping rate (Jr) on HHO-JOS. Section 4.2.2 exhibits the succeed JOS behavior and its exploration ability. Section 4.2.3 presents the HHO-JOS statistical analysis of Wilcoxon Signed Rank Test performance compared to its algorithms' competitors, i.e., the original HHO, HHO-DO, and HHO-SLO. Section 4.2.4 presents the dominancy comparative of JOS with other OBL developments on HHO. Section 4.2.5 shows the rank of HHO-JOS and its competitors based on the scoring metric. Section 4.2.6 concludes the impact of JOS on the improvement and effectiveness of HHO with comparison to its competitors. Section 4.2.7 compares the mean ranks of HHO-JOS to the leading optimization algorithms by using Friedman Mean Rank Evaluation. In section 4.2.8, the stability of the HHO-JOS convergent curve is compared to its competitors.

4.2.1. Experimental results of jumping rate (Jr) on CEC 2014 and CEC 2017

The Jumping Rate of DO influences the diversity of the hawks' population on the proposed HHO-JOS. Fig. 8 presents the results of Jr performance evaluation on the CEC 2014 and CEC 2017 using a scoring metric. The experiment was conducted from 0.05 to 0.95 with an increment of 0.05 on CEC 2014 and CEC 2017. In total, there are 19 Jr experimental results for each CEC. The overall trend shows that Jr at 0.25 is sufficient $Jr$for HHO-JOS.

In Fig. 8, a square marker represents the Jr on CEC 2014 and a star marker represents the Jr on CEC 2017. The blue lines present the SE score on CEC 2014 and CEC 2017. The red lines indicate the SR score on CEC 2014 and CEC 2017, and the green lines present the total score on CEC 2014 and CEC 2017. SE is delineated in a straight blue line, SR in a dashed red dot line, and the total score in a dashed green line. To recognize the trend in SE,SR, and the total score of SE and SR, we selected the top five jumping rates for both CEC from the 19 experimental results of each CEC. Next, we chose the best three jumping rates to obtain the best value of Jr.

The fluctuation in SE is influenced by the diversity of hawks' population of the proposed HHO-JOS. If Jr increases, it means the diversity of hawks' population is getting higher; therefore, the algorithm can escape from the local optima. This condition applies vice versa. The detailed

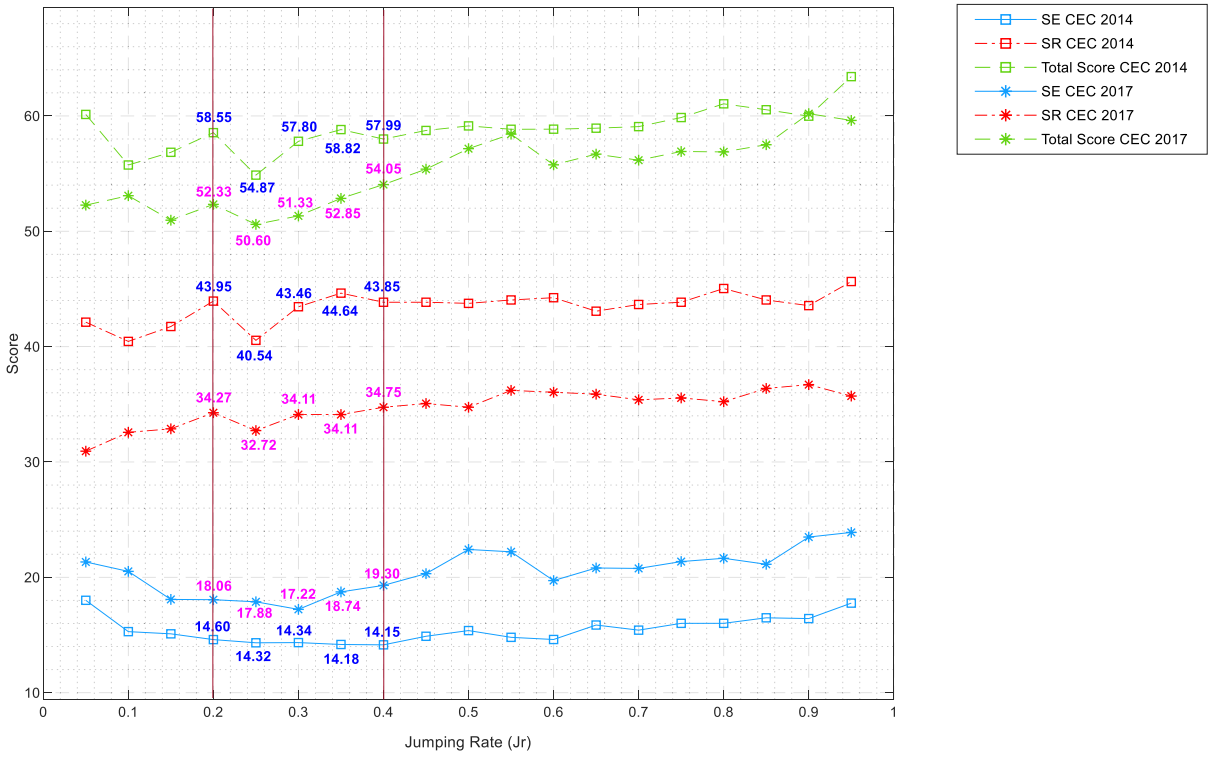


Fig. 8. The results of *SE*,*SR*, and total score of *SE* and *SR* varied by *Jr* on CEC 2014 and CEC 2017.

description of *SE* on CEC 2014 shows that the top-five jumping rates (*Jrs*) are 0.2, 0.25, 0.30, 0.35, and 0.4 with *SE* results at 14.6018, 14.3248, 14.3417, 14.1777, and 14.1472 respectively. Interestingly, the top-five ranked *Jrs* on CEC 2017 are the same as the top-five ranked *Jrs* on CEC 2014. For CEC 2017, the scores of the top-five ranked jumping rates are 18.061, 17.8774, 17.2215, 18.7365, and 19.3029. We, then, selected three best *Jrs* from those top-five ranked *Jrs*, i.e., 18.061, 17.8774, and 17.2215 on*Jr* 0.2, 0.25, and 0.3, respectively. In case of CEC 2014, the scores of the three best *Jrs* are 14.6018, 14.3248, and 14.3417 on *Jrs* 0.2, 0.25, and 0.3, respectively. Since the results of CEC 2017 are aligned with the results obtained from CEC 2014, *SE*, *SR*, and the total score of *SE* and *SR* are employed to determine the best jumping rate for both CEC 2014 and CEC 2017.

The line graph of *SR* results on *Jrs* at 0.2, 0.25, and 0.3 of CEC 2014 and CEC 2017 shows the same trend as the ranks. In detail, the 0.2 *Jr* on CEC 2014 and CEC 2017 reach the peaks at 34.27 and 43.95, respectively. However, for 0.25 *Jr* on CEC 2014 and CEC 2017, the sum of ranks are slightly lower at 32.72 and 40.54, then they raise again for 0.3 *Jr* on CEC 2014 and CEC 2017 at 34.11 and 43.36 respectively. In overall, *SR* and the total score of *SE* and *SR* share the same trend. Finally, after the analysis and consideration above, the best *Jr* among 0.2, 0.25, and 0.3 measuring from the experiments on CEC 2014 and CEC 2017 is clearly at 0.25.

4.2.2. The behavior of JOS

It is essential to examine the divergence of the proposed HHO-JOS compared to its original HHO from the population diversity. If the population is highly diverse, it means the population has difficulty converging. However, if the population has low diversity, then premature convergence might occur. The proposed JOS embedded on HHO shows the proportional balance on the diversity to avoid those occurrences.

The comparison experiment of the original HHO (blue circle o) and HHO-JOS (red star *) on two-dimensions of F6 and F8 on CEC 2017 are illustrated in Fig. 9(a) and Fig. 9(c), respectively. The global optimum location is depicted by G. Both Fig. 9(a) and Fig. 9(c) show that the original HHO was experiencing the stagnation condition trapped in the local optima. Meanwhile, Fig. 9(b) and Fig. 9(d) show the dominancy of HHO-JOS (the red line star) over HHO (the blue line). The x −axis depicts the number of iterations of the diversity. The y − axis represents the divergent movement's fluctuation by the average of the variance (σ^2) denoted in Eq. (22).

$$\sigma^2 = \frac{\sum_{i=1}^{N}\left(X_i - \overline{X}\right)^2}{N} \tag{22}$$

The comparison of HHO and HHO-JOS in two dimensions is shown in Fig. 9. The contour graphs on Fig. 9(a) and Fig. 9(c) illustrate the success of the HHO-JOS and it reached the global optimum. Meanwhile, Fig. 9 (b) shows that the original HHO in F6 is moving closer to zero point just after 50 iterations. However, the proposed HHO-JOS still shows the jumping movement. Further proof of the capability of HHO-JOS compared to HHO is illustrated in F8 with many various local optima. Compared to F6, the performance of HHO on F8 approaches zero-point swifter at 30 iterations, shown in Fig. 9(d), and the jumping experience of HHO-JOS appears more frequently. Therefore, in Fig. 9, we can see that HHO-JOS tries to enlarge the search space to escape the trap at the local optima and to reach the global optimum location.

The trends of original HHO and HHO-JOS diversity in two-dimensions (Fig. 9(b) and Fig. 9(d)) also show the same behavior in 10D, 30D, 50D, and 100D. Fig. 10 depicts one of the trends in 30D on F29 where the proficiency of fitness values of HHO and HHO-JOS reach 4636.1875 and 4101.4956, respectively. Note that the lower fitness value is better. The figure shows a rapid iteration which influences by the dimensions. Why benchmark function 29? The supporting evidence in Table 5 shows that F29 succeeds in competing with the opponent algorithm of HHO, HHO-DO, and HHO-SLO. From Fig. 10, we can still see that when the original HHO is already approaching the zero point, the HHO-JOS still tries to enlarge its diversity.

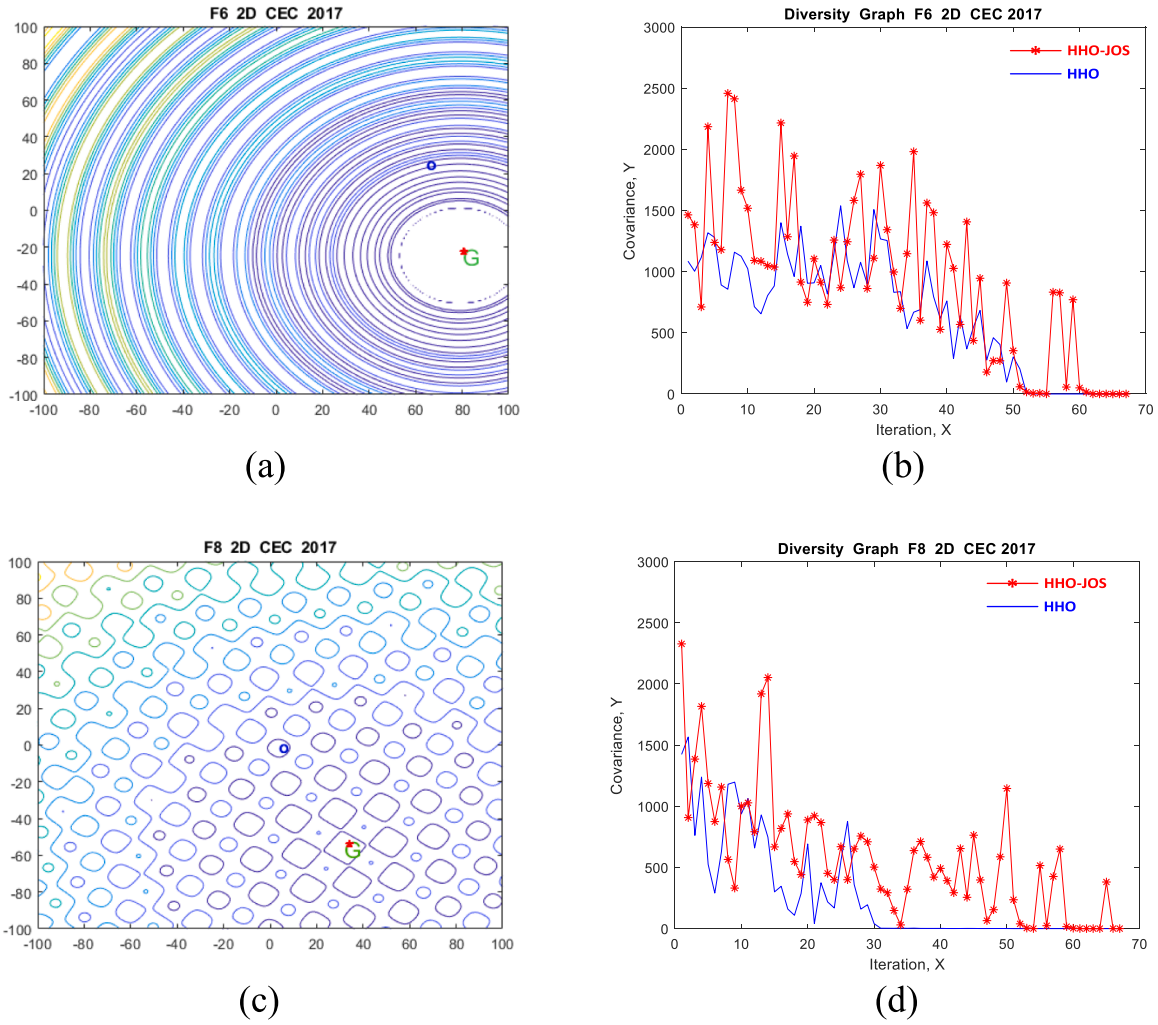


Fig. 9. Divergence of HHO and HHO-JOS on F6 and F8 CEC 2017 in two-dimensions.

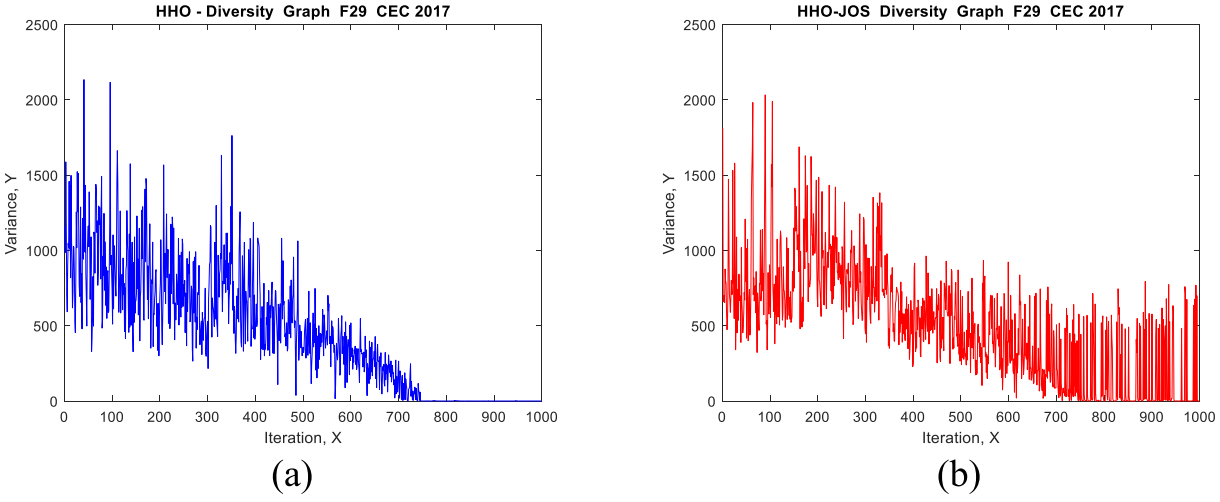


Fig. 10. The divergence of HHO and HHO-JOS on 30D of F29 in CEC 2017

4.2.3. The Wilcoxon Signed-Rank test on the winning HHO-JOS (Jr 0.25) comparing with HHO, HHO-DO, and HHO-SLO

The statistical analysis results obtained from Wilcoxon Signed-Rank Test (Woolson, 2005), as presented in Table 5, characterize the performance of the proposed HHO-JOS compared to its competitors' algorithm HHO, HHO-DO, and HHO-SLO on CEC 2014 and CEC 2017 on four types of dimension 10D, 30D, 50D, and 100D. The chosen competitors of HHO-JOS in Table 5 are the original HHO and the improved

Table 5
Wilcoxon analysis of the proposed HHO-JOS compared to the competitive algorithms HHO, HHO-DO, and HHO-SLO.

Proposed / Competitors	D	HHO-JOS					
		CEC 2014			CEC 2017		
		Win (+)	Tide (=)	Loss (-)	Win (+)	Tide (=)	Loss (-)
HHO	10D	19	3	8	23	0	6
	F	2,5,6,7,9,11,12,13, 15,16,17,19,22,24, 25,26,29,30	23,27,28	1,3,4,8,10, 14,20,21	1,3,5,6,7,8,9,11,13,16, 17,18,19,20,21,22,23,24, 26,27,28,29,30	-	4,10,12,14, 15,25
	30D	19	5	8	27	0	2
	F	2,5,6,7,8,9,10,11, 12,13,14,15,16,18, 19,20,22	23,25,27, 28,29	1,3,4,17, 21,24,26, 30	1,4,5,6,7,8,9,10,11,12,13, 14,15,16,17,18,19,20,21, 22,23,24,25,26,27,29,30	-	3,28
	50D	19	5	7	22	0	7
	F	1,2,5,6,7,8,9,10,11, 12,13,14,16,18,19, 22,24,26	23,25,27, 28,29	3,4,15,17, 20,21,30	1,5,6,7,8,9,10,13,15,16, 17,19,20,21,22,23,24,25, 26,27,29,30	-	3,4,11,12,14, 18,28
	100D	16	6	8	24	0	5
	F	1,2,5,6,7,8,9,10,11, 12,14,16,17,18,22, 24	23,25,26, 27,28,29	3,4,13,15, 19,20,21, 30	1,5,6,7,8,9,10,11,12,13, 14,15,16,17,18,19,20,21, 22,23,26,27,29,30	-	3,4,24,25,28
HHO-DO	10D	19	3	8	22	0	7
	F	2,6,7,8,9,10,11,13, 15,16,18,19,20,21, 22,24,25,26,30	23,27,28	1,3,4,5,12, 14,17,29	1,3,5,6,7,8,9,10,11,13,16, 17,18,19,20,21,22,23,26, 27,29,30	-	4,12,14,15, 24,25,28
	30D	20	5	5	26	0	3
	F	1,2,3,5,6,7,8,9,10, 11,12,13,14,15,16, 18,19,20,22,26	23,25,27, 28,29	4,17,21, 24,30	1,4,5,6,7,8,9,10,11,12,13, 15,16,17,18,19,20,21,22, 23,24,25,26,27,29,30	-	3,14,28
	50D	21	5	4	25	0	4
	F	1,2,3,5,6,7,8,9,10, 11,12,13,14,15,16, 18,19,20,22,24,26	23,25,27, 28,29	4,17,21,30	1,5,6,7,8,9,10,11,12,13, 14,15,16,17,19,20,21,22, 23,24,25,26,27,29,30	-	3,4,18,28
	100D	20	5	5	27	0	2
	F	1,2,3,4,5,6,7,8,9,10, 11,12,14,16,17,18, 20,22,24	23,25,26, 27,28,29	13,15,19, 21,30	1,4,5,6,7,8,9,10,11,12,13, 14,15,16,17,18,19,20,21, 22,23,24,26,27,28, 29,30	-	3,25
HHO-SLO	10D	17	3	10	17	0	12
	F	2,5,7,9,11,12,14,15, 16,17,18,21,22,24, 25,26,29	23,27,28	1,3,4,6,8, 10,13,19, 20,30	1,3,7,8,9,13,17,18,21,22,23 ,24,25,26,27,29,30	-	4,5,6,10,11, 12,14,15,16, 19,20,28
	30D	16	4	10	17	0	12
	F	2,5,6,7,8,9,11,12, 13,14,15,16,18,19, 29,30	23,25,27, 28	1,3,4,10, 17,20,21, 22,24,26	1,5,6,7,8,9,10,11,12,13, 15,18,19,20,21,22,29	-	3,4,14,16,17, 23,24,25,26, 27,28,30
	50D	16	5	9	17	0	12
	F	1,2,5,6,7,8,9,10,11, 12,14,16,18,22,24, 30	23,25,27, 28,29	3,4,13,15, 17,19,20, 21,26	1,5,6,7,8,9,10,11,13,15, 16,19,20,21,22,24,29	-	3,4,12,14,17, 18,23,25,26, 27,28,30
	100D	16	6	8	22	0	7
	F	1,2,5,6,7,8,9,10,11, 12,14,16,17,18,22, 24	23,25,26, 27,28,29	3,4,13,15, 19,20,21, 30	1,5,6,7,8,9,10,11,12,13, 14,15,16,18,19,20,21, 22,23,26,29,30	-	3,4,17,24,25, 27,28

version of HHO with Selective Leading Opposition (SLO) and Dynamic Opposition (DO). The purpose of the comparison is to recognize the performance of the proposed algorithm and the competitors' algorithms classified by the winning (+), tide (=), and loss (−) benchmark functions of the CEC 2014 and CEC 2017.

The benchmark functions of the CECs are presented in numbers as shown in Table 5. In general, the total number of winning benchmark functions in CEC 2014 is between 15 and 20. The number of tide functions are above 3 and below 6, and the loss is not more than 10. This condition shows the overall dominance of HHO-JOS over the competitors. Meanwhile, in CEC 2017, regardless of no tide, there are still losses around 2 to 12 with a various number of benchmark functions. Nevertheless, the number of winning benchmark functions in CEC 2017 is slightly better in the dimensions comparing to the number of winning functions in CEC 2014. When the HHO-JOS is compared to HHO-SLO, the number of loss functions HHO-JOS is greater than its competitors. Table 5 shows that HHO-SLO wins in the lower dimension. Despite the loss, HHO-JOS still shows a winning performance.

Comparing to all competitors on CEC 2014 in all dimensions, HHO-JOS constantly tides on functions 23, 27, and 28. If there are additional tide functions, mostly are 25, 26, 28, and 29. It means that the HHO-JOS competed on CEC 2014 will have tide condition with its competitors in complex functions, i.e., hybrid and composite functions. However, the loss benchmark functions in CEC 2014 are mostly 3, 4, 15, and 17. Therefore, the proposed HHO-JOS is suitable for complex problems. Regardless there are some tide and loss benchmark functions on CEC 2014, the overall condition of HHO-JOS is still winning mostly on benchmark functions 2, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 16, 18, 22, and 24.

Meanwhile, in CEC 2017, regardless of no tide, the HHO-JOS comparing to other competitors shows that benchmark functions 3 and 28 are mostly on the lost side. However, benchmark function 3 (F3) on 10D of every competitive algorithm is on the winning side. Regardless the naught benchmark function on CEC 2017, benchmark functions 1, 5, 7, 8, 9, 13, 17, 18, 21, 22, 23, 24, 25, 26, 27, 29, and 30 are frequently appeared on the winning position. To sum up, the results of the proposed HHO-JOS when comparing to the competitors in 10D, 30D, 50D, and 100D CEC 2014 and CEC 2017 show remarkable winning.

The benchmark functions of the CECs are presented in numbers as shown in Table 5. In general, the total number of winning benchmark functions in CEC 2014 is between 15 and 20. The number of tide functions are above 3 and below 6, and the loss is not more than 10. This

condition shows the overall dominance of HHO-JOS over the competitors. Meanwhile, in CEC 2017, regardless of no tide, there are still losses around 2 to 12 with a various number of benchmark functions. Nevertheless, the number of winning benchmark functions in CEC 2017 is slightly better in the dimensions comparing to the number of winning functions in CEC 2014. When the HHO-JOS is compared to HHO-SLO, the number of loss functions HHO-JOS is greater than its competitors. Table 5 shows that HHO-SLO wins in the lower dimension. Despite the loss, HHO-JOS still shows a winning performance.

Comparing to all competitors on CEC 2014 in all dimensions, HHO-JOS constantly tides on functions 23, 27, and 28. If there are additional tide functions, mostly are 25, 26, 28, and 29. It means that the HHO-JOS competed on CEC 2014 will have tide condition with its competitors in complex functions, i.e., hybrid and composite functions. However, the loss benchmark functions in CEC 2014 are mostly 3, 4, 15, and 17. Therefore, the proposed HHO-JOS is suitable for complex problems. Regardless there are some tide and loss benchmark functions on CEC 2014, the overall condition of HHO-JOS is still winning mostly on benchmark functions 2, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 16, 18, 22, and 24.

Meanwhile, in CEC 2017, regardless of no tide, the HHO-JOS comparing to other competitors shows that benchmark functions 3 and 28 are mostly on the lost side. However, benchmark function 3 (F3) on 10D of every competitive algorithm is on the winning side. Regardless the naught benchmark function on CEC 2017, benchmark functions 1, 5, 7, 8, 9, 13, 17, 18, 21, 22, 23, 24, 25, 26, 27, 29, and 30 are frequently appeared on the winning position. To sum up, the results of the proposed HHO-JOS when comparing to the competitors in 10D, 30D, 50D, and 100D CEC 2014 and CEC 2017 show remarkable winning.

The complete comparison results among the proposed HHO-JOS, HHO, HHO-DO, and HHO-SLO are shown in Table 5. To conceive the detailed knowledge beyond the numbers of benchmark functions, the experimental results were broken down regarding its dimension classification as presented in Tables 6–9. The purpose of this breakdown is to clearly recognize the pattern and characteristic benchmark functions on each classification. Further discussions are explained in the following subsections:

4.2.4. Wilcoxon analysis of HHO-JOS compared to HHO, HHO-DO, and HHO-SLO on 10D

Table 6 shows that benchmark functions 23, 27, and 28 on CEC 2014 frequently appear on the tide. The loss benchmark functions appearing in all 10D CEC 2014 are 1, 3, and 4. The stable winning benchmark functions are 2, 7, 9, 11, 15, 22, 24, 25, and 26. Meanwhile, on CEC 2017, the number of winnings of HHO-JOS compared to HHO and HHO-DO are relatively high on 23, 22, and 17, respectively. Despite that, the number of winning on HHO-SLO is equal at 17 but the difference is on the benchmark functions of CEC 2014 and CEC 2017.

4.2.5. Wilcoxon analysis of HHO-JOS compared to HHO, HHO-DO, and HHO-SLO on 30D

The tide benchmark functions, in Table 7, on 30D of CEC 2014 are constantly on 23, 25, 27, and 28; however, the benchmark function 29 wins on HHO-SLO. Moreover, the loss is constantly on 4, 17, 21, and 24. The winning is constantly on 2, 5, 6, 7, 8, 9, 11, 12, 13, 14, 15, 16, 18, and 19. It means that the constant winning benchmark is more than 50 percent of the winning benchmark functions. In CEC 2017, the winning is constantly on 1, 5, 6, 7, 8, 9, 10, 11, 12, 13, 15, 16, 18, 19, 20, 21, 22, and 29. These winning numbers are quite similar to CEC 2014; the difference in CEC 2017 only on F2 and additional functions on 20, 21, and 29; the loss is constantly on 3 and 28.

4.2.6. Wilcoxon analysis of HHO-JOS compared to HHO, HHO-DO, and HHO-SLO on 50D

The noticeable winning comparison 50D experimental results, in Table 8, shows a very slightly different in CEC 2014 and CEC 2017 comparing to the number of winning in 30D. For tide of 50D CEC 2014, the benchmark function 29 constantly appears. It means that the number of the tide on CEC 2014 stays average with the change.

The numbers of losses HHO-JOS compared to HHO and HHO-DO are equal, i.e., 7 and 4, in CEC 2014 and CEC 2017 respectively. However, when comparing HHO-JOS to HHO-SLO, the loss numbers of HHO-JOS is slightly increased on both CEC compared to other competitors. In other words, the HHO-SLO performance is relatively close to HHO-JOS as depicted in Table 10.

4.2.7. Wilcoxon analysis of HHO-JOS compared to HHO, HHO-DO, and HHO-SLO on 100D

Interestingly, there is a fairly minor change on the tide benchmark function of 100D CEC 2014, an additional benchmark function 26 shown in Table 9. Usually, the position benchmark function 26 CEC 2014 is either on the winning or the losing side. Table 9 shows a significant difference in winning and losing of HHO-JOS on HHO-DO on CEC 2017. The number of winning HHO-JOS is 27, and the loss is 2. There is no great change on the loss side of HHO-JOS on 100D CEC 2014 comparing to HHO, HHO-DO, and HHO-SLO; mostly, the HHO-JOS losses in benchmark functions 13, 15, 19, 21, and 30. In the case of CEC 2017, the loss of HHO-JOS is constant on benchmark functions 3 and 25 but still well perform on the winning side.

4.2.8. Dominancy comparative of JOS with other OBL development on HHO

The experimental results in Table 10 show a comparison among the proposed algorithm (HHO-JOS), HHO-DO, HHO-SLO, and HHO-SO. The total functions exhibited in Table 10 are evaluated in 100 dimensions on 29 benchmark functions of CEC 2017. The experiment concludes from the mean of 51 runs which were experimented successively according to each function. The sign (+) at the bottom of Table 10 indicates that the development of opposition is better than the original and this positive

Table 6
Wilcoxon analysis of HHO-JOS compared to HHO, HHO-DO, and HHO-SLO on 10D.

Proposed / Competitors	D	HHO-JOS					
		CEC 2014			CEC 2017		
		Win (+)	Tide (=)	Loss (-)	Win (+)	Tide (=)	Loss (-)
HHO	10D	19	3	8	23	0	6
	F	2,5,6,7,9,11,12,13,15,16,17,19,22,24,25,26,29,30	23,27,28	1,3,4,8,10,14,20,21	1,3,5,6,7,8,9,11,13,16,17,18,19,20,21,22,23,24,26,27,28,29,30	-	4,10,12,14,15,25
HHO-DO	10D	19	3	8	22	0	7
	F	2,6,7,8,9,10,11,13,15,16,18,19,20,21,22,24,25,26,30	23,27,28	1,3,4,5,12,14,17,29	1,3,5,6,7,8,9,10,11,13,16,17,18,19,20,21,22,23,26,27,29,30	-	4,12,14,15,24,25,28
HHO-SLO	10D	17	3	10	17	0	12
	F	2,5,7,9,11,12,14,15,16,17,18,21,22,24,25,26,29	23,27,28	1,3,4,6,8,10,13,19,20,30	1,3,7,8,9,13,17,18,21,22,23,24,25,26,27,29,30	-	4,5,6,10,11,12,14,15,16,19,20,28

Table 7
Wilcoxon analysis of HHO-JOS compared to HHO, HHO-DO, and HHO-SLO on 30D.

Proposed / Competitors	D	HHO-JOS					
		CEC 2014			CEC 2017		
		Win (+)	Tide (=)	Loss (-)	Win (+)	Tide (=)	Loss (-)
HHO	30D	19	5	8	27	0	2
	F	2,5,6,7,8,9,10,11, 12,13,14,15,16, 18,19,20,22	23,25,27, 28,29	1,3,4,17, 21,24,26, 30	1,4,5,6,7,8,9,10,11,12,13, 14,15,16,17,18,19,20,21, 22,23,24,25,26,27,29,30	-	3,28
HHO-DO	30D	20	5	5	26	0	3
	F	1,2,3,5,6,7,8,9,10, 11,12,13,14,15, 16,18,19,20,22,26	23,25,27, 28,29	4,17,21, 24,30	1,4,5,6,7,8,9,10,11,12,13, 15,16,17,18,19,20,21,22, 23,24,25,26,27,29,30	-	3,14,28
HHO-SLO	30D	16	4	10	17	0	12
	F	2,5,6,7,8,9,11,12, 13,14,15,16,18, 19,29,30	23,25,27, 28	1,3,4,10, 17,20,21, 22,24,26	1,5,6,7,8,9,10,11,12,13, 15,18,19,20,21,22,29	-	3,4,14,16,17, 23,24,25,26, 27,28,30

Table 8
Wilcoxon analysis of HHO-JOS compared to HHO, HHO-DO, and HHO-SLO on 50D.

Proposed / Competitors	D	HHO-JOS					
		CEC 2014			CEC 2017		
		Win (+)	Tide (=)	Loss (-)	Win (+)	Tide (=)	Loss (-)
HHO	50D	19	5	7	22	0	7
	F	1,2,5,6,7,8,9,10, 11,12,13,14,16, 18,19,22,24,26	23,25,27, 28,29	3,4,15,17, 20,21,30	1,5,6,7,8,9,10,13,15,16, 17,19,20,21,22,23,24, 25,26,27,29,30	-	3,4,11,12,14, 18,28
HHO-DO	50D	21	5	4	25	0	4
	F	1,2,3,5,6,7,8,9, 10,11,12,13,14, 15,16,18,19,20, 22,24,26	23,25,27, 28,29	4,17,21,30	1,5,6,7,8,9,10,11,12,13, 14,15,16,17,19,20,21, 22,23,24,25,26,27,29,30	-	3,4,18,28
HHO-SLO	50D	16	5	9	17	0	12
	F	1,2,5,6,7,8,9,10, 11,12,14,16,18, 22,24,30	23,25,27, 28,29	3,4,13,15,17, 19,20,21,26	1,5,6,7,8,9,10,11,13,15, 16,19,20,21,22,24,29	-	3,4,12,14,17, 18,23,25,26, 27,28,30

Table 9
Wilcoxon analysis of HHO-JOS compared to HHO, HHO-DO, and HHO-SLO on 100D.

Proposed / Competitors	D	HHO-JOS					
		CEC 2014			CEC 2017		
		Win (+)	Tide (=)	Loss (-)	Win (+)	Tide (=)	Loss (-)
HHO	100D	16	6	8	24	0	5
	F	1,2,5,6,7,8,9,10, 11,12,14,16,17, 18,22,24	23,25,26, 27,28,29	3,4,13,15, 19,20,21,30	1,5,6,7,8,9,10,11,12,13,14, 15,16,17,18,19,20,21,22, 23,26,27,29,30	-	3,4,24,25, 28
HHO-DO	100D	20	5	5	27	0	2
	F	1,2,3,4,5,6,7,8,9, 10,11,12,14,16, 17,18,20,22,24	23,25,26, 27,28,29	13,15,19, 21,30	1,4,5,6,7,8,9,10,11,12,13, 14,15,16,17,18,19,20,21, 22,23,24,26,27,28,29,30	-	3,25
HHO-SLO	100D	16	6	8	22	0	7
	F	1,2,5,6,7,8,9,10, 11,12,14,16,17, 18,22,24	23,25,26, 27,28,29	3,4,13,15,19, 20,21,30	1,5,6,7,8,9,10,11,12,13,14, 15,16,18,19,20,21,22,23, 26,29,30	-	3,4,17,24, 25,27,28

indication are also highlighted. The sign (−) denotes the value of objective function is slightly higher than the original HHO.

By using the original HHO as a baseline, we can summarize that the performance of HHO-JOS is similar to HHO-SLO, since both algorithms have positive results from 27 benchmark functions, and is following by HHO-DO and HHO-SO at 13 and 3 benchmark functions respectively. These numbers describe that HHO-JOS and HHO-SLO show the dominancy among their competitors. Nevertheless, there is a much greater different gap in the objective function between HHO-JOS versus the original HHO and HHO-SLO versus the original HHO. The difference of winning gap on benchmark functions obtained from HHO-JOS is signified by the asterisk (*) marker. We can conclude that the objective function on HHO-JOS completely outperforms from HHO-SLO, HHO-SO, HHO-DO, and the original version of HHO.

4.2.9. Scoring metric result

The main point of Table 11 is to show the impact and improvement provided by HHO-JOS comparing to the original version of HHO and also to determine the position of HHO-JOS when it is ranking among various optimization algorithms in metaheuristic based on the total score. The total score is a performance result evaluated by using the scoring metric formula (Awad et al., 2016) on CEC 2014 and CEC 2017. The position of HHO-JOS in CEC 2017 and CEC 2014 rankings are 6th and 3rd with the total scores of 13.81 and 38.58 respectively. The results confirm that HHO-JOS make a great improvement from the original HHO.

In CEC 2014, HHO-JOS and HHO are the 3rd and 5th on the rank respectively, and the 1st and 2nd positions are MTDE and SMA. ABC is the only algorithm that came between HHO-JOS and HHO. Nevertheless, the HHO-JOS is still in the leading position. For CEC 2017 ranking, the proposed HHO-JOS is the 6th and HHO is the 14th. Although HHO-JOS is not in the top 3 and is not able to defeat MTDE, SMA, ABC, SLPSO, and CMA-ES, there are several algorithms in between HHO-JOS and HHO. Those algorithms are PSO, GWO, SOGWO, AOA, KH and MA.

Table 10
Comparison of HHO-JOS, HHO-DO, HHO-SLO, and HHO-SO on the CEC 2017 benchmark functions (100 dimensions, 51 runs).

F	HHO	HHO-JOS	HHO-DO	HHO-SLO	HHO-SO
F1	2.18E + 08	**2.46E + 05***	**1.28E + 08**	**2.15E + 08**	2.71E + 08
F3	3.76E + 04	5.79E + 04	5.48E + 04	**3.23E + 04**	1.00E + 05
F4	8.37E + 02	**8.06E + 02**	8.88E + 02	**8.22E + 02**	8.98E + 02
F5	1.42E + 03	**1.31E + 03***	**1.38E + 03**	**1.40E + 03**	1.46E + 03
F6	6.79E + 02	**6.36E + 02***	**6.73E + 02**	**6.49E + 02**	**6.77E + 02**
F7	3.55E + 03	**2.69E + 03***	**3.28E + 03**	**3.18E + 03**	3.56E + 03
F8	1.85E + 03	**1.69E + 03***	**1.83E + 03**	**1.76E + 03**	1.89E + 03
F9	3.16E + 04	**2.32E + 04**	**2.70E + 04**	**3.08E + 04**	4.63E + 04
F10	1.85E + 04	**1.58E + 04***	**1.83E + 04**	**1.77E + 04**	1.88E + 04
F11	2.97E + 03	**2.65E + 03***	3.07E + 03	**2.63E + 03**	3.02E + 03
F12	2.79E + 08	**1.27E + 08***	3.48E + 08	**2.31E + 08**	3.30E + 08
F13	4.05E + 06	**4.79E + 04***	**2.43E + 05**	**3.50E + 06**	4.93E + 06
F14	8.10E + 05	**6.98E + 05**	9.00E + 05	8.38E + 05	1.16E + 06
F15	1.14E + 06	**2.69E + 04***	**8.61E + 04**	**7.91E + 05**	1.38E + 06
F16	7.04E + 03	**6.36E + 03***	**7.01E + 03**	**6.77E + 03**	7.12E + 03
F17	6.10E + 03	**5.91E + 03***	6.33E + 03	**5.68E + 03**	6.19E + 03
F18	1.79E + 06	**7.44E + 05***	1.99E + 06	**1.50E + 06**	2.06E + 06
F19	4.03E + 06	**1.46E + 05***	**3.98E + 06**	**1.03E + 06**	5.22E + 06
F20	5.84E + 03	**5.49E + 03***	6.05E + 03	**5.72E + 03**	**5.57E + 03**
F21	3.80E + 03	**3.37E + 03***	3.91E + 03	**3.49E + 03**	3.94E + 03
F22	2.24E + 04	**1.99E + 04***	**2.23E + 04**	**2.12E + 04**	2.26E + 04
F23	4.74E + 03	**4.12E + 03***	4.84E + 03	**4.33E + 03**	4.84E + 03
F24	5.55E + 03	**5.24E + 03***	5.62E + 03	**5.34E + 03**	5.83E + 03
F25	3.50E + 03	**3.47E + 03**	3.54E + 03	3.53E + 03	3.56E + 03
F26	2.39E + 04	**2.22E + 04***	**2.35E + 04**	**2.33E + 04**	**2.26E + 04**
F27	4.10E + 03	**3.98E + 03***	4.22E + 03	**3.89E + 03**	4.17E + 03
F28	3.56E + 03	3.57E + 03	3.59E + 03	**3.56E + 03**	3.59E + 03
F29	8.70E + 03	**7.87E + 03***	8.87E + 03	**8.05E + 03**	9.00E + 03
F30	2.06E + 07	**6.46E + 06***	2.56E + 07	**9.95E + 06**	2.89E + 07
	+	27	13	27	3
	–	2	16	2	26

Among the algorithms that came in between, SO on GWO is a development of OBL. However, SOGWO did not perform adequately on both CEC compared to its original version of GWO. As exhibited in Section 3, the proposed JOS is a joint of SLO and DO, where SLO is an improvement from the existing mechanism, SO. Therefore, the development of OBL effectively influences the performance of JOS on HHO which produces a sufficient impact on both CEC. Although the HHO-JOS position is not at the top rank, JOS has the capability to improve HHO to be considered as a leading algorithm.

4.2.10. The effectiveness of HHO-JOS

The effectiveness of HHO-JOS was measured by the Kruskal-Wallis (Ostertagová et al., 2014) test running on 100 dimensions, CEC 2017. The purpose of this test is to comprehend the whole picture of the development of HHO-JOS. HHO-JOS was compared with the state-of-the-art optimization algorithms having the scoring results in between HHO-JOS and HHO (see Table 11). Those algorithms are GOA, PSO, GWO, SOGWO, AOA, KH, and MA. Kruskal-Wallis classifies the average of final objective values of all benchmark functions of CEC 2017 from F1, F3-F30 on 100 dimensions in each algorithm as one group. The classification of the Kruskal-Wallis shown in Fig. 11 are based on the experimental results in Table 12. In case of 10, 30, and 50 dimensions, they produce relatively the same pattern as the 100 dimensions as shown in Fig. 11. The lower the mean rank is, the better performance we can achieve from the algorithm.

According to Fig. 11, HHO-JOS is a leader among its competitors and the original HHO, since HHO-JOS has the lowest mean ranks. For AOA, the mean ranks came last. The closest to HHO-JOS is GOA followed by HHO, MA, and KH. The mean ranks of PSO starts after 100. Moreover, the original GWO and its improvement, SOGWO, did not show any difference on the mean ranks. In other words, SO did not help GWO to perform well insolving problems in CEC 2017′s benchmark functions on 100 dimensions. Moreover, we obtained similar results when doing the experiments on different number of dimensions in the benchmark

Table 11
The total scoring of HHO-JOS and its competitors.

Algorithms	Abbreviation	Rank Result		Total Score	
		CEC' 17	CEC' 14	CEC' 17	CEC' 14
Multi-Trial Differential Evolution (Nadimi-Shahraki et al., 2020)	MTDE	1	1	100	94.1488
Slime Mould Algorithm (Li et al., 2020)	SMA	2	2	28.5391	63.4489
Artificial bee colony (Karaboga & Basturk, 2007)	ABC	3	4	22.6913	35.1390
Social Learning Particle Swarm Optimization (Cheng & Jin, 2015)	SLPSO	4	6	18.4424	25.0867
Covariance Matrix Adaptation - Evolution Strategy (Hansen et al., 2003)	CMA-ES	5	7	14.1466	23.8116
Harris' Hawks Optimization - Joint Opposite Selection (Proposed)	**HHO-JOS**	**6**	**3**	**13.5947**	**38.8128**
Grasshoper Optimization Algorithm (Saremi et al., 2017)	GOA	7	8	13.4489	20.9781
Particle Swarm Optimization (Kennedy & Eberhart, 1995)	PSO	8	13	11.9787	18.6372
Grey Wolf Optimizer (Mirjalili et al., 2014)	GWO	9	15	11.7845	18.1472
Selective Opposition Grey Wolf Optimizer (Dhargupta et al., 2020)	SOGWO	10	14	11.6023	18.1652
Archimedes Optimization Algorithm (Hashim et al., 2021)	AOA	11	9	10.6126	20.2093
Krill Herd (Gandomi & Alavi, 2012)	KH	12	11	10.4014	19.4595
Mayfly Algorithm (Zervoudakis & Tsafarakis, 2020)	MA	13	12	10.1963	18.9979
Harris Hawk Optimization (Heidari et al., 2019)	**HHO**	**14**	**5**	**9.3249**	**29.3596**
Seagull Optimization Algorithm (Dhiman & Kumar, 2019)	SOA	15	17	8.6387	15.9904
Cuckoo Search (Yang & Deb, 2009)	CS	16	18	8.0030	13.8071
Moth Search (Wang, 2018)	MS	17	10	7.9841	20.0939
Whale Optimization Algorithm (Mirjalili & Lewis, 2016)	WOA	18	16	7.6649	17.1633
Moth-Flame Optimization (Mirjalili, 2015)	MFO	19	19	6.7454	12.9566
Sine Cosine Algorithm (Mirjalili, 2016a)	SCA	20	24	6.1045	11.3072
Henry Gas Solubility Optimization (Hashim et al., 2019)	HGSO	21	20	5.9538	12.6783
Earthworm Optimisation	EWA	22	21	5.6666	12.0153

(*continued on next page*)

Table 11 (*continued*)

Algorithms	Abbreviation	Rank Result		Total Score	
		CEC' 17	CEC' 14	CEC' 17	CEC' 14
Algorithm (Wang et al., 2018)					
BAT (Yang et al., 2018)	BAT	23	28	5.5275	10.3959
Tunicate Swarm Algorithm (Kaur et al., 2020)	TSA	24	25	5.3843	11.0589
Rat Swarm Optimizer (Dhiman et al., 2020)	RSO	25	22	5.0345	11.5681
Elephant Herding Optimization (Wang, Deb, & Coelho, 2016)	EHO	26	23	4.9281	11.3875
Butterfly Optimization Algorithm (Arora & Singh, 2019)	BOA	27	26	4.6217	10.9592
Dragonfly Algorithm (Mirjalili, 2016b)	DA	28	30	4.3053	8.7935
Spotted Hyena Optimizer (Dhiman & Kumar, 2017)	SHO	29	27	4.1472	10.9173
Sooty Tern Optimization Algorithm (Dhiman & Kaur, 2019)	STOA	30	29	3.9517	10.2519
Genetic Algorithm (Whitley, 1994)	GA	31	31	3.8155	8.3520
Differential Evolution (Storn & Price, 1997)	DE	32	32	3.8096	7.7703

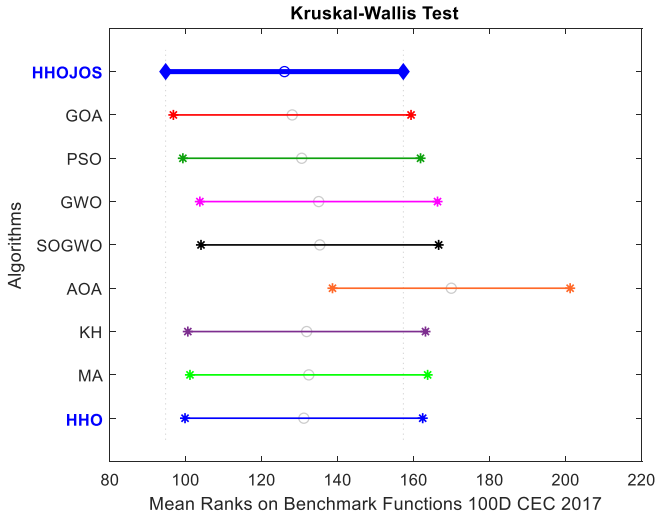


Fig. 11. Kruskal-Wallis test on HHO-JOS and its competitors (29 benchmark functions CEC 2017 in 100 dimensions)

functions of CEC 2017. Therefore, we can conclude that JOS helps HHO to achieve a competitive position with regard to the original version of HHO.

Table 12 presents the average of the best fitness values (*avg*) and standard deviation (*std*) of each algorithm. The lower the *avg* is, the more effective the algorithm be. The algorithms having the lowest *avg* in each benchmark function are highlighted in bold. The experimental results indicate that HHO-JOS is more effective than the other meta-heuristic algorithms in Table 12 for most of the benchmark functions, i.e., F1, F4, F9, F11, F12, F13, F14, F15, F18, F19, F25, and F28. When comparing to the original HHO, both HHO-JOS and HHO provide the same *avg* for the benchmark functions F25 and F28 at 3.5E + 03 and 3.6E + 03 respectively. In addition, for F25, GOA and KH also produce that same result as HHO-JOS and HHO. In case of F28, the effectiveness of KH and MA are also equivalent to HHO-JOS and HHO. However, there are cases that HHO-JOS is not the most effective algorithms as well. For example, considering PSO, the algorithm is slightly better than HHO-JOS in a few benchmark functions. Therefore, we can conclude that HHO-JOS is in a competitive position with regard to the original version of HHO and its competitors.

4.2.11. *Friedman Mean Rank evaluation of HHO-JOS and its competitors*

In this section, Friedman Mean Rank test was employed to verify the performance of HHO-JOS and to compare the results with nine optimization algorithms, i.e, the original version of HHO, GWO, SOGWO, the original version of DE, MTDE, j2020, MadDE, GSK, and AGSK. Friedman Mean Rank (Friedman, 1937) is a statistical test that evaluates groups of data by their ranks. Biswas et al. (2021) observed MadDE with three recent leading algorithms in CEC 2021 as follows: IMODE (Sallam et al., 2020), AGSK (Mohamed et al., 2020), and j2020 (Brest et al., 2020). Those algorithms started with a large number of population, and they took huge memory to deploy. However, in our experiment, the population size is fixed at $N = 30$. We found that IMODE encountered difficulty performing in this setting. Therefore, we did not include IMODE in the comparison. In this research, we evaluated the performance of HHO-JOS and compared the results with MadDE, AGSK, and j2020 algorithms by using the Friedman Mean Rank test on 10, 30, 50, and 100 dimensions of CEC 2017. The evaluation results presented in Table 13 are classified into two groups: Swarm Intelligence (SI) and Differential Evolution (DE). Those algorithms are sorted based on its groups by alphabetical order. The results indicated that the algorithms in SI, i.e, GWO, HHO, HHO-JOS and SOGWO, always came after the improved DE algorithms, i.e., AGSK, GSK, j2020, MadDE, and MTDE, when testing on 10, 30, 50, and 100 dimensions of CEC 2017. The trend of those algorithms also are presented by the bar charts in Fig. 12 and Fig. 13.

According to Table 13, the major reasons why the four algorithms in SI have lower ranks than the improved DE algorithms are the improved DE algorithm and its variations succeed in accelerating the operator of the original version of DE (Sallam et al., 2020) and diverting the population dynamically within the phase of mutation, crossover, and selection (Nadimi-Shahraki et al., 2020; Tanabe & Fukunaga, 2014). Although in the experiments we limited the number of population at $N = 30$, the improved DE algorithms still achieved higher ranks. For 10 and 30 dimensions, AGSK and GSK reached the top ranks respectively. Although the position of AGSK on 10 dimensions was on the top rank, in case of 30, 50 and 100 dimensions, AGSK consistently stayed at the 5th rank after MTDE, j2020, MadDE, and GSK. MTDE also experienced the same situation as AGSK.When tested on 10 and 30 dimensions, MTDE reached the second position, but it can only achieved the 4th rank on 50 and 100 dimensions. We also noticed that the results obtained from MTDE and j2020 always went to the different directions. j2020 won the first rank on 50 and 100 dimensions, but it was placed on the third position on 10 and 30 dimensions. In case of the algorithms in the SI group, the positions of GWO and SOGWO are the 8th and 7th respectively on 10 and 30 dimensions. The positions of GWO and SOGWO were switched when tested on 50 and 100 dimensions. Finally, the proposed HHO-JOS was obviously not on the top position, however, it always achieved the better performance than the original HHO and was able to compete with other leading swarm-based algorithms.

Fig. 12 and Fig. 13 visualize the experimental results in Table 13. The dashed line bar presents the Friedman mean rank of each optimization algorithm, and the solid bar describes the rank of each optimization algorithm. The lower the bar is, the better performance the algorithm has achieved. Overall, the trends of the experiment results on 10, 30, 50, and 100 dimensions CEC 2017 are similar to CEC 2014, as shown in Fig. 12 and Fig. 13. From the experimental results, we can confirm that SOGWO (which is an improved version of GWO) and AGSK (which is an

Table 12
The effectiveness of HHO-JOS and the competitive metaheuristic optimization algorithms (tested on 100 dimensions, CEC 2017).

F	HHO-JOS		GOA		PSO		GWO		SOGWO		AOA		KH		MA		HHO	
	Avg	Std	Avg	Std	Avg	Std	Avg	Std	Avg	Std	Avg	Std	Avg	Std	Avg	Std	Avg	Std
F1	**2.5E + 05**	9.8E + 04	2.1E + 09	3.4E + 09	4.4E + 10	1.9E + 10	4.8E + 10	9.7E + 09	4.7E + 10	8.9E + 09	2.6E + 11	1.3E + 10	9.9E + 08	1.5E + 09	7.3E + 08	3.0E + 08	2.2E + 08	2.3E + 07
F3	5.8E + 04	1.1E + 04	1.2E + 05	6.2E + 04	1.1E + 05	5.4E + 04	2.0E + 05	2.3E + 04	2.1E + 05	2.7E + 04	3.2E + 05	1.1E + 04	3.3E + 05	7.2E + 04	6.0E + 05	7.1E + 04	**3.8E + 04**	9.5E + 03
F4	**8.1E + 02**	5.5E + 01	9.8E + 02	3.1E + 02	5.5E + 03	3.3E + 03	4.5E + 03	1.2E + 03	4.3E + 03	8.7E + 02	9.5E + 03	1.3E + 04	9.0E + 02	1.0E + 02	9.5E + 02	1.0E + 02	8.4E + 02	6.3E + 01
F5	1.3E + 03	5.9E + 01	1.2E + 03	1.2E + 02	**9.9E + 02**	6.0E + 01	1.1E + 03	5.9E + 01	1.1E + 03	7.1E + 01	1.8E + 03	1.1E + 02	1.3E + 03	7.2E + 01	1.7E + 03	2.1E + 02	1.4E + 03	4.7E + 01
F6	6.4E + 02	5.5E + 00	6.6E + 02	7.5E + 00	**6.2E + 02**	5.1E + 00	6.4E + 02	5.0E + 00	6.4E + 02	4.7E + 00	6.9E + 02	7.7E + 00	6.6E + 02	4.8E + 00	6.9E + 02	1.7E + 01	6.8E + 02	3.0E + 00
F7	2.7E + 03	1.7E + 02	**1.6E + 03**	1.5E + 02	**1.6E + 03**	3.0E + 02	2.0E + 03	1.4E + 02	1.9E + 03	1.2E + 02	3.9E + 03	1.5E + 02	2.6E + 03	1.8E + 02	2.7E + 03	7.6E + 02	3.6E + 03	8.7E + 01
F8	1.7E + 03	6.2E + 01	1.5E + 03	9.3E + 01	**1.3E + 03**	7.3E + 01	1.4E + 03	7.2E + 01	1.4E + 03	5.8E + 01	2.3E + 03	9.8E + 01	1.7E + 03	6.7E + 01	2.1E + 03	2.0E + 02	1.8E + 03	6.8E + 01
F9	**2.3E + 04**	1.1E + 03	2.9E + 04	5.8E + 03	2.5E + 04	1.6E + 04	**2.3E + 04**	9.0E + 03	2.7E + 04	1.1E + 04	6.9E + 04	4.5E + 03	2.4E + 04	2.0E + 03	6.0E + 04	2.1E + 04	3.2E + 04	3.0E + 03
F10	1.6E + 04	1.5E + 03	1.6E + 04	1.5E + 03	**1.5E + 04**	1.4E + 03	**1.5E + 04**	2.5E + 03	**1.5E + 04**	2.6E + 03	2.9E + 04	9.6E + 02	1.6E + 04	1.2E + 03	2.3E + 04	4.2E + 03	1.8E + 04	1.3E + 03
F11	**2.7E + 03**	2.0E + 02	8.8E + 03	6.9E + 02	1.8E + 04	1.8E + 03	4.5E + 04	1.1E + 04	4.6E + 04	1.4E + 04	1.5E + 05	1.8E + 04	5.1E + 04	1.9E + 04	3.9E + 04	1.1E + 04	3.0E + 04	2.6E + 03
F12	**1.3E + 08**	6.3E + 07	1.1E + 09	1.3E + 09	1.5E + 10	1.2E + 10	9.5E + 09	6.1E + 09	9.6E + 09	5.6E + 09	1.9E + 11	1.9E + 10	9.8E + 07	5.6E + 07	5.2E + 08	2.0E + 08	2.8E + 08	8.2E + 07
F13	**4.8E + 04**	1.5E + 04	8.2E + 07	1.7E + 08	1.5E + 09	1.9E + 09	8.4E + 08	9.4E + 08	9.5E + 08	9.6E + 08	4.6E + 09	4.5E + 08	7.8E + 07	1.4E + 07	2.4E + 08	3.5E + 08	4.1E + 08	7.2E + 07
F14	**7.0E + 05**	2.3E + 05	9.2E + 05	9.2E + 05	2.1E + 06	3.1E + 06	4.6E + 06	3.9E + 06	3.7E + 06	2.2E + 06	4.2E + 07	1.9E + 07	3.3E + 06	1.3E + 06	1.4E + 06	6.4E + 05	8.1E + 05	3.0E + 05
F15	**2.7E + 04**	9.6E + 03	1.7E + 07	1.2E + 08	5.0E + 08	8.7E + 08	2.6E + 08	4.9E + 08	1.5E + 08	3.0E + 08	2.3E + 09	3.6E + 08	3.1E + 07	1.3E + 07	3.7E + 08	1.5E + 08	1.1E + 08	6.8E + 07
F16	6.4E + 03	8.3E + 02	5.7E + 03	5.7E + 02	6.2E + 03	8.0E + 02	5.9E + 03	6.6E + 02	**5.6E + 03**	5.4E + 02	2.1E + 04	2.7E + 03	6.7E + 03	4.3E + 02	6.5E + 03	8.0E + 02	7.0E + 03	7.3E + 02
F17	5.9E + 03	6.0E + 02	5.4E + 03	4.8E + 02	6.5E + 03	1.5E + 03	4.9E + 03	8.6E + 02	**4.7E + 03**	6.1E + 02	5.1E + 04	3.5E + 03	5.6E + 03	6.9E + 02	6.3E + 03	6.8E + 02	6.1E + 03	6.8E + 02
F18	**7.4E + 05**	2.6E + 05	1.5E + 06	1.5E + 06	2.2E + 06	2.0E + 06	3.2E + 06	2.0E + 06	4.3E + 06	2.9E + 06	7.0E + 06	3.2E + 06	3.5E + 06	1.9E + 06	4.3E + 06	2.1E + 06	1.8E + 06	6.5E + 05
F19	**1.5E + 05**	2.1E + 04	4.2E + 07	1.3E + 08	4.3E + 08	7.5E + 08	2.2E + 08	2.9E + 08	1.2E + 08	1.7E + 08	2.4E + 09	4.3E + 08	3.9E + 07	1.0E + 07	1.5E + 08	5.9E + 08	4.0E + 08	1.5E + 08
F20	5.5E + 03	5.1E + 02	5.0E + 03	5.5E + 02	4.8E + 03	4.6E + 02	**4.4E + 03**	4.0E + 02	4.5E + 03	7.0E + 02	7.1E + 03	3.7E + 02	5.5E + 03	5.5E + 02	5.9E + 03	7.3E + 02	5.8E + 03	5.1E + 02
F21	3.4E + 03	1.8E + 02	3.0E + 03	1.2E + 02	3.1E + 03	1.3E + 02	**2.9E + 03**	7.9E + 01	2.9E + 03	6.8E + 01	4.3E + 03	1.6E + 02	3.6E + 03	1.3E + 02	3.7E + 03	2.2E + 02	3.8E + 03	1.5E + 02
F22	2.0E + 04	1.3E + 03	1.8E + 04	1.2E + 03	**1.7E + 04**	1.6E + 03	1.8E + 04	3.7E + 03	**1.7E + 04**	1.5E + 03	3.3E + 04	1.2E + 03	2.1E + 04	1.6E + 03	2.3E + 04	2.7E + 03	2.2E + 04	1.1E + 03
F23	4.1E + 03	3.3E + 02	3.6E + 03	1.2E + 02	4.7E + 03	3.1E + 02	**3.5E + 03**	9.5E + 01	**3.5E + 03**	9.3E + 01	6.8E + 03	5.3E + 02	5.1E + 03	3.3E + 02	4.2E + 03	2.1E + 02	4.7E + 03	3.0E + 02
F24	5.2E + 03	3.9E + 02	**4.0E + 03**	1.1E + 02	6.2E + 03	4.3E + 02	4.2E + 03	1.4E + 02	4.1E + 03	1.4E + 02	1.1E + 04	2.0E + 02	5.9E + 03	4.6E + 02	5.0E + 03	2.7E + 02	5.6E + 03	3.1E + 02
F25	**3.5E + 03**	5.6E + 02	**3.5E + 03**	1.2E + 02	5.2E + 03	1.2E + 02	6.4E + 03	9.1E + 02	6.4E + 03	8.9E + 02	2.8E + 04	2.4E + 03	**3.5E + 03**	9.9E + 02	3.6E + 03	1.1E + 02	**3.5E + 03**	9.0E + 02
F26	2.2E + 03	3.9E + 02	**1.3E + 03**	2.0E + 02	2.0E + 03	3.9E + 02	1.4E + 03	1.1E + 02	1.5E + 03	1.2E + 02	5.3E + 03	2.8E + 02	2.6E + 03	4.4E + 02	2.4E + 03	3.2E + 02	2.4E + 03	2.0E + 02
F27	4.0E + 02	1.9E + 01	**3.5E + 02**	8.0E + 01	4.3E + 02	4.5E + 01	4.0E + 02	1.6E + 01	4.0E + 02	1.3E + 01	9.5E + 02	4.8E + 01	6.1E + 02	9.7E + 01	3.9E + 02	1.7E + 01	4.1E + 02	2.6E + 01
F28	**3.6E + 03**	4.9E + 02	4.0E + 03	9.9E + 02	9.0E + 03	2.3E + 03	8.2E + 03	1.3E + 03	8.1E + 03	1.1E + 03	3.7E + 04	5.2E + 03	**3.6E + 03**	1.4E + 03	**3.6E + 03**	4.7E + 03	**3.6E + 03**	4.9E + 03
F29	7.9E + 03	5.9E + 02	8.4E + 03	6.4E + 02	**7.5E + 03**	6.6E + 02	7.9E + 03	6.5E + 02	8.0E + 03	9.1E + 02	3.1E + 04	1.9E + 03	8.6E + 03	5.2E + 02	8.0E + 03	9.6E + 02	8.7E + 03	6.7E + 02
F30	6.5E + 06	4.2E + 06	1.3E + 08	2.4E + 08	1.6E + 09	1.4E + 09	7.2E + 08	8.2E + 08	1.1E + 09	1.2E + 09	4.0E + 10	4.8E + 09	9.9E + 06	6.3E + 06	**5.9E + 05**	2.1E + 06	2.1E + 07	5.6E + 06

improved version of GSK) did not perform well in some dimensions comparing to their original versions. However, the HHO-JOS consistently outperformed the original HHO in CEC 2014 and CEC 2017 in all dimensions. Moreover, we also conducted further analysis on the algorithms in both SI and DE groups by using mean and standard deviation. The analyzed results are presented in Table 14 and Table 15. The best mean value of each benchmark function is highlighted in bold. When more than one algorithms produced the same result, italic bold is used to highlight.

Table 14 reveals that HHO-JOS is still in a better position than its competitors in most of the CEC 2017's benchmark functions. HHO-JOS achieved the lowest mean values in 14 benchmark functions (F1, F3, F4, F11, F12, F13, F14, F15, F18, F19, F25, F28, F29, and F30). For benchmark functions F1, F13, F15, F18, F19 and F30, HHO-JOS showed a significant difference in the mean values comparing to the other algorithms in the same group. Considering F5, HHO-JOS tided with GWO and SOGWO. In Table 15, among the variations of DE algorithms, GSK is the leader in 10 benchmark functions of CEC 2017 (F5, F7, F8, F11, F16, F17, F18, F21, F24, and F29). MadDE, j2020, AGSK and MTDE won 8, 6, 3 and 2 times respectively. Although MadDE is a leading algorithm when competing in CEC 2021, the numbers of winning regarding the best mean value of MadDE in CEC 2017 is lower than GSK; 8 vs. 10. Moreover, AGSK, which is an improved version of GSK, did not perform well in CEC 2017 comparing to the original version. Nevertheless, the

Table 13
Friedman Mean Rank in 10D, 30D, 50D, and 100D CEC 2017.

MC	Algorithm	10D Friedman	10D Rank	30D Friedman	30D Rank	50D Friedman	50 Rank	100D Friedman	100D Rank
SI	GWO	7.35	8	7.25	8	6.85	6	6.72	6.5
	HHO	8.15	9	7.85	9	7.68	9	7.68	9
	HHO-JOS	6.78	6	6.95	6	6.98	8	6.72	6.5
	SOGWO	7.12	7	7.22	7	6.92	7	6.92	8
DE	AGSK	2.52	1	3.68	5	4.25	5	4.65	5
	DE	9.85	10	9.85	10	9.85	10	9.85	10
	GSK	3.73	5	2.48	1	3.15	2	3.15	2
	j2020	3.08	3	3.25	3	2.62	1	2.42	1
	MadDE	3.43	4	3.42	4	3.22	3	3.42	3
	MTDE	2.98	2	3.05	2	3.48	4	3.48	4

MC = Metaheuristic Clasification; SI = Swarm Intelligence; DE = Differential Evolution

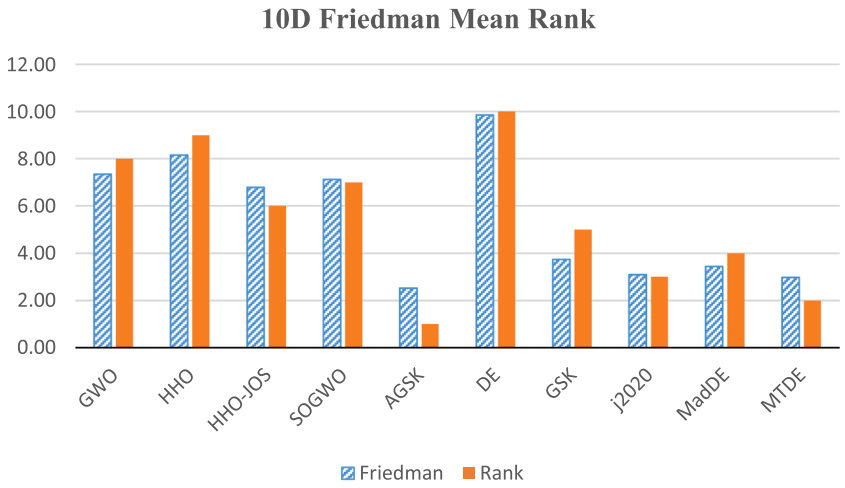


Fig. 12. A comparison of Friedman Mean Ranks tested on HHO-JOS and its competitors (29 benchmark functions CEC 2017 in 10 dimensions).

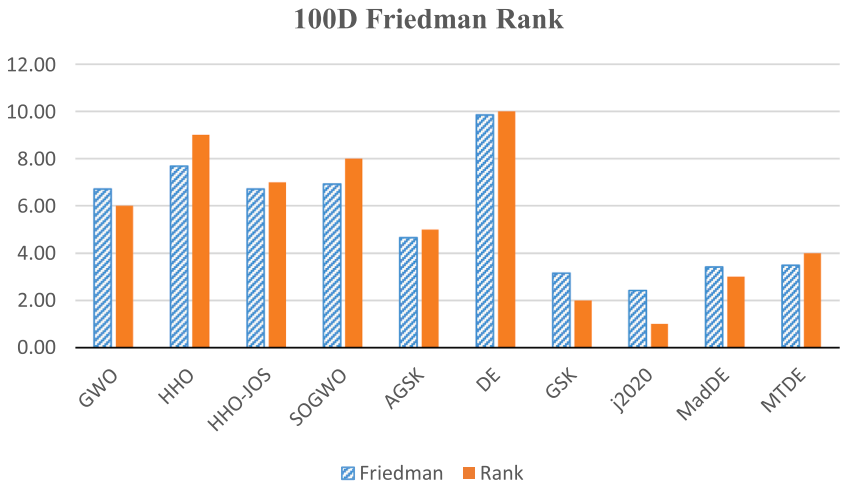


Fig. 13. A comparison of Friedman Mean Ranks tested on HHO-JOS and its competitors (29 benchmark functions CEC 2017 in 100 dimensions).

numbers of winning benchmark functions of AGSK are slightly higher than MTDE (3 vs. 2).

4.2.12. Convergence curves of the proposed HHO-JOS with the Jr at 0.25 and its competitors, tested on 100D CEC 2017

The convergence curves in Fig. 14 illustrate the strength of HHO-JOS compared with the original HHO and eight state-of-the-art optimization algorithms presented in Table 16, i.e., MA, MTDE, SMA, GOA, SLPSO, GWO, CMA-ES, and KH. Table 16 sorts the eight nature-inspired optimization algorithms in historical order, starting from the year 2012–2020. Fig. 12(i) shows the symbolics used to represent HHO-JOS, the original HHO, and the eight competitors. The convergence curves exhibit on 100 dimensions CEC 2017. The selected benchmark functions are F1, F5, F6, F7, F8, F10, F20, F22, and F28. The x −axis on the convergence curve presents the number of function evaluations (*nFE*), and the y −axis defines the best fitness of each algorithm. The numbers on x −axis are arranged based on the maximum function evaluations (*maxFE*), which is influenced by the number of dimensions. However, the numbers on y −axis are adjusted according to the benchmark function. The HHO-JOS marked in bold with a diamond marker and the

Table 14
Mean and Standard Deviation of HHO-JOS and swarm-based optimization algorithms (tested on 100 dimensions, CEC 2017).

F	HHO		HHO-JOS		GWO		SOGWO	
	Mean	Std	Mean	Std	Mean	Std	Mean	Std
F1	2.18E + 08	2.33E + 07	**2.46E + 05**	9.76E + 04	4.77E + 10	9.70E + 09	4.67E + 10	8.94E + 09
F3	3.76E + 04	9.50E + 03	**5.79E + 04**	1.07E + 04	2.04E + 05	2.32E + 04	2.09E + 05	2.65E + 04
F4	8.37E + 02	6.32E + 01	**8.06E + 02**	5.47E + 01	4.48E + 03	1.24E + 03	4.31E + 03	8.68E + 02
F5	1.42E + 03	4.75E + 01	***1.31E + 03***	5.90E + 01	***1.13E + 03***	5.86E + 01	***1.13E + 03***	7.07E + 01
F6	6.79E + 02	3.03E + 00	6.36E + 02	5.46E + 00	***6.35E + 02***	5.03E + 00	***6.35E + 02***	4.67E + 00
F7	3.55E + 03	8.72E + 01	2.69E + 03	1.66E + 02	1.95E + 03	1.41E + 02	**1.93E + 03**	1.17E + 02
F8	1.85E + 03	6.84E + 01	1.69E + 03	6.18E + 01	**1.43E + 03**	7.23E + 01	1.44E + 03	5.80E + 01
F9	3.16E + 04	3.01E + 03	2.32E + 04	1.09E + 03	**2.26E + 04**	9.00E + 03	2.71E + 04	1.15E + 04
F10	1.85E + 04	1.29E + 03	1.58E + 04	1.46E + 03	**1.51E + 04**	2.48E + 03	1.52E + 04	2.61E + 03
F11	2.97E + 03	2.62E + 02	**2.65E + 03**	2.03E + 02	4.55E + 04	1.10E + 04	4.57E + 04	1.36E + 04
F12	2.79E + 08	8.22E + 07	**1.27E + 08**	6.33E + 07	9.49E + 09	6.07E + 09	9.56E + 09	5.58E + 09
F13	4.05E + 06	7.23E + 05	**4.79E + 04**	1.52E + 04	8.39E + 08	9.42E + 08	9.50E + 08	9.63E + 08
F14	8.10E + 05	3.03E + 05	**6.98E + 05**	2.35E + 05	4.61E + 06	3.87E + 06	3.68E + 06	2.22E + 06
F15	1.14E + 06	6.77E + 05	**2.69E + 04**	9.60E + 03	2.58E + 08	4.86E + 08	1.52E + 08	3.01E + 08
F16	7.04E + 03	7.28E + 02	6.36E + 03	8.26E + 02	5.92E + 03	6.57E + 02	**5.64E + 03**	5.37E + 02
F17	6.10E + 03	6.77E + 02	5.91E + 03	5.96E + 02	4.91E + 03	8.63E + 02	**4.71E + 03**	6.10E + 02
F18	1.79E + 06	6.50E + 05	**7.44E + 05**	2.64E + 05	3.25E + 06	1.96E + 06	4.30E + 06	2.94E + 06
F19	4.03E + 06	1.46E + 06	**1.46E + 05**	2.07E + 05	2.15E + 08	2.95E + 08	1.19E + 08	1.69E + 08
F20	5.84E + 03	5.05E + 02	5.49E + 03	5.10E + 02	**4.39E + 03**	3.99E + 02	4.50E + 03	7.03E + 02
F21	3.80E + 03	1.53E + 02	3.37E + 03	1.83E + 02	***2.93E + 03***	7.90E + 01	***2.93E + 03***	6.77E + 01
F22	2.24E + 04	1.08E + 03	1.99E + 04	1.31E + 03	1.83E + 04	3.74E + 03	**1.73E + 04**	1.49E + 03
F23	4.74E + 03	3.03E + 02	4.12E + 03	3.26E + 02	3.53E + 03	9.50E + 01	**3.52E + 03**	9.25E + 01
F24	5.55E + 03	3.11E + 02	5.24E + 03	3.85E + 02	4.17E + 03	1.36E + 02	**4.13E + 03**	1.42E + 02
F25	3.50E + 03	8.96E + 01	**3.47E + 03**	5.59E + 01	6.41E + 03	9.12E + 02	6.43E + 03	8.95E + 02
F26	2.39E + 04	1.96E + 03	2.22E + 04	3.87E + 03	**1.42E + 04**	1.14E + 03	1.48E + 04	1.18E + 03
F27	4.10E + 03	2.58E + 02	3.98E + 03	1.92E + 02	**3.97E + 03**	1.59E + 02	4.01E + 03	1.31E + 02
F28	3.56E + 03	4.88E + 01	**3.57E + 03**	4.88E + 01	8.20E + 03	1.29E + 03	8.07E + 03	1.07E + 03
F29	8.70E + 03	6.65E + 02	**7.87E + 03**	5.88E + 02	7.91E + 03	6.49E + 02	7.95E + 03	9.09E + 02
F30	2.06E + 07	5.57E + 06	**6.46E + 06**	4.21E + 06	7.20E + 08	8.23E + 08	1.05E + 09	1.15E + 09

original HHO marked in bold with the arrow marker pointing to the right showing the gap difference of JOS improvement on HHO and the original HHO.

There are several trends on the convergence curve in Fig. 14. On most benchmark functions, SMA almost loses in the chosen benchmark functions. Surprisingly, on benchmark functions F10 and F22, the convergence curves of MTDE, CMA-ES and GWO have the mean values of best fitness that are close to SMA. In other words, in average, the best fitness values of SMA, MTDE, CMA-ES, and GWO do not converge well in F10 and F22. Furthermore, on benchmark functions F6 and F20, MTDE experiences a sharp slope. The winning algorithm on most benchmark functions is SLPSO. Although the average of best fitness values of HHO-JOS is not on the best performance, the strong point is the proposed HHO-JOS is able to compete against the original version of HHO. The gap difference of HHO-JOS confirms its competitiveness. Even though the benchmark function F7 shows a small gap difference, the benchmark functions F1, F5, F6, F8, F20, and F28 indicate greater improvement. In addition, the major improvement is on F10 and F22. In conclusion, HHO-JOS ecstatically made a promising development.

4.3. Computational complexity

Heidari et al. (2019) concluded that the main three processes of HHO are initialization, fitness evaluation, and updating of hawks.

The computational complexity of the HHO is defined as:

$$O(N \times (T + T \times D + 1))$$

where N is the number of hawks, $T$is the maximum number of iterations and D is the dimension of specific problems.

The computational complexity of JOS (SLO and DO) can be summarized as follows:

$O(\text{SLO}) = O(N \times T \times D_c)$ where D_c is the close dimension

$O(\text{DO}) = O(N \times Jr \times T \times D)$ where Jr is jumping rate

Therefore, the computational time for HHO-JOS is:

$O(\text{HHO-JOS})$

$= O(N \times (T + T \times D + 1) + N \times T \times D_c + N \times Jr \times T \times D) = O(N \times T(2 + D_c + D(T + Jr))$

Although the HHO-JOS and HHO have the same order, the complexity of the HHO-JOS sufficiently influences by the dimensionally change in the close distance (D_c) and jumping rate (Jr).

5. Discussion on the proposed JOS strategy and knowledge management (KM)

There are some connections between JOS and KM. The personalized and codification strategies in KM (Hansen et al., 2003) are complied with our proposed joint opposition learning technique (JOS) and HHO-JOS. The goals of the personalized and codification strategy recognize in KM are through person to person and person to documentation (a large number of members), successively. The benefit of the personalized strategy is the ability to solve high-level strategic problems by recognizing individual characteristic expertise. The impact of codification strategy is producing a fast, high-quality, and reliable information system. Today, those terms still exist in Attar et al. (2018), Bashir and Farooq (2019), Kianto et al. (2019), Ouriques et al. (2019), and Korimbocus et al. (2020). However, the way those strategies are delivered has been changed.

Interestingly, both personalized and codification strategies in KM naturally align with the proposed JOS, which combines SLO and DO. The SLO moves when the difference distance of the dimension on the current and the best position of the hawks are less than the threshold; this action is related to the term of personalizing strategy in KM, which represents person-to-person. Then, when DO is applied to all hawks, the codification strategy in KM wields its source for all members. Therefore, it is confirmed that the JOS lends itself well in knowledge management.

6. Conclusions

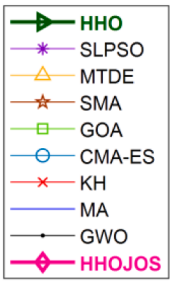
The Opposition Based Learning (OBL) technique offers an opportunity to enhance the performance of an algorithm. Many experiments

Table 15
Mean and Standard Deviation of Differential Evolution and its variations (tested on 100 dimensions, CEC 2017).

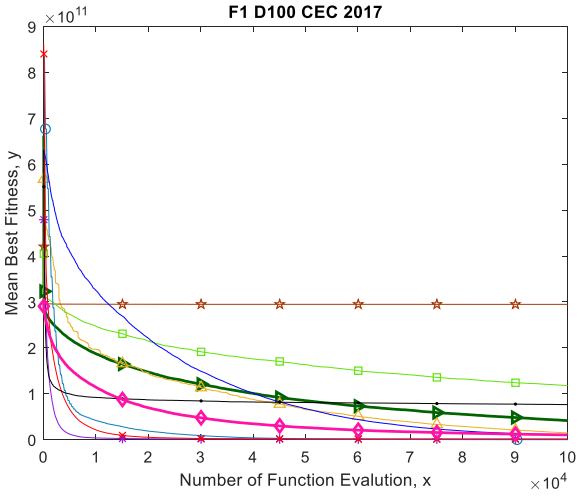
F	DE		MTDE		j2020		MadDE		GSK		AGSK	
	Mean	Std	Mean	Std	Mean	Std	Mean	Std	Mean	Std	Mean	Std
F1	6.57E + 11	2.89E + 10	4.47E + 03	5.49E + 03	**1.00E + 02**	5.23E-04	1.01E + 02	1.98E + 00	4.97E + 03	5.64E + 03	3.52E + 03	3.41E + 03
F3	2.16E + 17	1.76E + 17	3.01E + 02	4.15E + 00	**1.39E + 04**	5.62E + 03	2.37E + 05	6.42E + 04	7.81E + 03	3.62E + 03	5.70E + 04	1.48E + 04
F4	3.95E + 05	4.90E + 04	6.30E + 02	2.70E + 01	6.37E + 02	4.68E + 01	**4.87E + 02**	5.86E + 01	5.99E + 02	5.23E + 01	5.95E + 02	5.01E + 01
F5	3.28E + 03	8.68E + 01	8.82E + 02	4.27E + 01	8.58E + 02	6.74E + 01	1.14E + 03	4.67E + 01	**7.72E + 02**	4.99E + 01	1.45E + 03	7.37E + 01
F6	7.71E + 02	5.67E + 00	6.19E + 02	1.89E + 00	6.04E + 02	1.72E + 00	6.04E + 02	3.71E + 00	6.09E + 02	2.93E + 00	**6.00E + 02**	4.41E-03
F7	1.44E + 04	5.61E + 02	1.65E + 03	9.20E + 01	1.32E + 03	1.22E + 02	2.10E + 03	1.50E + 02	**1.19E + 03**	8.49E + 01	1.72E + 03	7.19E + 01
F8	4.05E + 03	1.13E + 02	1.15E + 03	2.57E + 01	1.15E + 03	4.74E + 01	1.46E + 03	4.74E + 01	**1.07E + 03**	5.07E + 01	1.77E + 03	7.45E + 01
F9	2.14E + 05	1.31E + 04	7.05E + 03	1.19E + 03	**3.48E + 03**	1.44E + 03	1.80E + 04	1.14E + 03	3.90E + 03	2.91E + 03	4.52E + 03	1.67E + 03
F10	3.68E + 04	1.25E + 03	1.49E + 04	6.61E + 02	1.38E + 04	1.38E + 03	**1.18E + 04**	9.10E + 02	3.01E + 04	6.11E + 02	2.41E + 04	7.19E + 02
F11	8.24E + 07	4.81E + 08	1.76E + 03	9.07E + 01	1.79E + 03	1.60E + 02	4.80E + 03	2.21E + 03	**1.44E + 03**	8.37E + 01	2.10E + 03	1.25E + 02
F12	4.38E + 11	2.63E + 10	1.22E + 06	3.05E + 05	4.67E + 05	2.03E + 05	**9.02E + 04**	3.41E + 04	5.69E + 05	2.10E + 05	2.72E + 05	1.17E + 05
F13	1.11E + 11	7.23E + 09	1.09E + 04	5.52E + 03	5.26E + 03	3.02E + 03	**4.17E + 03**	1.92E + 03	5.67E + 03	3.93E + 03	6.46E + 03	4.96E + 03
F14	2.21E + 09	1.02E + 09	4.68E + 04	2.42E + 04	**1.71E + 03**	6.47E + 01	3.60E + 04	2.79E + 04	7.06E + 03	4.78E + 03	4.48E + 04	5.75E + 04
F15	8.25E + 10	8.24E + 09	5.38E + 03	2.78E + 03	3.97E + 03	3.27E + 03	**2.69E + 03**	1.38E + 03	3.68E + 03	2.91E + 03	2.80E + 03	1.12E + 03
F16	5.47E + 04	7.39E + 03	4.41E + 03	5.52E + 02	4.73E + 03	3.69E + 02	4.68E + 03	4.09E + 02	**3.56E + 03**	5.53E + 02	5.89E + 03	2.29E + 02
F17	1.47E + 09	6.68E + 08	4.25E + 03	3.94E + 02	4.04E + 03	2.58E + 02	4.09E + 03	2.42E + 02	**3.61E + 03**	6.04E + 02	4.84E + 03	1.81E + 02
F18	1.12E + 09	3.38E + 08	8.08E + 04	3.47E + 04	3.10E + 04	1.30E + 04	2.26E + 05	1.82E + 05	**2.56E + 04**	2.19E + 04	3.84E + 05	1.98E + 05
F19	8.37E + 10	7.04E + 09	5.41E + 03	3.96E + 03	3.40E + 03	2.39E + 03	3.39E + 03	1.34E + 03	4.22E + 03	2.94E + 03	**2.66E + 03**	8.29E + 02
F20	1.02E + 04	4.03E + 02	**4.17E + 03**	5.95E + 02	4.20E + 03	2.78E + 02	4.30E + 03	3.59E + 02	4.76E + 03	1.33E + 03	4.98E + 03	2.03E + 02
F21	5.34E + 03	1.45E + 02	2.68E + 03	4.09E + 01	2.66E + 03	5.54E + 01	2.75E + 03	4.95E + 01	**2.59E + 03**	4.65E + 01	3.18E + 03	6.84E + 01
F22	3.94E + 04	8.74E + 02	1.68E + 04	9.58E + 02	1.63E + 04	1.21E + 03	**1.52E + 04**	9.78E + 02	3.15E + 04	3.70E + 03	2.62E + 04	9.05E + 02
F23	9.97E + 03	4.65E + 02	3.21E + 03	3.30E + 01	**3.07E + 03**	5.27E + 01	3.15E + 03	5.17E + 01	3.12E + 03	5.76E + 01	3.51E + 03	2.45E + 01
F24	2.19E + 04	9.18E + 02	3.75E + 03	3.75E + 01	3.66E + 03	6.94E + 01	3.88E + 03	6.99E + 01	**3.64E + 03**	9.64E + 01	4.18E + 03	7.96E + 01
F25	1.72E + 05	1.78E + 04	**3.26E + 03**	6.38E + 01	3.30E + 03	7.00E + 01	3.28E + 03	5.85E + 01	3.30E + 03	6.27E + 01	3.31E + 03	5.32E + 01
F26	2.20E + 05	1.60E + 04	1.08E + 04	4.80E + 02	**9.88E + 03**	7.62E + 02	1.71E + 04	6.33E + 03	1.45E + 04	5.23E + 03	1.69E + 04	1.42E + 03
F27	2.38E + 04	1.22E + 03	3.69E + 03	6.42E + 01	3.45E + 03	4.34E + 01	3.54E + 03	5.79E + 01	3.68E + 03	1.14E + 02	**3.39E + 03**	5.30E + 01
F28	1.10E + 05	7.00E + 03	3.42E + 03	2.31E + 01	3.37E + 03	2.66E + 01	**3.34E + 03**	4.14E + 01	3.37E + 03	3.01E + 01	3.36E + 03	2.85E + 01
F29	5.71E + 07	2.77E + 07	6.03E + 03	7.75E + 02	5.56E + 03	3.52E + 02	6.05E + 03	3.55E + 02	**5.35E + 03**	4.90E + 02	6.72E + 03	2.24E + 02
F30	9.86E + 10	1.28E + 10	9.72E + 03	3.14E + 03	8.11E + 03	4.00E + 03	**7.82E + 03**	2.82E + 03	1.21E + 04	4.82E + 03	9.77E + 03	3.06E + 03

testify that the idea of OBL and the extended idea of OBL such as Quasi, Reflective, Dynamic Opposite (DO), Selective Opposition (SO), and Selective Leading Opposition (SLO) still have weaknesses based on the scoring metric results. Therefore, we proposed Joint Opposite Selection (JOS) which is a combination of the two opposition learning strategies: SLO and DO by using the Random Jump Strategy (RJS).

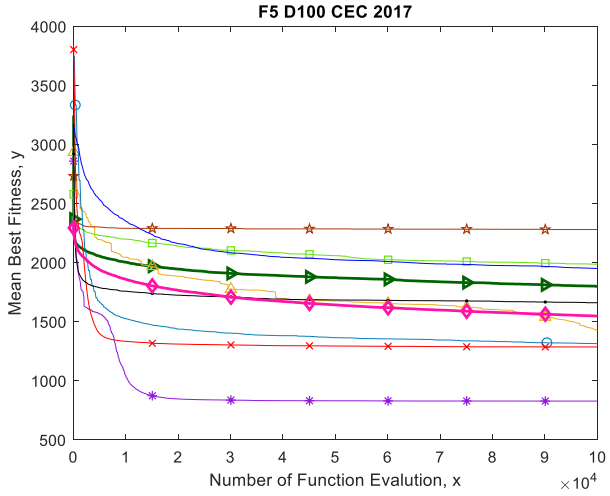
Note that the SLO is our invention strategy, which is inherited from SO. The original SO changes the dimensions on the faraway search agents, whereas the SLO changes the dimensions on the close search agents. Meanwhile, the DO is a merged idea of Quasi and Reflective Opposition. This joint opposition learning technique succeeds in balancing the exploration and exploitation phases. When SLO achieved in exploitation phase which might trapped in the local optima, the DO tries to overcome the drawback by enlarging its capability in the search space again to escape from the trap in the local optima. The proposed JOS is embedded in HHO (HHO-JOS). Harris' Hawks Optimization



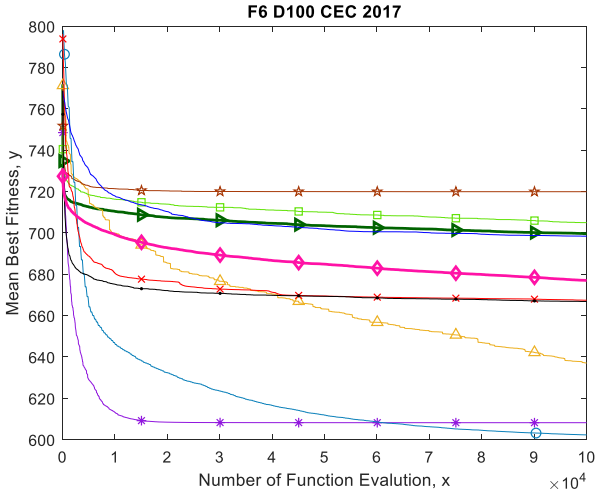
(i)



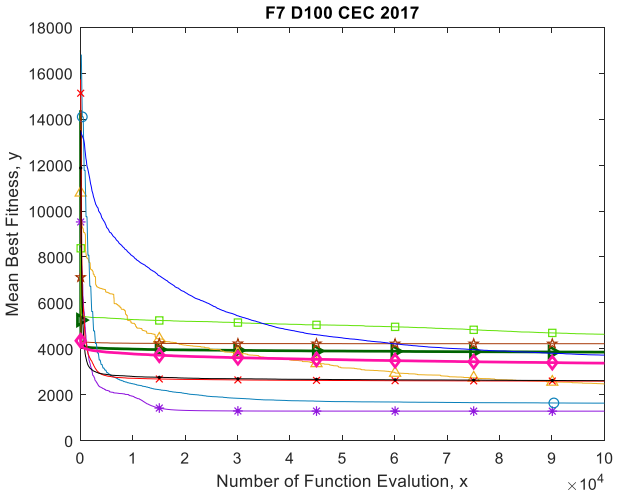
(ii)



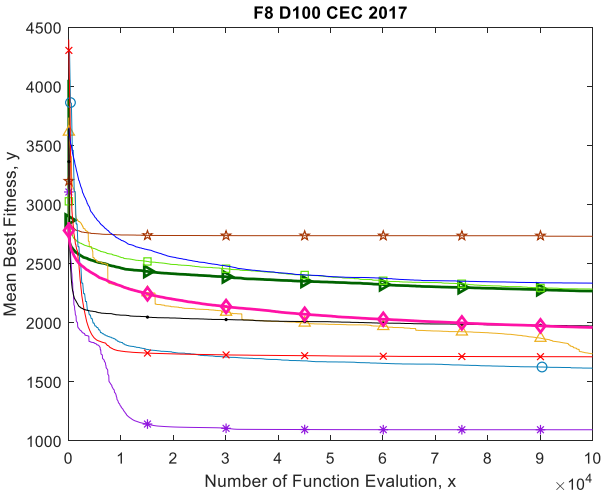
(iii)



(iv)



(v)



(vi)

Fig. 14. Convergence curves of the proposed HHO-JOS with the *Jr* at 0.25 and its competitors tested on 100D CEC 2017.

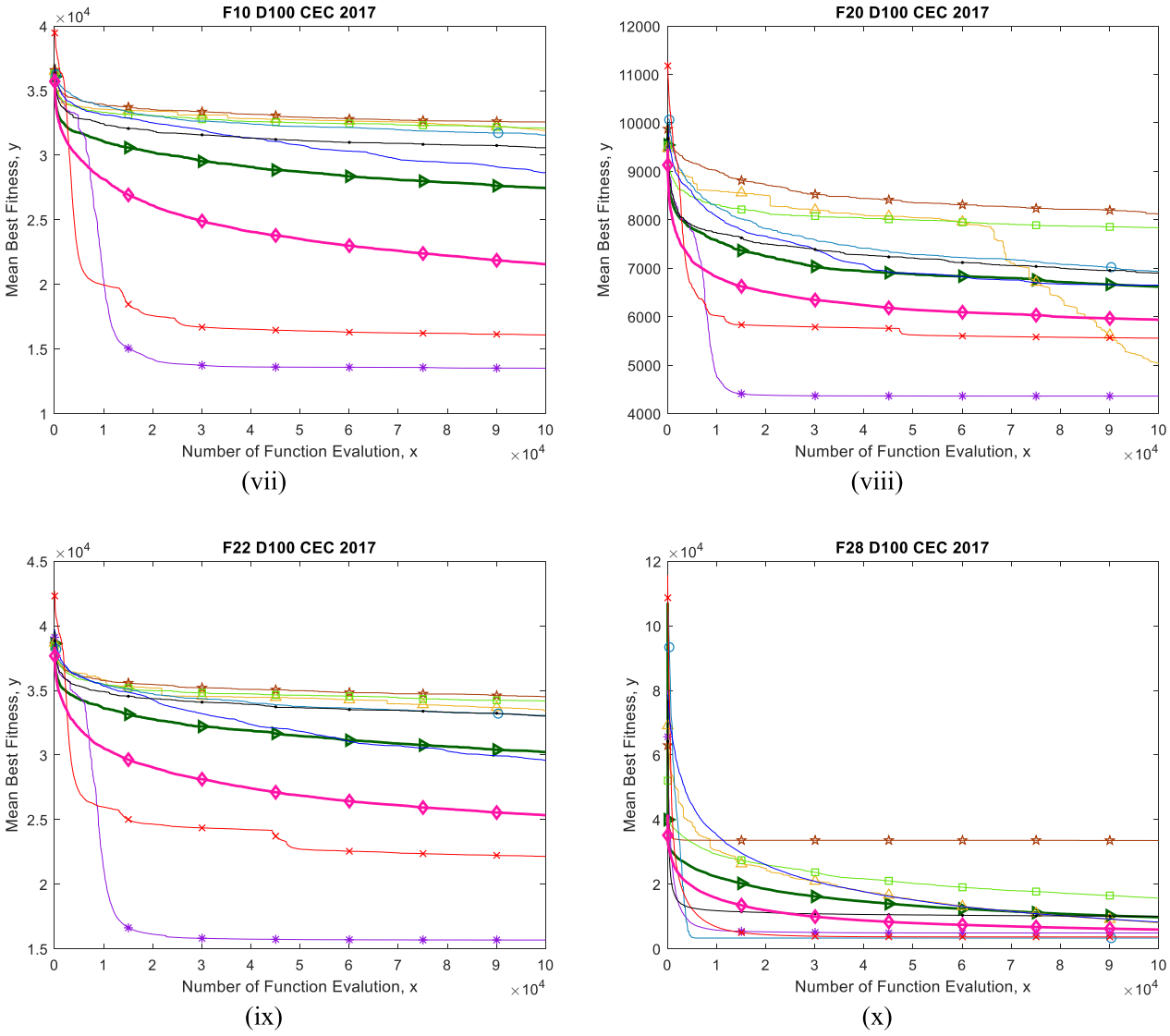


Fig. 14. (*continued*).

Table 16
Competitive nature-inspired optimization algorithms evaluated on convergence curves.

No	Algorithms	Abbreviation	Year
1	Mayfly Algorithm (Zervoudakis & Tsafarakis, 2020)	MA	2020
2	Multi-Trial vector-based Differential Evolution (Nadimi-Shahraki et al., 2020)	MTDE	2020
3	Slime Mould Algorithm (Li et al., 2020)	SMA	2020
4	Harris' Hawks Optimization (Heidari et al., 2019)	HHO	2019
5	Grasshopper Optimization Algorithm (Saremi et al., 2017)	GOA	2017
6	Social Learning Particle Swarm Optimization (Cheng & Jin, 2015)	SLPSO	2015
7	Grey Wolf Optimizer (S. Mirjalili et al., 2014)	GWO	2014
8	Covariance Matrix Adaption-Evolution Strategy (Hansen et al., 2003)	CMA-ES	2013
9	Krill Herd (Gandomi & Alavi, 2012)	KH	2012

(HHO) is a novel optimization algorithm in Swarm Intelligent that applied the hunting strategy of hawks called the "surprise pounce" strategy. The proposed JOS succeeds in upgrading the ability of HHO for higher single objective problems in CEC 2014 and CEC 2017.

From the experimental results, we conclude as follows:

(a) A sufficient adjustment of JOS can be obtained by setting a linear decrement threshold value on SLO and Random Jump Strategy with *Jr* at 0.25 on DO.
(b) JOS succeeds to diverse hawks' population by enlarging their exploration ability to escape the trap at the local optima.
(c) Wilcoxon signed-rank test shows major improvement of HHO-JOS comparing to its competitors (the original HHO, HHO-DO, and HHO-SLO) by winning in most benchmark functions and a greater difference on objective function values.
(d) Based on the scoring metric, HHO-JOS produces a promising result compared to its original version of HHO. Although HHO-JOS is not on the top rank, HHO-JOS still shows dominancy among other state-of-the-art algorithms.
(e) Kruskal-Wallis statistical analysis confirms the effectiveness of HHO-JOS when solving single-objective optimization problems.
(f) Friedman Mean Rank emphasized that HHO-JOS can compete with other leading optimization algorithms in CEC 2017, however, there are some of the leading algorithms in CEC 2021 that perform better than HHO-JOS.
(g) The convergent curves of HHO-JOS on most benchmark functions produce a major improvement on the average fitness value compared to the original HHO.

(h) Both SLO and DO in the proposed JOS strategy naturally conform to the strategies used in Knowledge Management: personalized and codification.

Therefore, according to the experimental results, the growth development of OBL influences the proposed JOS performance. The JOS will be greatly favorable for future techniques embedded in optimization algorithms and the real-world applications, especially for expert system applications. The authors also confirm the originality of this research. In the future work, (a) the proposed JOS can be used to improved other novel algorithms to find the best solution for engineering design problems, (b) HHO-JOS can be preferably optimization approach to enhance the quality of feature selection, and (c) HHO-JOS can be used to determine the localization accuracy for wireless sensor network purposes, and many more.

CRediT authorship contribution statement

Florentina Yuni Arini: Software, Data curation, Methodology, Formal analysis, Writing – original draft, Visualization. **Sirapat Chiewchanwattana:** Supervision, Conceptualization, Methodology, Investigation, Writing – review & editing. **Chitsutha Soomlek:** Supervision, Conceptualization, Methodology, Formal analysis, Writing – review & editing. **Khamron Sunat:** Conceptualization, Validation, Investigation, Formal analysis.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgment

This research was conducted at the Advanced Smart Computing Laboratory, Artificial Intelligence Research Center (AIRC), Khon Kaen University. The research is fully funded by KKU Scholarship for ASEAN and GMS Countries of Academic Year of 2019. The authors would like to thank the shared code and knowledge all the researchers.

References

Abd Elaziz, M., Oliva, D., & Xiong, S. (2017). An improved Opposition-Based Sine Cosine Algorithm for global optimization. *Expert Systems with Applications, 90*, 484–500. https://doi.org/10.1016/j.eswa.2017.07.043

Ahmed, B. S. (2016). Test case minimization approach using fault detection and combinatorial optimization techniques for configuration-aware structural testing. *Engineering Science and Technology, an International Journal, 19*(2), 737–753. https://doi.org/10.1016/j.jestch.2015.11.006

Arora, S., & Singh, S. (2019). Butterfly optimization algorithm: A novel approach for global optimization. *Soft Computing, 23*(3), 715–734. https://doi.org/10.1007/s00500-018-3102-4

Attar, M. M., Kang, K., & Sohaib, O. (2018). Organisational culture, knowledge sharing and intellectual capital: Directions for future research. Proceedings of the 31st International Business Information Management Association Conference, IBIMA 2018: Innovation Management and Education Excellence through Vision 2020, 852–857. 10.11648/j.ijber.20200901.12.

Awad, N. H., Ali, M. Z., Suganthan, P. N., Liang, J. J., & Qu, B. Y. (2016). CEC2017-BoundContrained/Definitions of CEC2017 benchmark suite final version updated. pdf at master · P-N-Suganthan/CEC2017-BoundContrained · GitHub. Technical Report 201311, Computational Intelligence Laboratory, Zhengzhou University and Nanyang Technological University. https://github.com/P-N-Suganthan/CEC2017-BoundContrained/blob/master/Definitions of CEC2017 benchmark suite final version updated.pdf.

Bashir, M., & Farooq, R. (2019). The synergetic effect of knowledge management and business model innovation on firm competence: A systematic review. In International Journal of Innovation Science (Vol. 11, Issue 3, pp. 362–387). Emerald Group Publishing Ltd. 10.1108/IJIS-10-2018-0103.

Bednarz, J. C. (1988). Cooperative hunting in Harris' hawks (Parabuteo unicinctus). *Science, 239*(4847), 1525–1527. https://doi.org/10.1126/science.239.4847.1525

Biswas, S., Saha, D., De, S., Cobb, A.D., Das S., & Jalaian, B. A. (2021). Improving Differential Evolution through Bayesian Hyperparameter Optimization. 2021 IEEE Congress on Evolutionary Computation (CEC), Kraków, Poland. https://github.com/subhodipbiswas/MadDE.

Blomkvist, O., Bremberg, S., & Zauer, R. (2012). *Mathematical modeling of flocking behavior*. Royal Institute of Technology.

Boussaïd, I., Lepagnot, J., & Siarry, P. (2013). A survey on optimization metaheuristics. *Information Sciences, 237*, 82–117. https://doi.org/10.1016/j.ins.2013.02.041

Brest, J., Maucec, M. S., & Boskovic, B. (2020). Differential evolution algorithm for single objective bound-constrained optimization: Algorithm j2020. In *2020 IEEE congress on evolutionary computation, CEC 2020 – conference proceedings*. https://doi.org/10.1109/CEC48606.2020.9185551

Chawla, M., & Duhan, M. (2018). Levy flights in metaheuristics optimization algorithms–a review. *Applied Artificial Intelligence, 32*(9–10), 802–821. https://doi.org/10.1080/08839514.2018.1508807

Chen, Z., Zhou, S., & Luo, J. (2017). A robust ant colony optimization for continuous functions. *Expert Systems with Applications, 81*, 309–320. https://doi.org/10.1016/j.eswa.2017.03.036

Cheng, R., & Jin, Y. (2015). A social learning particle swarm optimization algorithm for scalable optimization. *Information Sciences, 291*, 43–60. https://doi.org/10.1016/j.ins.2014.08.039

Chetty, S., & Adewumi, A. O. (2014). Comparison study of swarm intelligence techniques for the annual crop planning problem. *IEEE Transactions on Evolutionary Computation, 18*(2), 258–268. https://doi.org/10.1109/TEVC.423510.1109/TEVC.2013.2256427

Coello Coello, C. A. (2009). Evolutionary multi-objective optimization: Some current research trends and topics that remain to be explored. *Frontiers of Computer Science in China, 3*(1), 18–30. https://doi.org/10.1007/s11704-009-0005-7

Deb, K. (2014). Multi-objective optimization. In Search Methodologies: Introductory Tutorials in Optimization and Decision Support Techniques, Second Edition. 10.1007/978-1-4614-6940-7_15.

Derigs, U., & Nickel, N. H. (2003). Meta-heuristic based decision support for portfolio optimization with a case study on tracking error minimization in passive portfolio management. In OR Spectrum (Vol. 25, Issue 3, pp. 345–378). Springer Verlag. 10.1007/s00291-003-0127-5.

Dhargupta, S., Ghosh, M., Mirjalili, S., & Sarkar, R. (2020). Selective opposition based grey wolf optimization. *Expert Systems with Applications, 151*, 113389. https://doi.org/10.1016/j.eswa.2020.113389

Dhiman, G., Garg, M., Nagar, A., Kumar, V., & Dehghani, M. (2020). A novel algorithm for global optimization: Rat Swarm Optimizer. *Journal of Ambient Intelligence and Humanized Computing, 1*, 3. https://doi.org/10.1007/s12652-020-02580-0

Dhiman, G., & Kaur, A. (2019a). STOA: A bio-inspired based optimization algorithm for industrial engineering problems. *Engineering Applications of Artificial Intelligence, 82*, 148–174. https://doi.org/10.1016/j.engappai.2019.03.021

Dhiman, G., & Kumar, V. (2017). Spotted hyena optimizer: A novel bio-inspired based metaheuristic technique for engineering applications. *Advances in Engineering Software, 114*, 48–70. https://doi.org/10.1016/j.advengsoft.2017.05.014

Dhiman, G., & Kumar, V. (2019b). Seagull optimization algorithm: Theory and its applications for large-scale industrial engineering problems. *Knowledge-Based Systems, 165*, 169–196. https://doi.org/10.1016/j.knosys.2018.11.024

Dorigo, M., Birattari, M., & Stützle, T. (2006). Ant colony optimization artificial ants as a computational intelligence technique. *IEEE Computational Intelligence Magazine, 1*(4), 28–39. https://doi.org/10.1109/CI-M.2006.248054

Ergezer, M., Simon, D., & Du, D. (2009). Oppositional biogeography-based optimization. *Conference Proceedings – IEEE International Conference on Systems, Man and Cybernetics.*. https://doi.org/10.1109/ICSMC.2009.5346043

Friedman, M. (1937). The Use of Ranks to Avoid the Assumption of Normality Implicit in the Analysis of Variance. *Journal of the American Statistical Association, 32*(200), 675–701. https://doi.org/10.1080/01621459.1937.10503522

Gandomi, A. H., & Alavi, A. H. (2012). Krill herd: A new bio-inspired optimization algorithm. *Communications in Nonlinear Science and Numerical Simulation, 17*(12), 4831–4845. https://doi.org/10.1016/j.cnsns.2012.05.010

Gupta, S., & Deep, K. (2019). A hybrid self-adaptive sine cosine algorithm with opposition based learning. *Expert Systems with Applications, 119*, 210–230. https://doi.org/10.1016/j.eswa.2018.10.050

Halim, A. H., Ismail, I., & Das, S. (2020). Performance assessment of the metaheuristic optimization algorithms: An exhaustive review. *Artificial Intelligence Review, 1–87*. https://doi.org/10.1007/s10462-020-09906-6

Hansen, M. T., Nohira, N., & Tierney, T. J. (2003). What's your strategy for managing knowledge? *Harvard Business Review*.

Hansen, N., Müller, S. D., & Koumoutsakos, P. (2003). Reducing the time complexity of the derandomized evolution strategy with covariance matrix adaptation (CMA-ES). *Evolutionary Computation, 11*(1), 1–18. https://doi.org/10.1162/106365603321828970

Harifi, S., Khalilian, M., Mohammadzadeh, J., & Ebrahimnejad, S. (2019). Emperor Penguins Colony: A new metaheuristic algorithm for optimization. *Evolutionary Intelligence, 12*(2), 211–226. https://doi.org/10.1007/s12065-019-00212-x

Hashim, F. A., Houssein, E. H., Mabrouk, M. S., Al-Atabany, W., & Mirjalili, S. (2019). Henry gas solubility optimization: A novel physics-based algorithm. *Future Generation Computer Systems, 101*, 646–667. https://doi.org/10.1016/j.future.2019.07.015

Hashim, F. A., Hussain, K., Houssein, E. H., Mabrouk, M. S., & Al-Atabany, W. (2021). Archimedes optimization algorithm: A new metaheuristic algorithm for solving optimization problems. *Applied Intelligence, 51*(3), 1531–1551. https://doi.org/10.1007/s10489-020-01893-z

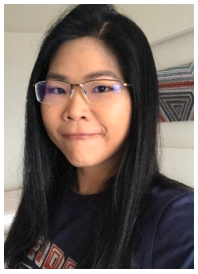
Heidari, A. A., Mirjalili, S., Faris, H., Aljarah, I., Mafarja, M., & Chen, H. (2019). Harris hawks optimization: Algorithm and applications. *Future Generation Computer Systems, 97*, 849–872. https://doi.org/10.1016/j.future.2019.02.028

Henson, S. M., & Hayward, J. L. (2010). The mathematics of animal behavior: An interdisciplinary dialogue. *Notices of the American Mathematical Society, 57*(10).
Houssein, E. H., Hosney, M. E., Oliva, D., Mohamed, W. M., & Hassaballah, M. (2020). A novel hybrid Harris hawks optimization and support vector machines for drug design and discovery. *Computers and Chemical Engineering, 133*, 106656. https://doi.org/10.1016/j.compchemeng.2019.106656
Houssein, E. H., Saad, M. R., Hussain, K., Zhu, W., Shaban, H., & Hassaballah, M. (2020). Optimal sink node placement in large scale wireless sensor networks based on harris' hawk optimization algorithm. *IEEE Access, 8*, 19381–19397. https://doi.org/10.1109/Access.628763910.1109/ACCESS.2020.2968981
Hu, X., Shonkwiler, R., & Spruill, M. (1994). Random restarts in global optimization. 1–32. https://smartech.gatech.edu/handle/1853/31310.
Hussain, K., Mohd Salleh, M. N., Cheng, S., & Shi, Y. (2019). Metaheuristic research: A comprehensive survey. *Artificial Intelligence Review, 52*(4), 2191–2233. https://doi.org/10.1007/s10462-017-9605-z
Jahani, E., & Chizari, M. (2018). Tackling global optimization problems with a novel algorithm – Mouth Brooding Fish algorithm. *Applied Soft Computing Journal, 62*, 987–1002. https://doi.org/10.1016/j.asoc.2017.09.035
Jain, M., Maurya, S., Rani, A., & Singh, V. (2018). Owl search algorithm: A novel nature-inspired heuristic paradigm for global optimization. *Journal of Intelligent and Fuzzy Systems, 34*(3), 1573–1582. https://doi.org/10.3233/JIFS-169452
Jamil, M., & Zepernick, H. J. (2013). Lévy flights and global optimization. In *Swarm intelligence and bio-inspired computation* (pp. 49–72). Elsevier Inc.. https://doi.org/10.1016/B978-0-12-405163-8.00003-X
Kamaruzaman, A. F., Zain, A. M., Yusuf, S. M., & Udin, A. (2013). Levy flight algorithm for optimization problems – A literature review. *Applied Mechanics and Materials, 421*, 496–501. https://doi.org/10.4028/www.scientific.net/AMM.421.496
Karaboga, D., & Basturk, B. (2007). A powerful and efficient algorithm for numerical function optimization: Artificial bee colony (ABC) algorithm. *Journal of Global Optimization, 39*(3), 459–471. https://doi.org/10.1007/s10898-007-9149-x
Kaur, S., Awasthi, L. K., Sangal, A. L., & Dhiman, G. (2020). Tunicate Swarm Algorithm: A new bio-inspired based metaheuristic paradigm for global optimization. *Engineering Applications of Artificial Intelligence, 90*, 103541. https://doi.org/10.1016/j.engappai.2020.103541
Kennedy, J., & Eberhart, R. (1995). Particle swarm optimization. *IEEE International Conference on Neural Networks - Conference Proceedings, 4*. https://doi.org/10.4018/ijmfmp.2015010104
Kianto, A., Shujahat, M., Hussain, S., Nawaz, F., & Ali, M. (2019). The impact of knowledge management on knowledge worker productivity. *Baltic Journal of Management, 14*(2), 178–197. https://doi.org/10.1108/BJM-12-2017-0404
Kiranyaz, S. (2014). Particle swarm optimization. *Adaptation, Learning, and Optimization, 15*, 45–82. https://doi.org/10.1007/978-3-642-37846-1_3
Korimbocus, M. A., Towokul, T., & Nagowah, S. D. (2020). A survey of knowledge capture and knowledge sharing techniques in agile software companies. *Advances in Intelligent Systems and Computing, 1089*, 567–578. https://doi.org/10.1007/978-981-15-1483-8_47
Li, J., Chen, T., Zhang, T., & Li, Y. X. (2016). A cuckoo optimization algorithm using elite opposition-based learning and chaotic disturbance. *Journal of Software Engineering, 10*(1), 16–28. https://doi.org/10.3923/jse.2016.16.28
Li, J., Li, Y. X., Tian, S. S., & Zou, J. (2019). Dynamic cuckoo search algorithm based on Taguchi opposition-based search. *International Journal of Bio-Inspired Computation, 13*(1), 59. https://doi.org/10.1504/IJBIC.2019.097728
Li, J., Li, Y. xiang, Tian, S. sha, & Xia, J. lin. (2020). An improved cuckoo search algorithm with self-adaptive knowledge learning. Neural Computing and Applications, 32(16). 10.1007/s00521-019-04178-w.
Li, J., Xiao, D. dan, Lei, H., Zhang, T., & Tian, T. (2020). Using cuckoo search algorithm with Q-learning and genetic operation to solve the problem of logistics distribution center location. *Mathematics, 8*(2), 149. https://doi.org/10.3390/math8020149
Li, S., Chen, H., Wang, M., Heidari, A. A., & Mirjalili, S. (2020). Slime mould algorithm: A new method for stochastic optimization. *Future Generation Computer Systems, 111*, 300–323. https://doi.org/10.1016/j.future.2020.03.055
Li, W., & Wang, G. G. (2021). Elephant herding optimization using dynamic topology and biogeography-based optimization based on learning for numerical optimization. *Engineering with Computers*. https://doi.org/10.1007/s00366-021-01293-y
Li, W., Wang, G. G., & Alavi, A. H. (2020). Learning-based elephant herding optimization algorithm for solving numerical optimization problems. *Knowledge-Based Systems, 195*, 105675. https://doi.org/10.1016/j.knosys.2020.105675
Liang, J. J., Qu, B. Y., & Suganthan, P. N. (2013). Problem Definitions and Evaluation Criteria for the CEC 2014 Special Session and Competition on Single Objective Real-Parameter Numerical Optimization. http://www.ntu.edu.sg/home/EPNSugan/index_files/CEC2014.
Lim, S. R., Kim, Y. R., Woo, S. H., Park, D., & Park, J. M. (2013). System optimization for eco-design by using monetization of environmental impacts: A strategy to convert bi-objective to single-objective problems. *Journal of Cleaner Production, 39*, 303–311. https://doi.org/10.1016/j.jclepro.2012.07.040
Liu, Y., Chong, G., Heidari, A. A., Chen, H., Liang, G., Ye, X., et al. (2020). Horizontal and vertical crossover of Harris hawk optimizer with Nelder-Mead simplex for parameter estimation of photovoltaic models. *Energy Conversion and Management, 223*, 113211. https://doi.org/10.1016/j.enconman.2020.113211
Lu, X., & Zhou, Y. (2008). A novel global convergence algorithm: Bee collecting pollen algorithm. Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), 5227 LNAI, 518–525. 10.1007/978-3-540-85984-0_62.
Mahdavi, S., Rahnamayan, S., & Deb, K. (2018). Opposition based learning: A literature review. *Swarm and Evolutionary Computation, 39*, 1–23. https://doi.org/10.1016/j.swevo.2017.09.010
Mirjalili, S. (2015). Moth-flame optimization algorithm: A novel nature-inspired heuristic paradigm. *Knowledge-Based Systems, 89*, 228–249. https://doi.org/10.1016/j.knosys.2015.07.006
Mirjalili, S. (2016a). SCA: A Sine Cosine Algorithm for solving optimization problems. *Knowledge-Based Systems, 96*, 120–133. https://doi.org/10.1016/j.knosys.2015.12.022
Mirjalili, S. (2016b). Dragonfly algorithm: A new meta-heuristic optimization technique for solving single-objective, discrete, and multi-objective problems. *Neural Computing and Applications, 27*(4), 1053–1073. https://doi.org/10.1007/s00521-015-1920-1
Mirjalili, S., & Lewis, A. (2016). The Whale Optimization Algorithm. *Advances in Engineering Software, 95*, 51–67. https://doi.org/10.1016/j.advengsoft.2016.01.008
Mirjalili, S., Mirjalili, S. M., & Lewis, A. (2014). Grey Wolf Optimizer. *Advances in Engineering Software, 69*, 46–61. https://doi.org/10.1016/j.advengsoft.2013.12.007
Mohamed, A. W., Hadi, A. A., Mohamed, A. K., & Awad, N. H. (2020). Evaluating the Performance of Adaptive GainingSharing Knowledge Based Algorithm on CEC 2020 Benchmark Problems. In *2020 IEEE Congress on Evolutionary Computation, CEC 2020 – Conference Proceedings*. https://doi.org/10.1109/CEC48606.2020.9185901
Morales-Castañeda, B., Zaldívar, D., Cuevas, E., Fausto, F., & Rodríguez, A. (2020). A better balance in metaheuristic algorithms: Does it exist? *Swarm and Evolutionary Computation, 54*, Article 100671. https://doi.org/10.1016/j.swevo.2020.100671
Mucherino, A., & Seref, O. (2007). Monkey search: A novel metaheuristic search for global optimization. *AIP Conference Proceedings, 953*, 162–173. https://doi.org/10.1063/1.2817338
Nadimi-Shahraki, M. H., Taghian, S., & Mirjalili, S. (2021). An improved grey wolf optimizer for solving engineering problems. *Expert Systems with Applications, 166*, 113917. https://doi.org/10.1016/j.eswa.2020.113917
Nadimi-Shahraki, M. H., Taghian, S., Mirjalili, S., & Faris, H. (2020). MTDE: An effective multi-trial vector-based differential evolution algorithm and its applications for engineering design problems. *Applied Soft Computing Journal, 97*, 106761. https://doi.org/10.1016/j.asoc.2020.106761
Neshat, M., Sepidnam, G., & Sargolzaei, M. (2013). Swallow swarm optimization algorithm: A new method to optimization. *Neural Computing and Applications, 23*(2), 429–454. https://doi.org/10.1007/s00521-012-0939-9
Neumann, F., & Wegener, I. (2007). Can Single-Objective Optimization Profit from Multiobjective Optimization? In Multiobjective Problem Solving from Nature. 10.1007/978-3-540-72964-8_6.
Olorunda, O., & Engelbrecht, A. P. (2008). Measuring exploration/exploitation in particle swarms using swarm diversity. 2008 IEEE Congress on Evolutionary Computation, CEC 2008, 1128–1134. 10.1109/CEC.2008.4630938.
Ostertagová, E., Ostertag, O., & Kováč, J. (2014). Methodology and application of the Kruskal-Wallis test. *Applied Mechanics and Materials, 611*, 115–120. https://doi.org/10.4028/www.scientific.net/AMM.61110.4028/www.scientific.net/AMM.611.115
Ouriques, R. A. B., Wnuk, K., Gorschek, T., & Svensson, R. B. (2019). Knowledge management strategies and processes in agile software development: A systematic literature review. *International Journal of Software Engineering and Knowledge Engineering, 29*(03), 345–380. https://doi.org/10.1142/S0218194019500153
Pholdee, N., & Bureerat, S. (2014). Comparative performance of meta-heuristic algorithms for mass minimisation of trusses with dynamic constraints. *Advances in Engineering Software, 75*, 1–13. https://doi.org/10.1016/j.advengsoft.2014.04.005
Pierezan, J., & Dos Santos Coelho, L. (2018, September 28). Coyote Optimization Algorithm: A New Metaheuristic for Global Optimization Problems. 2018 IEEE Congress on Evolutionary Computation, CEC 2018 - Proceedings. 10.1109/CEC.2018.8477769.
Rahnamayan, S., Tizhoosh, H. R., & Salama, M. M. A. (2007). Opposition-based differential evolution (ODE) with variable jumping rate. Proceedings of the 2007 IEEE Symposium on Foundations of Computational Intelligence, FOCI 2007, 81–88. 10.1109/FOCI.2007.372151.
Rahnamayan, Shahryar. (2007). Opposition-Based Differential Evolution [University of Waterloo]. http://hdl.handle.net/10012/2784.
Rahnamayan, Shahryar, Tizhoosh, H. R., & Salama, M. M. A. (2007). Quasi-oppositional differential evolution. 2007 IEEE Congress on Evolutionary Computation, CEC 2007, 2229–2236. 10.1109/CEC.2007.4424748.
Rajabioun, R. (2011). Cuckoo optimization algorithm. *Applied Soft Computing Journal, 11*(8), 5508–5518. https://doi.org/10.1016/j.asoc.2011.05.008
Rodríguez-Esparza, E., Zanella-Calzada, L. A., Oliva, D., Heidari, A. A., Zaldivar, D., Pérez-Cisneros, M., et al. (2020). An efficient Harris hawks-inspired image segmentation method. *Expert Systems with Applications, 155*, 113428. https://doi.org/10.1016/j.eswa.2020.113428
Rojas-Morales, N., Riff Rojas, M. C., & Montero Ureta, E. (2017). A survey and classification of Opposition-Based Metaheuristics. *Computers and Industrial Engineering, 110*, 424–435. https://doi.org/10.1016/j.cie.2017.06.028
Sallam, K. M., Elsayed, S. M., Chakrabortty, R. K., & Ryan, M. J. (2020). Improved multi-operator differential evolution algorithm for solving unconstrained problems. In *2020 IEEE congress on evolutionary computation, CEC 2020 – Conference proceedings*. https://doi.org/10.1109/CEC48606.2020.9185577
Saremi, S., Mirjalili, S., & Lewis, A. (2017). Grasshopper Optimisation Algorithm: Theory and application. *Advances in Engineering Software, 105*, 30–47. https://doi.org/10.1016/j.advengsoft.2017.01.004
Sathish, T. (2019). Profit maximization in reverse logistics based on disassembly scheduling using hybrid bee colony and bat optimization. *Transactions of the Canadian Society for Mechanical Engineering, 43*(4), 551–559. https://doi.org/10.1139/tcsme-2019-0017
Selvam, P. P., & Narayanan, R. (2019). Random Restart Local Search Optimization technique for sustainable energy-generating induction machine. *Computers and*

Electrical Engineering, 73, 268–278. https://doi.org/10.1016/j.compeleceng.2018.11.023
Sharma, N., & Gupta, V. (2020). ScienceDirect ScienceDirect Meta-heuristic based optimization of WSNs Localisation Problem-a Survey-NC-ND license (http://creativecommons.org/licenses/by-nc-nd/4.0/) Peer-review under responsibility of the scientific committee of the International Conference on Smart Sustainable Intelligent Computing and Applications under ICITETM2020. Procedia Computer Science, 173, 36–45. 10.1016/j.procs.2020.06.006.
Storn, R., & Price, K. (1997). Differential evolution – A simple and efficient heuristic for global optimization over continuous spaces. *Journal of Global Optimization, 11*(4). https://doi.org/10.1023/A:1008202821328
Suganthan, P. N., Hansen, N., Liang, J. J., Deb, K., Chen, Y. P., Auger, A., & Tiwari, S. (2005). Problem definitions and evaluation criteria for the CEC 2005 special session on real-parameter optimization. Technical Report, Nanyang Technological University, Singapore, May 2005 AND KanGAL Report 2005005, IIT Kanpur, India. https://www.researchgate.net/publication/235710019.
Sulaiman, M. H., Mustaffa, Z., Saari, M. M., Daniyal, H., Musirin, I., & Daud, M. R. (2019). Barnacles mating optimizer: An evolutionary algorithm for solving optimization. Proceedings - 2018 IEEE International Conference on Automatic Control and Intelligent Systems, I2CACIS 2018, 99–104. 10.1109/I2CACIS.2018.8603703.
Tan, D., Luo, W., & Liu, Q. (2009). Multi-objective particle swarm optimization algorithm for engineering constrained optimization problems. 2009 IEEE International Conference on Granular Computing, GRC 2009. 10.1109/GRC.2009.5255064.
Tanabe, R., & Fukunaga, A. S. (2014). Improving the search performance of SHADE using linear population size reduction. Proceedings of the 2014 IEEE Congress on Evolutionary Computation, CEC 2014. 10.1109/CEC.2014.6900380.
Tizhoosh, H. R. (2005). Opposition-based learning: A new scheme for machine intelligence. Proceedings - International Conference on Computational Intelligence for Modelling, Control and Automation, CIMCA 2005 and International Conference on Intelligent Agents, Web Technologies and Internet, 1, 695–701. 10.1109/cimca.2005.1631345.
Tron, E., & Margaliot, M. (2004). Mathematical modeling of observed natural behavior: A fuzzy logic approach. *Fuzzy Sets and Systems, 146*(3), 437–450. https://doi.org/10.1016/j.fss.2003.09.005
Tsai, C. W., & Liu, S. J. (2019). SEIM: Search economics for influence maximization in online social networks. *Future Generation Computer Systems, 93*, 1055–1064. https://doi.org/10.1016/j.future.2018.08.033
Tsibulsky, V., & Norman, A. (2007). Mathematical models of behavior of individual animals. *Current Pharmaceutical Design, 13*(15), 1571–1595. https://doi.org/10.2174/138161207780765873
Tubishat, M., Idris, N., Shuib, L., Abushariah, M. A. M., & Mirjalili, S. (2020). Improved Salp Swarm Algorithm based on opposition based learning and novel local search algorithm for feature selection. *Expert Systems with Applications, 145*, 113122. https://doi.org/10.1016/j.eswa.2019.113122
Wang, G. G. (2018). Moth search algorithm: A bio-inspired metaheuristic algorithm for global optimization problems. *Memetic Computing, 10*(2), 151–164. https://doi.org/10.1007/s12293-016-0212-3
Wang, G. G., Deb, S., & Coelho, L. D. S. (2016). Elephant Herding Optimization. Proceedings - 2015 3rd International Symposium on Computational and Business Intelligence, ISCBI 2015, 1–5. 10.1109/ISCBI.2015.8.
Wang, G. G., Deb, S., & Dos Santos Coelho, L. (2018). Earthworm optimisation algorithm: A bio-inspired metaheuristic algorithm for global optimisation problems. *International Journal of Bio-Inspired Computation, 12*(1), 1. https://doi.org/10.1504/IJBIC.2018.093328
Wang, G. G., Deb, S., Gandomi, A. H., & Alavi, A. H. (2016). Opposition-based krill herd algorithm with Cauchy mutation and position clamping. *Neurocomputing, 177*, 147–157. https://doi.org/10.1016/j.neucom.2015.11.018
Wang, H., Li, K., & Pedrycz, W. (2020). An elite hybrid metaheuristic optimization algorithm for maximizing wireless sensor networks lifetime with a sink node. *IEEE Sensors Journal, 20*(10), 5634–5649. https://doi.org/10.1109/JSEN.736110.1109/JSEN.2020.2971035
Wang, H., Wu, Z., Rahnamayan, S., Liu, Y., & Ventresca, M. (2011). Enhancing particle swarm optimization using generalized opposition-based learning. *Information Sciences, 181*(20), 4699–4714. https://doi.org/10.1016/j.ins.2011.03.016
Whitley, D. (1994). A genetic algorithm tutorial. *Statistics and Computing, 4*(2). https://doi.org/10.1007/BF00175354
Wolpert, D. H., & Macready, W. G. (1997). No free lunch theorems for optimization. *IEEE Transactions on Evolutionary Computation, 1*(1), 67–82. https://doi.org/10.1109/4235.585893
Woolson, R. F. (2005). Wilcoxon Signed-Rank Test. In Encyclopedia of Biostatistics. John Wiley & Sons, Ltd. 10.1002/0470011815.b2a15177.
Wu, J., Wang, Y. G., Burrage, K., Tian, Y. C., Lawson, B., & Ding, Z. (2020). An improved firefly algorithm for global continuous optimization problems. *Expert Systems with Applications, 149*, 113340. https://doi.org/10.1016/j.eswa.2020.113340
Wu, X., Zhou, Y., & Lu, Y. (2017). Elite opposition-based water wave optimization algorithm for global optimization. *Mathematical Problems in Engineering, 2017*, 1–25. https://doi.org/10.1155/2017/3498363
Xu, J., & Zhang, J. (2014). Exploration-exploitation tradeoffs in metaheuristics: Survey and analysis. Proceedings of the 33rd Chinese Control Conference, CCC 2014, 8633–8638. 10.1109/ChiCC.2014.6896450.
Xu, Q., Wang, L., Wang, N., Hei, X., & Zhao, L. (2014). A review of opposition-based learning from 2005 to 2012. In Engineering Applications of Artificial Intelligence (Vol. 29, pp. 1–12). Elsevier Ltd. 10.1016/j.engappai.2013.12.004.
Xu, Y., Yang, Z., Li, X., Kang, H., & Yang, X. (2020). Dynamic opposite learning enhanced teaching–learning-based optimization. *Knowledge-Based Systems, 188*, 104966. https://doi.org/10.1016/j.knosys.2019.104966
Xue, J., & Shen, B. (2020). A novel swarm intelligence optimization approach: Sparrow search algorithm. *Systems Science & Control Engineering, 8*(1), 22–34. https://doi.org/10.1080/21642583.2019.1708830
Yang, C. H., Liu, Y. T., & Chuang, L. Y. (2011). DNA motif discovery based on ant colony optimization and expectation maximization. IMECS 2011 - International MultiConference of Engineers and Computer Scientists 2011, 1, 169–174. https://www.researchgate.net/publication/50864202_DNA_Motif_Discovery_Based_on_Ant_Colony_Optimization_and_Expectation_Maximization.
Yang, X., & Huang, Z. (2012). Opposition-based Artificial Bee Colony with dynamic Cauchy mutation for function optimization. International Journal of Advancements in Computing Technology, 4(4), 56–62. 10.4156/ijact.vol4.issue4.8.
Yang, X. S. (2010). A new metaheuristic Bat-inspired Algorithm. *Studies in Computational Intelligence, 284*, 65–74. https://doi.org/10.1007/978-3-642-12538-6_6
Yang, X. S., & Deb, S. (2009). Cuckoo search via Lévy flights. 2009 World Congress on Nature and Biologically Inspired Computing, NABIC 2009 - Proceedings, 210–214. 10.1109/NABIC.2009.5393690.
Yang, X. S., Deb, S., Fong, S., He, X., & Zhao, Y. X. (2016). From swarm intelligence to metaheuristics: nature-inspired optimization algorithms. *Computer, 49*(9), 52–59. https://doi.org/10.1109/MC.2016.292
Yang, X. S., Deb, S., Zhao, Y. X., Fong, S., & He, X. (2018). Swarm intelligence: Past, present and future. *Soft Computing, 22*(18), 5923–5933. https://doi.org/10.1007/s00500-017-2810-5
Yazdani, M., & Jolai, F. (2016). Lion Optimization Algorithm (LOA): A nature-inspired metaheuristic algorithm. *Journal of Computational Design and Engineering, 3*, 24–36. https://doi.org/10.1016/j.jcde.2015.06.003
Yong, W., Tao, W., Cheng-Zhi, Z., & Hua-Juan, H. (2016). A New Stochastic Optimization Approach: Dolphin Swarm Optimization Algorithm. *International Journal of Computational Intelligence and Applications, 15*(02), 1650011. https://doi.org/10.1142/S1469026816500115
Zervoudakis, K., & Tsafarakis, S. (2020). A mayfly optimization algorithm. *Computers and Industrial Engineering, 145*, 106559. https://doi.org/10.1016/j.cie.2020.106559
Zhang, H., Wang, Z., Chen, W., Heidari, A. A., Wang, M., Zhao, X., et al. (2021). Ensemble mutation-driven salp swarm algorithm with restart mechanism: Framework and fundamental analysis. *Expert Systems with Applications, 165*, Article 113897. https://doi.org/10.1016/j.eswa.2020.113897

Florentina Yuni Arini obtained her bachelor's (2002) and master's (2009) in Computer Science from Universitas Gadjah Mada, Yogyakarta, Indonesia. She is a lecturer at the Department of Computer Science, Universitas Negeri Semarang, Indonesia. Currently, she is a Ph.D. scholar at the Department of Computer Science, Khon Kaen University, Thailand. Her research interest focus on Swarm Intelligence and its application.

Sirapat Chiewchanwattana is an Associate Professor at the Department of Computer Science, Khon Kaen University, Thailand. She received her Ph.D. (2005) in Computer Science from Chulalongkorn University, Thailand. She received her M. S. (1993) in Computer Science from the National Institute of Development Administration, Thailand, and B.S. (1984) in Statistics from Khon Kaen University, Thailand. Her research interests are machine learning, neural networks, soft computing, and pattern recognition.



Chitsutha Soomlek is an Assistant Professor at the Department of Computer Science, Khon Kaen University, Thailand. She received her Ph.D. (2013) and MA.Sc. (2006) in Electronic Systems Engineering from the University of Regina, Canada. In 2004, Soomlek received her B.E. in Computer Engineering from King Mongkut's Institute of Technology Ladkrabang, Bangkok, Thailand. Her research interests are in intelligent agent, multi-agent systems, computational intelligence for software engineering and secure software development, code smells, and software quality.

Khamron Sunat is an Assistant Professor at the Department of Computer Science, Khon Kaen University, Thailand. He received B.S. (1990) in Chemical Engineering from Chulalongkorn University, Thailand, M.S. (1999) in Computational Science, from Chulalongkorn University, Thailand, and Ph.D. (2003) in Computer Science from Chulalongkorn University, Thailand. His research interests are neural networks, pattern recognition, computer vision, soft computing, fuzzy systems, evolutionary computing, and machine learning.