Received July 16, 2021, accepted August 23, 2021, date of publication September 7, 2021, date of current version September 23, 2021.

Digital Object Identifier 10.1109/ACCESS.2021.3111033

Hybrid Sine Cosine and Fitness Dependent Optimizer for Global Optimization

PO CHAN CHIU[1,2,3], ALI SELAMAT[1,2,4,5], (Member, IEEE), ONDREJ KREJCAR[5], AND KING KUOK KUOK[6]

[1]School of Computing, Faculty of Engineering, Universiti Teknologi Malaysia, Johor Bahru, Johor 81310, Malaysia
[2]Media and Game Innovation Center of Excellence (MaGICX), Universiti Teknologi Malaysia, Johor Bahru, Johor 81310, Malaysia
[3]Faculty of Computer Science and Information Technology, Universiti Malaysia Sarawak, Kota Samarahan, Sarawak 94300, Malaysia
[4]Malaysia Japan International Institute of Technology (MJIIT), Universiti Teknologi Malaysia Kuala Lumpur, Kuala Lumpur 54100, Malaysia
[5]Faculty of Informatics and Management, University of Hradec Kralove, 500 03 Hradec Kralove, Czech Republic
[6]Faculty of Engineering, Computing and Science, Swinburne University of Technology Sarawak Campus, Kuching, Sarawak 93350, Malaysia

Corresponding authors: Po Chan Chiu (pcchiu@unimas.my) and Ali Selamat (aselamat@utm.my)

This work was supported in part by Universiti Teknologi Malaysia (UTM) under Research University Grant Vot-20H04, in part by Malaysia Research University Network (MRUN) under Grant Vot 4L876, in part by the Ministry of Higher Education through the Fundamental Research Grant Scheme (FRGS) under Grant Vot FRGS/1/2018/ICT04/UTM/01/1, in part by the Specific Research Project (SPEV) through the Faculty of Informatics and Management, University of Hradec Kralove, Czech Republic, under Grant 2102-2021, and in part by the Skim Latihan Akademik IPTA (SLAI) Scholarship through the Ministry of Higher Education Malaysia.

ABSTRACT The fitness-dependent optimizer (FDO), a newly proposed swarm intelligent algorithm, is focused on the reproductive mechanism of bee swarming and collective decision-making. To optimize the performance, FDO calculates velocity (pace) differently. FDO calculates weight using the fitness function values to update the search agent position during the exploration and exploitation phases. However, the FDO encounters slow convergence and unbalanced exploitation and exploration. Hence, this study proposes a novel hybrid of the sine cosine algorithm and fitness-dependent optimizer (SC-FDO) for updating the velocity (pace) using the sine cosine scheme. This proposed algorithm, SC-FDO, has been tested over 19 classical and 10 IEEE Congress of Evolutionary Computation (CEC-C06 2019) benchmark test functions. The findings revealed that SC-FDO achieved better performances in most cases than the original FDO and well-known optimization algorithms. The proposed SC-FDO improved the original FDO by achieving a better exploit-explore tradeoff with a faster convergence speed. The SC-FDO was applied to the missing data estimation cases and refined the missingness as optimization problems. This is the first time, to our knowledge, that nature-inspired algorithms have been considered for handling time series datasets with low and high missingness problems (10%-90%). The impacts of missing data on the predictive ability of the proposed SC-FDO were evaluated using a large weather dataset from 1985 until 2020. The results revealed that the imputation sensitivity depends on the percentages of missingness and the imputation models. The findings demonstrated that the SC-FDO based multilayer perceptron (MLP) trainer outperformed the other three optimizer trainers with the highest average accuracy of 90% when treating the high-low missingness in the dataset.

INDEX TERMS Fitness dependent optimizer, sine cosine algorithm, missing data, high missing rates, imputation, metaheuristic algorithms, optimization, neural network.

I. INTRODUCTION

Nature-inspired algorithms, also known as metaheuristic algorithms, have received great attention from technology, engineering, management, and different areas of study to solve problems with optimization. Nature-inspired algorithms include particle swarm optimization (PSO) [1], differential evaluation (DE) [2] and genetic algorithm (GA) [3]. Some of the recent nature-inspired algorithms are sine cosine algorithm (SCA) [4], [5], fitness dependent optimizer (FDO) [6], wingsuit flying search (WFS) [7], whale optimization algorithm (WOA) [8], [9], butterfly optimization algorithm (BOA) [10], [11], dragonfly algorithm (DA) [12], [13], grey wolf optimizer (GWO) [14],

The associate editor coordinating the review of this manuscript and approving it for publication was Juntao Fei.

moth-flame optimization algorithm (MFO) [15]–[18], root-based optimization algorithm [19], coot algorithm [20] and colony predation algorithm [21].

This paper focuses on the fitness-dependent optimizer (FDO) proposed by Abdullah and Rashid [6]. The FDO is inspired by the reproductive mechanism of bee swarming and collective decision-making. The FDO has been evaluated with well-known benchmark test functions and achieved good performance than other nature-inspired algorithms, namely DA, PSO, GA, and WOA. The FDO has been effectively optimized the controller of a multi-source interconnected power system [22], [23]. Meanwhile, an adaptive FDO (AFDO) algorithm based on the first fit (FF) heuristic approach is proposed to handle the problem of one-dimensional bin packing [24]. The AFDO has effectively explored the search space with the lowest fitness values within an acceptable time for the discrete optimization problems.

Furthermore, Muhammed *et al.* [25] developed an improved fitness-dependent optimizer (IFDO) algorithm based on alignment and cohesion strategy to update the scout bees' location. The introduction of a random weight factor (wf), alignment and cohesion features in the IFDO improved the convergence speed of the FDO. Still, the enhancement features increased the algorithm's space complexity and led to slower exploitations in some cases. Additionally, Daraz *et al.* [26] has successfully adopted the IFDO to optimize the automatic generation controller of a multi-source interconnected power system in the restructured environment. Next, Mohammed and Rashid [27] embedded chaos theory into the original FDO. The chaotic fitness-dependent optimizer (CFDO) has successfully improved the search capability and prevented the algorithm from falling into local optima; however, it is not always accurate in some cases when the problem is highly complex. The comparison of the FDO and its variations is discussed in Table 1.

Recently, many researchers have proposed several improved fitness-dependent optimizers from different perspectives to improve the original FDO. According to the No Free Lunch (NFL) theorem [28], a single optimization approach is impossible to manage all optimization problems adequately. Although the FDO and FDO variants outperformed several optimization algorithms, in some cases, they encounter slow convergence, poor exploitation and exploration, and memory wastage as a result of inefficient memory allocation.

In addition, the FDO lacked exploitability and suffered from slow convergence. Nevertheless, the most significant advantages of the FDO are its power of exploration and simplicity. The FDO's exploration and exploitation are mainly influenced by the fitness weight mechanism that guides scout bee decision-making. The fitness weight mechanism increased the diversity of solutions and strengthened the exploration ability of the FDO. For simple optimization problems, the fitness weight mechanism increased the exploration level of the FDO and escaped the search from local optima. However, the convergence speed of the FDO would increase and it is easily trapped into local optima if the optimization problems are complex. Therefore, the motivation of this work was to propose a balanced and straightforward way of gaining a better exploit-explore tradeoff algorithm with a faster convergence speed.

Based on the shortcomings of the FDO and its variants, we introduce an enhanced version of the FDO and hybrid it with SCA, a recent efficient population-based optimization algorithm. The enhancement of FDO is called a sine cosine-fitness dependent optimizer (SC-FDO). The key benefit of SCA is its high exploitation potential in the search solution [4]. Hence, the exploitation ability of the FDO is enhanced by incorporating SCA features to refine the best neighboring search and the FDO to explore the entire search space for promising solutions.

Additionally, in the related literature, a comprehensive position-updating strategy is commonly valuable to boost the efficiency of the swarm intelligent algorithms in the search space [29]–[34]. Inspired by this, a modified pace-updating equation is introduced to substitute the pace equation in the FDO. Another improvement is the proposed SC-FDO employs a global fitness weight (fw^*) that is best in earlier iterations to tune the random weight factors (wf) adaptively during the search process. Moreover, a conversion parameter is suggested for balancing the exploration and exploitation of the search spaces. The proposed SC-FDO also uses the best solution-updating strategy for reducing the computational time of the original FDO. The proposed SC-FDO is tested over well-known benchmark test functions and evaluated with existing nature-inspired optimization algorithms to verify the algorithm's efficiency. The numerical results and statistical analysis indicated that the proposed SC-FDO obtained the global best solution with higher accuracy than the compared optimization algorithms. Furthermore, the proposed SC-FDO has been extended to handle the problems of high missing values in datasets. The results revealed that the proposed SC-FDO achieved higher imputation accuracy and lower computational time compared to the FDO and IFDO imputation.

The contributions of this paper are:

1. A modified pace-updating equation, random weight factor (wf) and global fitness weight (fw^*) strategy, conversion parameter strategy and the best solution-updating strategy are introduced to boost the performances of the original FDO.
2. The numerical experiments and statistical analysis have shown the superior capability of the proposed SC-FDO on the benchmark test function, compared with well-known nature-inspired algorithms.
3. The missing data estimation experiments demonstrated that the SC-FDO based multilayer perceptron MLP) trainer can impute missing data for a low and large proportion of missingness with higher prediction accuracy while consuming lower computational time than to the original FDO and IFDO.

TABLE 1. The comparison of the FDO and its variations.

Variants of FDO	Author	Real-world applications	Strengths	Limitation
Fitness dependent optimizer (FDO)	Abdullah and Rashid (2019) [6]	❖ Aperiodic antenna array designs [6] ❖ Multi-source interconnected power system [22][23]	❖ Uses fitness function to generate suitable fitness weights for guiding the search agents during exploitation and exploration phases. ❖ Good at exploration.	❖ Poor exploitation. ❖ Imbalance of exploration and exploitation ability. ❖ Slow convergence.
Adaptive fitness-dependent optimizer (AFDO)	Abdul-Minaam et al. (2020) [24]	❖ One-dimensional bin packing [24]	❖ Adapted first-fit heuristic. ❖ Effectively explored the search space with the lowest fitness values within an acceptable time. ❖ To solve discrete optimization problems.	❖ Inefficient memory allocation leads to memory wastage. ❖ Not able to solve continuous optimization problems.
Improved fitness-dependent optimizer (IFDO)	Muhammed et al. (2020) [25]	❖ Aperiodic antenna array designs [25] ❖ Pedestrian evacuation model [25] ❖ Multi-source interconnected power system in the restructured environment [26]	❖ Uses alignment and cohesion to update the scout bees' location. ❖ Perform weight factor (wf) randomization.	❖ Increase space complexity of the algorithm. ❖ Take longer execution time. ❖ Poor exploitation.
Chaotic fitness dependent optimizer (CFDO)	Mohammed and Rashid (2021) [27]	❖ Pressure vessel design [27] ❖ Task assignment problem [27]	❖ Embedded chaotic maps. ❖ Local optima avoidance due to the dynamic and superlative way of generating random numbers. ❖ Enhanced the search capability of the original FDO.	❖ In some instances, it is highly complex and not consistently accurate.

The remainder of this paper is described as follows: Section II reviews the fundamentals of the FDO. The proposed SC-FDO is presented in Section III. Section IV discusses the numerical experiments and analysis of the SC-FDO on the benchmark test function. Section V provides the missing data imputation technique based on SC-FDO in solving high missing rates datasets. Section VI concludes the findings of this study, and Section VII describes the limitation and future works.

II. FITNESS DEPENDENT OPTIMIZER

The FDO is a newly designed swarm intelligent algorithm presented by Abdullah and Rashid [6], which was inspired by bee swarming characteristics during reproduction. The FDO is a PSO-based algorithm that imitates the position updating mechanism of the PSO. However, the FDO calculates velocity (pace) in a different strategy. It employs a fitness function to produce appropriate weights, and these weights will facilitate the search agents to balance exploration and exploitation.

In nature, bees live in groups (colonies) called hives, containing queen bee, worker bee, and scout bee. The queen bee is a decision-maker to keep the hive under control and lays all the eggs to maintain the hive population. The worker bees are responsible for all the works in the hive except reproducing. Meanwhile, the scout bees are responsible for finding a new home for future swarms. When a bee colony grows massively, the available space becomes smaller. Thus, the colony tries to solve the space problem by swarming, in which one colony becomes two colonies. The scout bees will find a nearby location for the swarm during swarming, approximately a few meters from the hive. The bees will leave the hive and temporarily cluster around their queen in the new place for one to few days. Then, the scout bees will travel in a small group, about 20 to 50 bees, to search for new hives. After finding the new hives, the scout bees communicate by moving their legs and wings to determine the most suitable hive. When the decision is taken, the rest of the bees fly off and move to the hive, where it begins its new colony life.

Inspired by the bee collective decision-making process, the FDO uses fitness weight (fw) to guide the search agents in identifying the best solution. Each hive represents a possible solution exploited by a search agent (artificial scout bee), and the best hive is defined as the global optimum solution. The hive specifications include volume, location, and size, represent the fitness function of the solution.

The FDO algorithm starts by assigning the scout bees population with random solutions, using the upper and lower boundaries. The scout bees search for hives using a combination of a random walk and fitness weight mechanism. The scout bees change their position by adding pace to the current position. The movement of the scout bees is calculated as:

$$x_{i,t+1} = x_{i,t} + pace \tag{1}$$

where i is the current search scout bee (search agent), t is the current iteration, x is the scout bee, and $pace$ is the movement rate and direction of the scout bee. The $pace$ is dependent on the value of fitness weight fw. The fw is calculated according to (2).

$$fw = \left| \frac{x^*_{i,tfitness}}{x_{i,tfitness}} \right| - wf \tag{2}$$

where wf is the weight factor (either 0 or 1), $x^*_{i,tfitness}$ is the fitness function of the global best solution and $x_{i,tfitness}$ is the fitness function of the current solution. Further, the conditions for fw are expressed as below:

$$\left\{ \begin{array}{ll} fw = 1 or fw = 0 or x_{i,tfitness} = 0, \;\; pace = x_{i,t} * r & \\ fw > 0 \; and \; fw < 1 & (3) \\ \left\{ \begin{array}{ll} r < 0, \quad pace = \left(x_{i,t} - x^*_{i,t}\right) * fw * -1 & (4) \\ r \geq 0, \quad pace = \left(x_{i,t} - x^*_{i,t}\right) * fw & (5) \end{array} \right. & \end{array} \right\}$$

where r $[-1, 1]$ is Levy random number, the Levy flight from [35] has been employed due to its good distribution curve. The pseudocode of FDO is presented in Fig. 1 [6].

III. THE HYBRID SINE COSINE-FITNESS DEPENDENT OPTIMIZER (SC-FDO)

In this section, the proposed hybrid sine cosine-fitness dependent optimizer (SC-FDO) is presented. The sine cosine algorithm is partially embedded into the FDO algorithm to improve the performance of the original FDO in terms of convergence speed, searching accuracy and balance of exploitation and exploration ability in the search space. The proposed SC-FDO is illustrated in Fig. 2. In this approach, four modifications are applied: (1) the modified pace-updating equation in search phase, (2) random weight factor (wf) and global fitness weight (fw^*) strategy, (3) the conversion parameter strategy, and (4) the best solution-updating strategy.

A. MODIFIED PACE-UPDATING EQUATION

The use of the fitness weight (fw) mechanism in the FDO inevitably leads to slow convergence. Due to the strong exploration ability of the FDO, the pace-updating strategy in (3)-(5) may increase the diversity of solutions that cause difficulty in finding the global optimum solution. Thus, the concept of modified pace-updating is introduced in this section to improve the convergence speed and balance of exploitation and exploration ability of the original FDO.

```
Initialize scout bee population, $x_{i,t}$ (i = 1,2,3,…,n)
While iteration (t) limit not reached
 For each artificial scout bee, $x_{i,t}$
     Find best artificial scout bee, $x^*_{i,t}$
     Generate random walk r in [-1,1] range
     If ($x_{i,t}$ fitness = 0) (avoid divide by zero)
         fitness weight = 0
     Else
         calculate fitness weight, equation (2)
     End if
     If (fitness weight = 1 or fitness weight = 0)
         calculate pace using equation (3)
     Else
         If (random number >=0)
             calculate pace using equation (5)
         Else
             calculate pace using equation (4)
         End if
     End if
     Calculate $x_{i,t+1}$ using equation (1)
     If ($x_{i,t+1}$ fitness < $x_{i,t}$ fitness)
         move accepted and pace saved
     Else
         calculate $x_{i,t+1}$ using equation (1) with previous pace
         If ($x_{i,t+1}$ fitness < $x_{i,t}$ fitness)
             move accepted and pace saved
         Else
             maintain current position (don't move)
         End if
     End if
   End for
End while
```

FIGURE 1. The pseudocode of FDO [6].

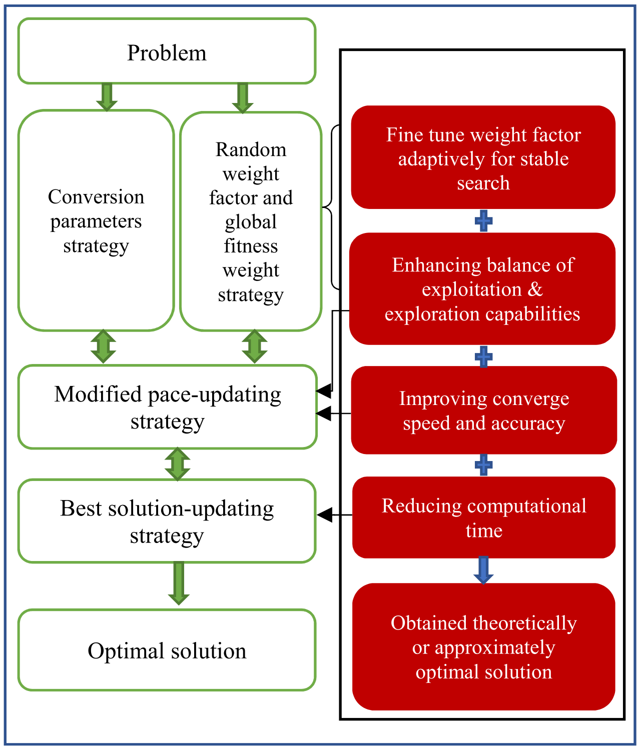

Problem
Conversion parameters strategy
Random weight factor and global fitness weight strategy
Modified pace-updating strategy
Best solution-updating strategy
Optimal solution
Fine tune weight factor adaptively for stable search
Enhancing balance of exploitation & exploration capabilities
Improving converge speed and accuracy
Reducing computational time
Obtained theoretically or approximately optimal solution

FIGURE 2. The proposed SC-FDO.

In [4], the work shows that the sine cosine algorithm (SCA) has high exploitation of the search space. SCA is a population-based algorithm introduced by Mirjalili [4]. We introduce a sine cosine scheme into the pace-updating mechanism of the original FDO. First, the modified pace-updating mechanism starts the search process to explore different promising solutions and foster the search to exploit the prominent regions. In addition, the modified pace-updating mechanism guides the search agents to achieve exploration and exploitation balancing. The modified pace-updating equation is calculated based on the following equations:

$$\left\{\begin{array}{ll} fw = 1, \quad pace = x_{i,t} * r & (6) \\ fw = 0, \quad pace = x_{i,t} + r_1 * \cos(r_2) & \\ \quad * (r_3 * x^*_{i,t} - x_{i,t}) * r & (7) \\ fw > 0 \; and \; fw < 1 & \\ \left\{\begin{array}{l} r < 0, \\ pace = (x_{i,t} + r_1 * sin(r_2) \\ \quad * (r_3 * x^*_{i,t} - x_{i,t}) * fw) * -1 \\ r \geq 0, \; pace = x_{i,t} + r_1 * \sin(r_2) \\ \quad * \left(r_3 * x^*_{i,t} - x_{i,t}\right) * fw \end{array}\right. & \begin{array}{l} \\ \\ (8) \\ \\ (9) \end{array} \end{array}\right\}$$

where r is Levy random number, r_1, r_2 and r_3 are random variables, $x^*_{i,t}$ is the global best solution that has been discovered (up until now), $x_{i,t}$ is the current solution, and $fw \in [0, 1]$ is the fitness weight of the scout bees.

If the current solution and the global best solution have the same fitness value, the pace is calculated as expressed in (6). In (7)-(9), $r_1 * \cos(r_2)$ or $r_1 * sin(r_2)$ guides the scout bees toward exploration or exploitation. If the value of $\cos(r_2)$ or $sin(r_2)$ is greater than 1 or less than -1; the scout bees explore the diversity of solutions. However, if the value of $r_1 * \cos(r_2)$ or $r_1 * sin(r_2)$ is in the $[-1, 1]$ range, the scout bees exploit the search solution.

In terms of mathematical complexity, the proposed SC-FDO has the same time complexity as the original FDO. For each iteration, the time complexity of the SC-FDO is $O(p * dim + p * COF)$, where dim is the optimization problem's dimension, $p$is the population size, and COF is the cost of the objective function. For all iterations, the space complexity of SC-FDO is $O(p * COF + p * pace(include\ sine\ cosine\ functions))$. Meanwhile, the original FDO's space complexity is $O(p * COF + p * pace)$ for all iterations. Another FDO's variant, the IFDO's space complexity is $O\big(p * COF + p * pace + (alignment * 1/cohesion)\big)$. Thus, the space complexity of the SC-FDO is slightly increased compared to FDO but lower than the IFDO's space complexity.

B. RANDOM WEIGHT FACTOR AND GLOBAL FITNESS WEIGHT STRATEGY

To further improve the search performance of the proposed SC-FDO, a random weight factor (wf) and global fitness weight parameter (fw^*) are embedded into the searching process. The proposed SC-FDO also incorporates an improved fitness weight (fw) calculation to increase the convergence and quality of the solutions. The calculation of the improved fitness weight (fw) follows this formula [25].

$$fw = \left\lfloor \frac{x^*_{i,tfitness}}{x_{i,tfitness}} \right\rfloor \tag{10}$$

where $x^*_{i,tfitness}$ is the fitness function of the global best solution and $x_{i,tfitness}$ is the fitness function of the current solution. The fw value is calculated according to the following equations:

$$\left\{\begin{array}{ll} fw_t > wf_t, & nfw_t = fw_t - wf_t \quad (11) \\ fw_t \leq wf_t, & nfw_t = fw_t(12) \quad (12) \end{array}\right.$$

where fw_t is the current fitness weight, nfw_t is the new fitness weight at the t^{th} iteration and wf_t is the current weight factor in the [0, 1] range. The work of [6] recommended that the values of weight factor parameter, wf in (2) be fine-tuned manually for each optimization problem. If wf is equal to 1, it represents a high level of convergence and a low chance of converge. If wf is equal to 0, the search is more stable, and it is not affecting the value of fitness weight (fw). However, this may cause bias with respect to unknown optimization problems.

Therefore, the proposed SC-FDO introduces a random weight factor (wf) that permits the wf value to be uniformly distributed across the scout bee population. To further optimize the random weight factor (wf), this study proposes a global fitness weight parameter (fw^*). The fw^* represents the value of fitness weight for the global best solution obtained so far by any search agents over all the iterations. The fw^* is used to fine-tuning the random weight factors (wf) adaptively during the search process. For example, if the current fitness weight is greater than the global fitness weight, then a new weight factor is generated. The mathematical calculation is according to the following (13).

$$fw^* < fw_t, \quad wf_t = wf_{t-1} * r_0 \tag{13}$$

where fw^* is the global fitness weight of the global best solution, wf_t is the current weight factor in the $[0, wf_{t-1}]$ range, t is the iteration and r_0 [0, 1] is the uniformly distributed random number. It implies that each iteration has a different weight factor parameter in the [0, 1] range. The values of random wf decreased from wf to 0 throughout iterations to obtain a stable search.

C. CONVERSION PARAMETER STRATEGY

In the modified pace-updating equation, parameter r_1, r_2 and r_3 convert search from exploration to exploitation at the promising areas. The parameter r_1, as expressed in (14) determines the region of the next solution. A large r_1 value encourages global exploration, meanwhile a smaller r_1 value encourages local exploitation towards the destination. To achieve a balanced exploration and exploitation, r_1 is linearly decreased from $a$to 0 and expressed as follow:

$$r_1(t) = a * \left(1 - \frac{t}{t_{max}}\right) \tag{14}$$

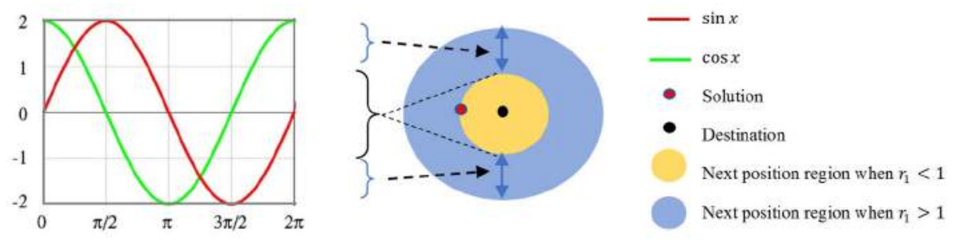


FIGURE 3. Sine and cosine in [−2, 2] range.

where t is the current iteration, t_{max} is the maximum iteration and a is a constant. In this study, the constant a has the same value as the several previous studies [4], [16], [32]–[34], in which a is equal to 2.

Furthermore, the parameter r_2 [0, 2π] in (7)-(9) defines the direction of the movement, either towards or outwards the destination and r_3 [0, 2] is the random weight of the global best solution ($x^*_{i,t}$) with the uniform probability distribution, either stochastically emphasize ($r_3 > 1$) or deemphasize ($r_3 < 1$) the impact of distance on the movement. In addition, the movement of the scout bees is defined as follow:

$$distance\ from\ bee\ scout = r_3 * x^*_{i,t} - x_{i,t} \tag{15}$$

The impacts of sine, cosine, and the parameters in (7)-(9) are presented in Fig. 3. If the value of r_1 is greater than 1, the solutions allow the search agents to explore the outside spaces between their corresponding destinations. Meanwhile, the sine and cosine functions enable a solution to be repositioned relative to another solution by exploiting the neighboring space if the value of r_1 is less than 1. Hence, the conversion parameter strategy is employed to enhance the scout bees' exploration and exploitation balancing.

D. BEST SOLUTION-UPDATING STRATEGY

Another improvement is the best solution-updating strategy used in the proposed SC-FDO. The existing FDO finds the best solution at the beginning of each iteration, consuming more computational time when searching for the global best solution. In contrast, the SC-FDO improves FDO by periodically updating the position around the global best solution (up until now) to obtain the best search region during exploration while exploiting and updating the global best solutions found by each iteration. Consequently, the search moves towards the global best solution over all previous iterations.

For example, if the current search agent position is superior to the previous position, the search agent will be updated with the current position as the global best solution. Hence, the SC-FDO takes less time to achieve better results than the original FDO and finally reduces the execution time of the proposed SC-FDO.

In conclusion, we introduce the modified pace-updating equation, the random weight factor (wf) and global fitness weight parameter (fw^*), the conversion parameter strategy, and the best solution-updating strategy in the proposed

```
Initialize the parameters and hive positions for scout bee population, $x_{i,t}$ (i = 1,2,3,…,n)
Calculate the objective function value for each scout bee and update global best solution, $x^*_{i,t}$
While (t <= $t_{max}$)
    Calculate the parameter $r_1$ using equation (14)
    For each scout bee, $x_{i,t}$
        Update the best scout bee, $x^*_{i,t}$
        Update the parameters $r_2$ and $r_3$
        If ($x_{i,t}$ fitness = 0) (avoid divide by zero)
            fitness weight = 0
        Else
            Calculate fitness weight, $fw_t$ using equation (10)
        End if
        Calculate random weight factor, $wf_t$ using equation (13)
        Update fitness weight, $fw_t$ using equations (11)-(12)
        Create random walk $r$ using Levy flight
        If (fitness weight = 1)
            Calculate pace using equation (6)
        Else if (fitness weight = 0)
            Calculate pace using equation (7)
        Else
            If (random number >=0)
                Calculate pace using equation (9)
            Else
                Calculate pace using equation (8)
            End if
        End if
        Calculate $x_{i,t+1}$ using equation (1)
        If ($x_{i,t+1}$ fitness < $x_{i,t}$ fitness)
            Move accepted and pace saved
        Else
            Update the parameters $r_3$
            Calculate pace using equation (15)
            If ($x_{i,t+1}$ fitness < $x_{i,t}$ fitness)
                Move accepted and pace saved
            Else
                Remain current position
            End if
        End if
        Update the global best solution
        Update the global fitness weight, $fw^*$
    End for
End while
```

FIGURE 4. The pseudocode of the proposed sine cosine-fitness dependent optimizer (SC-FDO).

SC-FDO algorithm. By integrating the strength of SCA to exploit the refine search area for the best solutions, the efficiency of the SC-FDO is improved. The pseudocode of

SC-FDO is presented in Fig. 4, while the flowchart of SC-FDO is illustrated in Fig. 5.

IV. NUMERICAL EXPERIMENT AND RESULTS

The proposed SC-FDO is implemented and evaluated over a group of 29 benchmark test functions, as listed in Table 2 [4], [6] and Table 3 [6], [36].

A. EVALUATION CRITERIA

The following measures are applied to access the results of the benchmark test functions.

1) STATISTICAL MEAN

The statistical mean is the average values of the optimal solution that are obtained by executing the optimization algorithm for N number of times, and it is computed according to (16).

$$mean = \frac{1}{N}\sum\nolimits_{i=1}^{N} A_i \tag{16}$$

where A_i is the optimal solution of the run time i.

2) STATISTICAL STANDARD DEVIATION (STD)

Statistical standard deviation (std) measures the differences of each optimal solution from the mean, as defined in (17). It computes the stability and robustness of the optimization algorithm.

$$std = \sqrt{\frac{1}{N-1}\sum\nolimits_{i=1}^{N} (A_i - mean)^2} \tag{17}$$

3) STATISTICAL MEAN EXECUTION TIME

Statistical mean execution time is the average computational time taken by the optimization algorithm executing each benchmark test function.

4) WILCOXON RANK SUM TEST

Wilcoxon rank-sum test is a non-parametric test for two independent groups [37], and it is used to assess whether the distributions of observations obtained between the proposed algorithm and benchmark algorithm are systematically different.

B. BENCHMARK TEST FUNCTIONS

The proposed SC-FDO was compared with six well-known nature-inspired algorithms, namely FDO [6], IFDO [25], SCA [4], WOA [8], PSO [1], and BOA [10]. The test functions for the benchmark can be categorized into unimodal functions (BF1-BF7), multimodal functions (BF8–BF13), and composite functions (BF14-BF19), listed in Table 2. The remaining BF20-BF29 test functions from CEC-C06 2019 [6], [36] are employed to evaluate the proposed SC-FDO further, as shown in Table 3.

For each benchmark test function, all the algorithms were tested with 30 runs. In the work of [4], [6], [25], a total of 30 search agents and a maximum number of 500 iterations were used in the experiments. Thus, the population size was fixed to 30, and the maximum number of iterations was 500. The experiments were conducted in a test environment, equipped with a Windows 10 operating system, an Intel (R) Core (TM) i7 processor with 16 GB RAM, and a programming tool of MATLAB R2018a.

The algorithm parameter settings are set the same as the original compared algorithms. For the parameter settings in the FDO [6], the wf parameter was equal to 0 for all the test functions except BF2 and BF6, in which wf was equal to 1. The other parameters settings are as the followings: SC-FDO: a = 2, wf [0, 1] and IFDO wf [0, 1].

Each algorithm was evaluated by three indexes: average value, standard deviation, and execution time. Tables 4, 5, and 6 show the comparison results in average values, standard deviation, and execution time of each algorithm for all the test functions.

C. COMPARISON OF SC-FDO WITH EXISTING OPTIMIZATION ALGORITHMS

This section evaluates the proposed SC-FDO with six existing nature-inspired algorithms, such as FDO, IFDO, SCA, WOA, PSO, and BOA.

As seen in Table 4, the proposed SC-FDO has the first rank as it outperformed well in 15 test functions compared to the other six optimization algorithms in BF1, BF2, BF3, BF4, BF5, BF8, BF9, BF10, BF11, BF14, BF20, BF21, BF22, BF25, and BF29. The IFDO, FDO, and WOA have the second, third, and fourth ranks, respectively in average value. However, the BOA recorded the lowest ranking in the performance comparison. The following is the rank of algorithms for generating values that are close to the theoretical optimal average values: (1) SC-FDO (2) IFDO (3) FDO (4) SCA (5) WOA (6) PSO (7) BOA.

For the evaluation of exploitation (BF1-BF7), the results indicated that the proposed SC-FDO achieved the theoretically optimal average values of 0 in the test functions: BF1, BF2, BF3, and BF4. The study proved that the proposed SC-FDO is effective in exploitation and convergence because it has high searching precision of unimodal test functions in BF1, BF2, BF3, BF4, and BF5 test functions than the IFDO, FDO, WOA, SCA, PSO, and BOA. Hence, the modified pace-updating strategy is beneficial for enhancing the existing FDO and IFDO's exploitation ability and subsequently improved the exploitation and convergence speed of the proposed SC-FDO.

For the evaluation of exploration (BF8-BF19), the proposed SC-FDO outperformed the other six optimization algorithms in most test functions (BF8, BF9, BF10, BF11, and BF14). Specifically, the proposed SC-FDO could obtain the theoretically optimal average value of 0 for the BF9 and BF11 test functions. In addition, the SC-FDO has comparative results with the other algorithms in BF15, BF16, BF17, BF18, and BF19. The results also evinced that the proposed SC-FDO has significantly improved the original FDO and IFDO. Hence, the modified pace-updating and conversion parameters enhancements greatly eliminated local

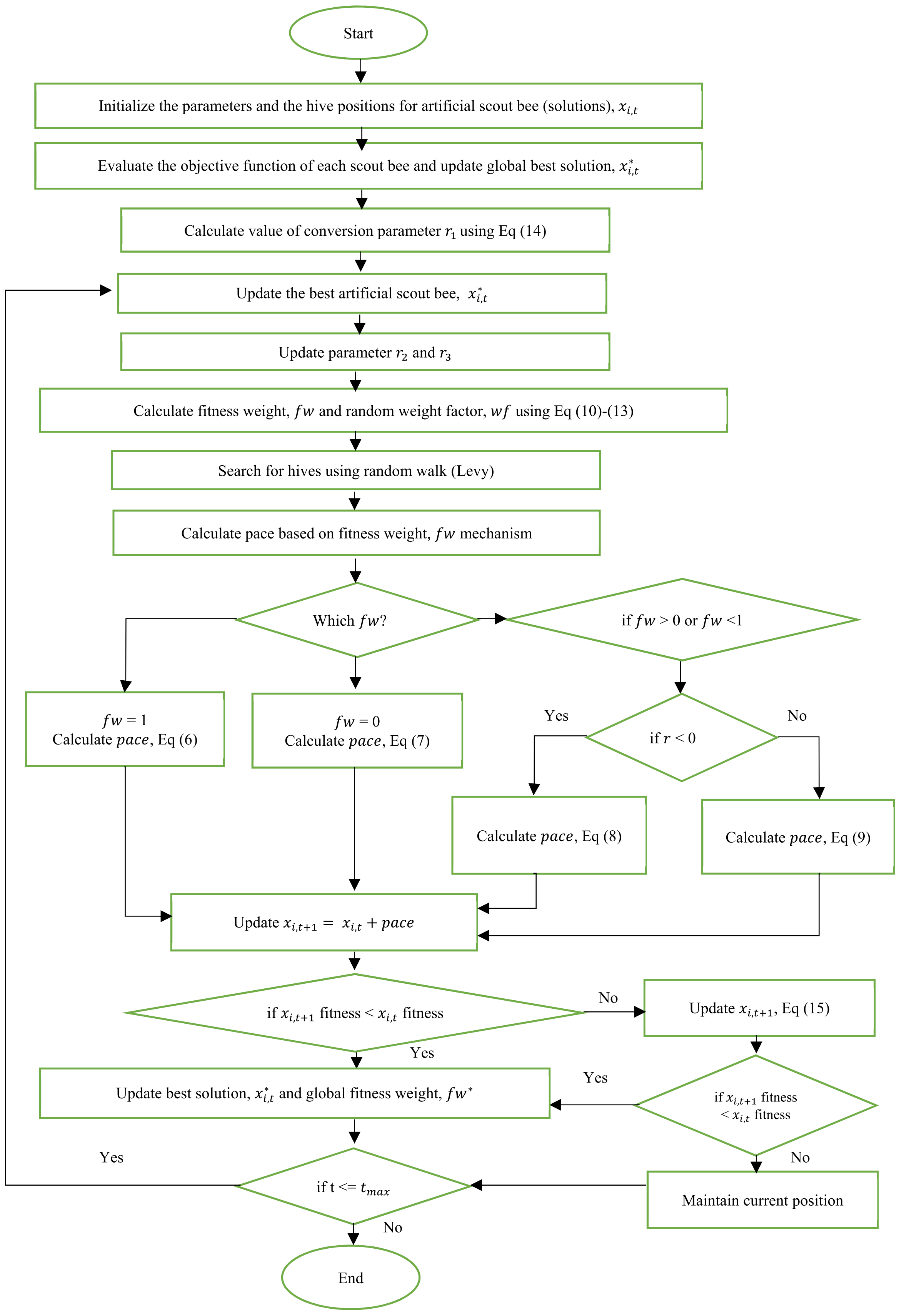

Start
Initialize the parameters and the hive positions for artificial scout bee (solutions), $x_{i,t}$
Evaluate the objective function of each scout bee and update global best solution, $x^*_{i,t}$
Calculate value of conversion parameter r_1 using Eq (14)
Update the best artificial scout bee, $x^*_{i,t}$
Update parameter r_2 and r_3
Calculate fitness weight, fw and random weight factor, wf using Eq (10)-(13)
Search for hives using random walk (Levy)
Calculate pace based on fitness weight, fw mechanism
Which fw?
if $fw > 0$ or $fw < 1$
$fw = 1$ Calculate $pace$, Eq (6)
$fw = 0$ Calculate $pace$, Eq (7)
Yes
if $r < 0$
No
Calculate $pace$, Eq (8)
Calculate $pace$, Eq (9)
Update $x_{i,t+1} = x_{i,t} + pace$
if $x_{i,t+1}$ fitness < $x_{i,t}$ fitness
No
Update $x_{i,t+1}$, Eq (15)
Yes
Update best solution, $x^*_{i,t}$ and global fitness weight, fw^*
Yes
if $x_{i,t+1}$ fitness < $x_{i,t}$ fitness
No
Maintain current position
Yes
if t <= t_{max}
No
End

FIGURE 5. The flowchart of the proposed SC-FDO.

TABLE 2. Benchmark test function [4], [6].

Function	Dim	Range	Shift position	f_{min}
$BF1(x) = \sum_{i=1}^{n} x_1^2$	10	[-100, 100]	[-30, -30, … -30]	0
$BF2\ (x) = \sum_{i=1}^{n} \lvert x_i \rvert + \prod_{i=1}^{n} \lvert x_i \rvert$	10	[-10,10]	[-3, -3, … -3]	0
$BF3\ (x) = \sum_{i=1}^{N} \left(\sum_{j-1}^{i} x_j \right)^2$	10	[-100, 100]	[-30, -30, … -30]	0
$BF4 = max_i\{\lvert x_i \rvert, 1 \le i\ \le n\}$	10	[-100, 100]	[-30, -30, … -30]	0
$BF5 = \sum_{i=1}^{n-1} [100(x_{i+1} - x_1^2)^2 +\ (x_i - 1)^2]$	30	[-30,30]	[-15, -15, … -15]	0
$BF6 = \sum_{i=1}^{n} ([x_i + 0.5])^2$	10	[-100, 100]	[-750, … -750]	0
$BF7 = \sum_{i=1}^{n} ix_1^4 + random\ [0,1]$	10	[-1.28,1.28]	[-0.25, …-0.25]	0
$BF8 = \sum_{i=1}^{n} -x_i^2\ sin\left(\sqrt{\lvert x_i \rvert}\right)$	10	[-500, 500]	[-300, … -300]	-418.9829
$BF9 = \sum_{i=1}^{n} [x_i^2 - 10\cos(2\pi x_i) + 10]$	10	[-5.12,5.12]	[-2, -2, …-2]	0
$BF10 = -20\exp\left(-0.2\sqrt{\sum_{i=1}^{n} x_i^2}\right) - \exp\left(\frac{1}{n}\sum_{i=1}^{n} cos(2\pi x_i)\right) + 20 + e$	10	[-32, 32]		0
$BF11 = \frac{1}{4000}\sum_{i=1}^{n} x_i^2 - \prod_{i=1}^{n} cos\left(\frac{x_i}{\sqrt{i}}\right) + 1$	10	[-600, 600]	[-400, … -400]	0
$BF12 = \frac{\pi}{n}\left\{10\,sin(\pi y_1) + \sum_{i=1}^{n-1} (y_i - 1)^2\ [1 + 10\ sin^2(\pi y_{i+1})] +\ (y_n - 1)^2\right\} + \sum_{i=1}^{n} u(x_i, 10,100,4)$ $y_i\ = 1 + \frac{x+1}{4}\ ,\ \ u(x_i, a, k, m) = \begin{cases} k(x_i - a)^m\ x_i > a \\ 0 - a < x_i < a \\ k(-x_i - a)^m\ x_i < -a \end{cases}$	10	[-50,50]	[-30, 30, … 30]	0
$BF13 = 0.1\left\{sin^2(3\pi x_1) + \sum_{i=1}^{n} (x_i - 1)^2[1 + sin^2(3\pi x_i + 1)] + (x_n - 1)^2[1 + sin^2(2\pi x_n)]\right\} + \sum_{i=1}^{n} u(x_i, 5,100,4)$	10	[-50,50]	[-100, … -100]	0
$BF14(CF1)$: $f1, f2, f3, \ldots, f10\ =$ Sphere function $\delta 1, \delta 2, \delta 3 \ldots \delta 10 = [1,1,1, \ldots, 1]$ $\lambda 1, \lambda 2, \lambda 3 \ldots \lambda 10 = [\frac{5}{100}, \frac{5}{100}, \frac{5}{100}, \ldots, \frac{5}{100}]$	10	[-5, 5]		0

TABLE 2. ***(Continued.)*** **Benchmark test function [4], [6].**

$BF15(CF2)$: $f1, f2, f3 \ldots f10 = Griewank's\ function$ $\delta1, \delta2, \delta3 \ldots \delta10 = [1,1,1, \ldots,1]$ $\lambda1, \lambda2, \lambda3 \ldots \lambda10 = [\frac{5}{100}, \frac{5}{100}, \frac{5}{100}, \ldots, \frac{5}{100}]$	10	[-5, 5]	0
$BF16(CF3)$: $f1, f2, f3 \ldots f10 = Griewank's\ function$ $\delta1, \delta2, \delta3 \ldots \delta10 = [1,1,1, \ldots,1]$ $\lambda1, \lambda2, \lambda3 \ldots \lambda10 = [1,1,1, \ldots,1]$	10	[-5, 5]	0
$BF17(CF4)$: $f1, f2 = Ackley's\ function$ $f3, f4 = Rastrigin's\ function$ $f5, f6 = Weierstrass\ function$ $f7, f8 = Griewank's\ function$ $f9, f10 = Sphere\ function$ $\delta1, \delta2, \delta3 \ldots \delta10 = [1,1,1, \ldots,1]$ $\lambda1, \lambda2, \lambda3 \ldots \lambda10 = [\frac{5}{32}, \frac{5}{32}, 1, 1, \frac{5}{0.5}, \frac{5}{0.5}, \frac{5}{100}, \frac{5}{100}, \frac{5}{100}, \frac{5}{100}]$	10	[-5, 5]	0
$BF18(CF5)$: $f1, f2 = Rastrigin's\ function$ $f3, f4 = Weierstrass\ function$ $f5, f6 = Griewank's\ function$ $f7, f8 = Ackley's\ function$ $f9, f10 = Sphere\ function$ $\delta1, \delta2, \delta3 \ldots \delta10 = [1,1,1, \ldots,1]$ $\lambda1, \lambda2, \lambda3 \ldots \lambda10 = [\frac{1}{5}, \frac{1}{5}, \frac{5}{0.5}, \frac{5}{0.5}, \frac{5}{100}, \frac{5}{100}, \frac{5}{32}, \frac{5}{32}, \frac{5}{100}, \frac{5}{100}]$	10	[-5, 5]	0
$BF19(CF6)$: $f1, f2 = Rastrigin's\ function$ $f3, f4 = Weierstrass\ function$ $f5, f6 = Griewank's\ function$ $f7, f8 = Ackley's\ function$ $f9, f10 = Sphere\ function$ $\delta1, \delta2, \delta3 \ldots \delta10 = [0.1,0.2,0.3,0.4,0.5,0.6,0.7,0.8,0.9,1]$ $\lambda1, \lambda2, \lambda3 \ldots \lambda10 = [0.1 * \frac{1}{5}, 0.2 * \frac{1}{5}, 0.3 * \frac{5}{0.5}, 0.4 * \frac{5}{0.5}, 0.5 * \frac{5}{100}, 0.6 * \frac{5}{100}, 0.7 * \frac{5}{32}, 0.8 * \frac{5}{32}, 0.9 * \frac{5}{100}, 1 * \frac{5}{100}]$	10	[-5, 5]	0

TABLE 3. **The 100-digit challenge: CEC-06 2019 benchmark [6], [36].**

Function	Dim	Range	f_{min}
$BF20$: $Storn's\ Chebyshev\ polynomial\ fitting\ problem$	9	[-8192, 8192]	1
$BF21$: $Inverse\ Hilbert\ matrix\ problem$	16	[-16384, 16384]	1
$BF22$: $Lennard-Jones\ minimum\ energy\ cluster$	18	[-4,4]	1
$BF23$: $Rastrigin's\ function$	10	[-100, 100]	1
$BF24$: $Griewangk's function$	10	[-100, 100]	1
$BF25$: $Weierstrass\ function$	10	[-100, 100]	1
$BF26$: $Modified\ Schwefel's function$	10	[-100, 100]	1
$BF27$: $Expanded\ Schaffer's\ F6\ function$	10	[-100, 100]	1
$BF28$: $Happy\ Cat\ function$	10	[-100, 100]	1
$BF29$: $Ackley\ function$	10	[-100, 100]	1

TABLE 4. The average values of the proposed SC-FDO and other algorithms for the 29 benchmark test functions.

Function	SC-FDO	FDO	IFDO	SCA	WOA	PSO	BOA
BF1	**0.00E+00**	1.87E-34	4.60E-27	2.05E-12	1.06E-76	5.65E-10	1.01E-17
BF2	**0.00E+00**	5.52E-04	1.33E-05	1.58E-09	1.79E-52	1.00E+00	9.58E-11
BF3	**0.00E+00**	5.28E-14	7.98E-07	1.30E-01	1.18E+02	3.34E+02	7.67E-19
BF4	**0.00E+00**	1.39E-12	9.05E-13	2.85E-03	2.87E+00	1.08E-02	6.48E-11
BF5	**2.69E+01**	4.85E+01	6.37E+01	4.32E+04	2.80E+01	3.15E+04	2.89E+01
BF6	2.02E-02	1.06E-06	**1.33E-16**	4.45E-01	1.04E-03	3.76E-10	1.15E+00
BF7	4.49E-02	5.96E-01	5.41E-01	2.16E-03	2.44E-03	1.04E-02	**1.85E-03**
BF8	**-1.02E+04**	-2.93E+03	-7.02E+03	-2.14E+03	-3.35E+03	-3.29E+03	-1.97E+03
BF9	**0.00E+00**	2.45E+00	2.17E+00	1.35E+00	1.40E+00	9.90E+00	3.85E+01
BF10	**3.26E-15**	2.20E-14	6.57E-15	8.80E-02	4.80E-15	6.64E-01	8.40E-11
BF11	**0.00E+00**	7.81E-02	6.70E-02	7.05E-02	4.95E-02	1.04E-01	1.59E-02
BF12	4.54E-02	1.04E-02	3.88E+00	1.07E-01	6.23E-03	**4.35E-08**	1.98E-01
BF13	1.73E-01	3.67E-03	1.83E+00	3.24E-01	3.85E-02	**2.28E-09**	5.56E-01
BF14	**6.36E+01**	7.00E+01	**6.80E+01**	1.32E+02	1.17E+02	1.76E+02	2.37E+02
BF15	2.28E+02	1.41E+02	1.58E+02	**1.24E+02**	2.00E+02	2.24E+02	3.17E+02
BF16	3.55E+02	**2.00E+02**	2.51E+02	3.91E+02	5.05E+02	3.09E+02	5.61E+02
BF17	5.17E+02	4.13E+02	**3.75E+02**	4.53E+02	5.73E+02	5.42E+02	7.31E+02
BF18	1.56E+02	1.01E+02	**6.46E+01**	1.33E+02	2.29E+02	2.08E+02	2.36E+02
BF19	7.09E+02	8.05E+02	7.95E+02	**5.67E+02**	8.04E+02	8.24E+02	8.55E+02
BF20	**4.25E+04**	6.69E+07	4.99E+08	6.41E+09	3.16E+10	3.22E+10	1.37E+05
BF21	**1.73E+01**	**1.73E+01**	**1.73E+01**	1.75E+01	1.74E+01	**1.73E+01**	1.85E+01
BF22	**1.27E+01**	**1.27E+01**	**1.27E+01**	**1.27E+01**	**1.27E+01**	**1.27E+01**	**1.27E+01**
BF23	1.59E+03	3.10E+01	**2.83E+01**	1.63E+03	3.69E+02	2.02E+03	5.81E+03
BF24	1.69E+00	1.13E+00	**1.10E+00**	2.23E+00	1.99E+00	1.79E+00	2.87E+00
BF25	**8.27E+00**	8.96E+00	8.89E+00	1.12E+01	9.82E+00	9.32E+00	1.14E+01
BF26	5.93E+01	4.74E+01	**3.94E+01**	8.55E+02	6.46E+02	4.28E+02	1.18E+03
BF27	4.50E+00	4.22E+00	**4.17E+00**	6.17E+00	5.87E+00	5.96E+00	6.58E+00
BF28	4.85E+00	**2.41E+00**	2.48E+00	1.15E+02	5.25E+00	1.85E+02	8.98E+02
BF29	**1.81E+01**	2.00E+01	1.82E+01	2.04E+01	2.03E+01	2.03E+01	2.05E+01
Rank	**1**	**3**	**2**	**5**	**4**	**6**	**7**

optima problems and optimized the balance of exploitation and exploration in the proposed SC-FDO.

For CEC-C06 2019 (BF20-BF29) evaluation, the proposed SC-FDO performed better than the other algorithms in BF20, BF21, BF22, BF25, and BF29 tests functions. The results also revealed that the proposed SC-FDO has significantly improved the original FDO, in which the proposed improvements in SC-FDO have successfully enhanced the ability to avoid local optima and converge towards the global optimum during optimization.

Furthermore, Table 5 indicates that the SC-FDO topped the standard deviation ranking among all the optimization algorithms. The IFDO and FDO shared the second-ranking, followed by WOA and SCA. The SC-FDO outperformed well in 15 functions (BF1, BF2, BF3, BF4, BF5, BF9, BF10, BF11, BF14, BF20, BF21, BF22, BF25, BF26 and BF29). For BF1, BF2, BF3, BF4, BF9, and BF11 test functions, the SC-FDO achieved the theoretical optimal standard deviation, in which the values were 0. In addition, the standard deviation values of the proposed SC-FDO on most test functions are within small ranges and ranked first in standard deviation, indicating that the SC-FDO algorithm has better stability and is able to search optimal solutions in a smaller range than the original FDO and IFDO. The reason is that the adaptation of the global fitness weight (fw^*), random weight factor (wf) and conversion parameter strategies, which balance exploration and exploitation of the search space, have led to a convergence on the global optimum. However, the PSO and BOA did not perform well in standard deviation. The BSO is at the bottom of the ranking, while PSO is the lowest standard deviation in the test cases.

For all the test functions, the average execution time used by each algorithm over 30 runs is shown in Table 6. The PSO has the minimum execution time, followed by the WOA, BOA, SCA, SC-FDO, FDO and IFDO. The average execution time used by the SC-FDO is higher than the PSO,

TABLE 5. The average standard deviation of the proposed SC-FDO and other algorithms for the 29 benchmark test functions.

Function	SC-FDO	FDO	IFDO	SCA	WOA	PSO	BOA
BF1	**0.00E+00**	6.94E-34	1.82E-26	5.87E-12	3.13E-76	1.93E-09	1.85E-17
BF2	**0.00E+00**	7.32E-04	6.44E-05	2.72E-09	9.52E-52	3.16E+00	1.03E-10
BF3	**0.00E+00**	1.41E-13	4.36E-06	5.60E-01	1.61E+02	1.29E+03	1.58E-18
BF4	**0.00E+00**	3.95E-12	2.53E-12	1.05E-02	4.81E+00	1.26E-02	7.00E-11
BF5	**2.69E+01**	5.77E+01	7.89E+01	1.03E+05	2.80E+01	5.15E+04	2.89E+01
BF6	5.53E-02	1.59E-06	**7.29E-16**	4.75E-01	1.35E-03	5.97E-10	1.21E+00
BF7	5.98E-02	6.85E-01	6.07E-01	2.72E-03	3.66E-03	1.14E-02	**2.04E-03**
BF8	2.90E+04	2.97E+03	8.75E+03	2.15E+03	3.40E+03	3.30E+03	**1.97E+03**
BF9	**0.00E+00**	2.97E+00	2.39E+00	5.29E+00	5.66E+00	1.29E+01	4.43E+01
BF10	**3.66E-15**	7.99E-14	6.80E-15	4.82E-01	5.32E-15	3.64E+00	8.92E-11
BF11	**0.00E+00**	9.14E-02	7.25E-02	1.56E-01	1.47E-01	1.22E-01	8.72E-02
BF12	5.61E-02	5.68E-02	5.25E+00	1.16E-01	1.12E-02	**2.32E-07**	2.26E-01
BF13	1.90E-01	1.82E-02	3.99E+00	3.34E-01	6.43E-02	**1.04E-08**	5.84E-01
BF14	**9.01E+01**	1.05E+02	1.17E+02	1.35E+02	1.71E+02	1.95E+02	2.55E+02
BF15	2.37E+02	1.73E+02	1.95E+02	**1.33E+02**	2.18E+02	2.47E+02	3.41E+02
BF16	3.61E+02	**2.34E+02**	2.72E+02	3.96E+02	5.29E+02	3.36E+02	5.70E+02
BF17	5.25E+02	4.31E+02	**3.95E+02**	4.55E+02	5.86E+02	5.69E+02	7.41E+02
BF18	1.68E+02	1.40E+02	**1.17E+02**	1.46E+02	2.77E+02	2.52E+02	2.76E+02
BF19	7.31E+02	8.23E+02	8.15E+02	**5.79E+02**	8.19E+02	8.38E+02	8.61E+02
BF20	**4.26E+04**	9.12E+07	7.16E+08	1.18E+10	4.93E+10	6.06E+10	1.48E+05
BF21	**1.73E+01**	**1.73E+01**	**1.73E+01**	1.75E+01	1.74E+01	**1.73E+01**	1.85E+01
BF22	**1.27E+01**	**1.27E+01**	**1.27E+01**	**1.27E+01**	**1.27E+01**	**1.27E+01**	**1.27E+01**
BF23	2.05E+03	3.28E+01	**2.98E+01**	1.74E+03	4.24E+02	3.00E+03	6.61E+03
BF24	1.71E+00	1.13E+00	**1.11E+00**	2.23E+00	2.03E+00	1.90E+00	2.92E+00
BF25	**8.33E+00**	8.99E+00	8.94E+00	1.13E+01	9.87E+00	9.44E+00	1.14E+01
BF26	**8.99E+01**	1.02E+02	1.01E+02	8.75E+02	7.28E+02	4.85E+02	1.19E+03
BF27	4.53E+00	4.26E+00	**4.21E+00**	6.18E+00	5.89E+00	5.98E+00	6.59E+00
BF28	4.92E+00	**2.41E+00**	2.48E+00	1.48E+02	6.09E+00	5.37E+02	9.81E+02
BF29	**1.84E+01**	2.00E+01	1.91E+01	2.04E+01	2.03E+01	2.03E+01	2.05E+01
Rank	**1**	**2.5**	**2.5**	**5**	**4**	**6**	**7**

WOA, BOA, and SCA; however, it revealed that the proposed SC-FDO has lower average execution time than the original FDO and IFDO. Although the space complexity of the SC-FDO is slightly higher than the original FDO's space complexity, the introduction of the best solution-updating approach has significantly decreased the computing time of the proposed SC-FDO. Specifically, there is a significant reduction of the average execution time in the SC-FDO, approximately 87% and 89% of the original FDO and IFDO, respectively. Hence, the findings proved that the proposed SC-FDO has successfully reduced the original FDO and IFDO's computational time and substantially enhanced the efficiency of the original FDO and IFDO.

Furthermore, a comparison between the convergence curve of the SC-FDO and other algorithms on twelve representative test functions is presented in Fig. 6. For the BF1, BF2, BF3, BF4, BF5, BF9, BF11, BF20, and BF25 test functions, the SC-FDO converged faster than the other algorithms, consequently reduced the exploration and exploitation time when finding the optimal global solution. This finding shows that the effect of the proposed conversion parameter and adaptive sine and cosine functions can significantly optimize the exploration and exploitation ability of the SC-FDO.

However, the SC-FDO converged less quickly than other algorithms at the beginning of the iteration for BF10, BF14, and BF29 in Fig. 6. Interestingly, the SC-FDO significantly increased the convergence rate and accuracy as the iterations approached 80, 466 and 220 iterations. The SC-FDO eventually converged closer to the optimal global solution. Therefore, this study concluded that the SC-FDO achieved better results than the existing FDO, IFDO, SCA, WOA, PSO, and BOA in terms of convergence precision and speed.

Therefore, it can be concluded that the proposed SC-FDO is superior to the other optimization algorithms as it ranked first among all the compared optimization algorithms in terms of average values and standard deviation for the benchmark test function.

TABLE 6. The average execution time in seconds obtained by the proposed SC-FDO and other algorithms for the 29 benchmark test functions.

Function	SC-FDO	FDO	IFDO	SCA	WOA	PSO	BOA
BF1	0.8153	1.3415	51.5708	0.0318	0.0233	**0.0174**	0.0236
BF2	0.7554	1.5623	51.7337	0.0332	0.0247	**0.0189**	0.0254
BF3	1.0626	3.5154	56.0892	0.0610	0.0524	**0.0466**	0.0533
BF4	0.8266	1.4799	52.8787	0.0321	0.0232	**0.0176**	0.0242
BF5	2.1478	3.3919	454.5471	0.0722	0.0414	0.0419	**0.0365**
BF6	0.8502	1.4505	52.1921	0.0331	0.0234	**0.0177**	0.0237
BF7	0.9941	3.1662	54.9211	0.0520	0.0433	**0.0372**	0.0433
BF8	1.1244	2.2013	53.4520	0.0407	0.0318	**0.0259**	0.0325
BF9	0.8425	1.7409	54.0513	0.0342	0.0257	**0.0211**	0.0271
BF10	0.8580	1.6959	52.1359	0.0372	0.0274	**0.0227**	0.0281
BF11	0.9524	2.4860	53.0158	0.0454	0.0353	**0.0313**	0.0382
BF12	1.8225	7.1087	57.6510	0.1116	0.1029	**0.0957**	0.1047
BF13	1.8167	7.1394	57.4141	0.1118	0.1026	**0.0955**	0.1049
BF14	216.1311	1659.1242	1709.7733	24.4809	**24.4376**	24.5960	24.5465
BF15	219.1252	1696.5576	1738.1238	24.8657	24.7778	**24.4432**	24.8073
BF16	218.9035	1507.4357	1744.7410	24.6945	**24.6758**	24.8351	24.7862
BF17	239.4499	1913.1311	1897.7324	27.1037	**26.9636**	27.2118	27.0615
BF18	239.5473	1958.1535	1893.1001	27.1184	**26.9986**	27.2815	27.0858
BF19	236.5180	1880.2623	1901.5558	**27.1572**	27.2272	27.2814	27.1841
BF20	39.2986	233.2854	270.2622	3.2628	**3.2461**	3.2560	3.2514
BF21	1.2271	1.7830	160.5939	0.0409	0.0257	**0.0226**	0.0233
BF22	1.6639	3.6787	200.4955	0.0687	0.0513	0.0489	**0.0482**
BF23	1.0426	2.2694	53.1968	0.0450	0.0360	**0.0284**	0.0335
BF24	1.0790	2.3763	64.5387	0.0466	0.0372	**0.0306**	0.0348
BF25	12.8990	73.8102	124.4422	1.0471	1.0345	1.0320	**1.0290**
BF26	1.0419	2.3825	53.2395	0.0445	0.0364	**0.0295**	0.0347
BF27	1.0525	2.4033	53.2941	0.0465	0.0374	**0.0303**	0.0358
BF28	0.9737	2.0934	52.2668	0.0404	0.0299	**0.0244**	0.0310
BF29	1.0766	2.5715	53.5268	0.0486	0.0384	**0.0329**	0.0388
Rank	5	6	7	4	2	1	3

** Bold font denotes the best result.

D. WILCOXON RANK SUM TEST

In the Wilcoxon rank-sum test, it is assumed that there is no difference among the compared algorithms in the null hypothesis, Ho. The alternative hypothesis, H1 assumes that there is a difference between the compared algorithms for the average values of the test functions in Table 4. The Wilcoxon rank-sum tests indicated that the null hypothesis Ho is rejected, and the results of SC-FDO are different from those compared algorithms, at the 0.05 significance level. Therefore, the SC-FDO results are statistically significant compared with the benchmark algorithms, as presented in Table 7.

V. SC-FDO BASED MULTILAYER PERCEPTRON TRAINER

The SC-FDO is employed as a trainer to train and optimize multilayer perceptron (MLP) network, abbreviate as SC-FDO based MLP trainer.

Fig. 7 shows the structure of the multilayer perceptron (MLP) neural network. This network is also called a feedforward neural network (FFNN). It is the most frequently applied learning technique in MLP due to its stability and ease of use [38], [39].

The weighted sums of inputs are computed according to (18).

$$p_j = \sum\nolimits_{i=1}^{n} \left(w_{ij} * x_i\right) + b_j, \quad \text{j} = 1, 2, \ldots, \text{h} \tag{18}$$

where n is the number of the input nodes, w_{ij} is the connection weight from the i^{th} input node to the j^{th} hidden node, x_i is the ith input and b_j is the bias of the j^{th} hidden node.

The output of each hidden layer is defined as follows:

$$P_j = tansig\left(p_j\right) = \frac{2}{\left(1 + \exp\left(-2 * p_j\right)\right) - 1}, \quad \text{j} = 1, 2, \ldots, \text{h} \tag{19}$$

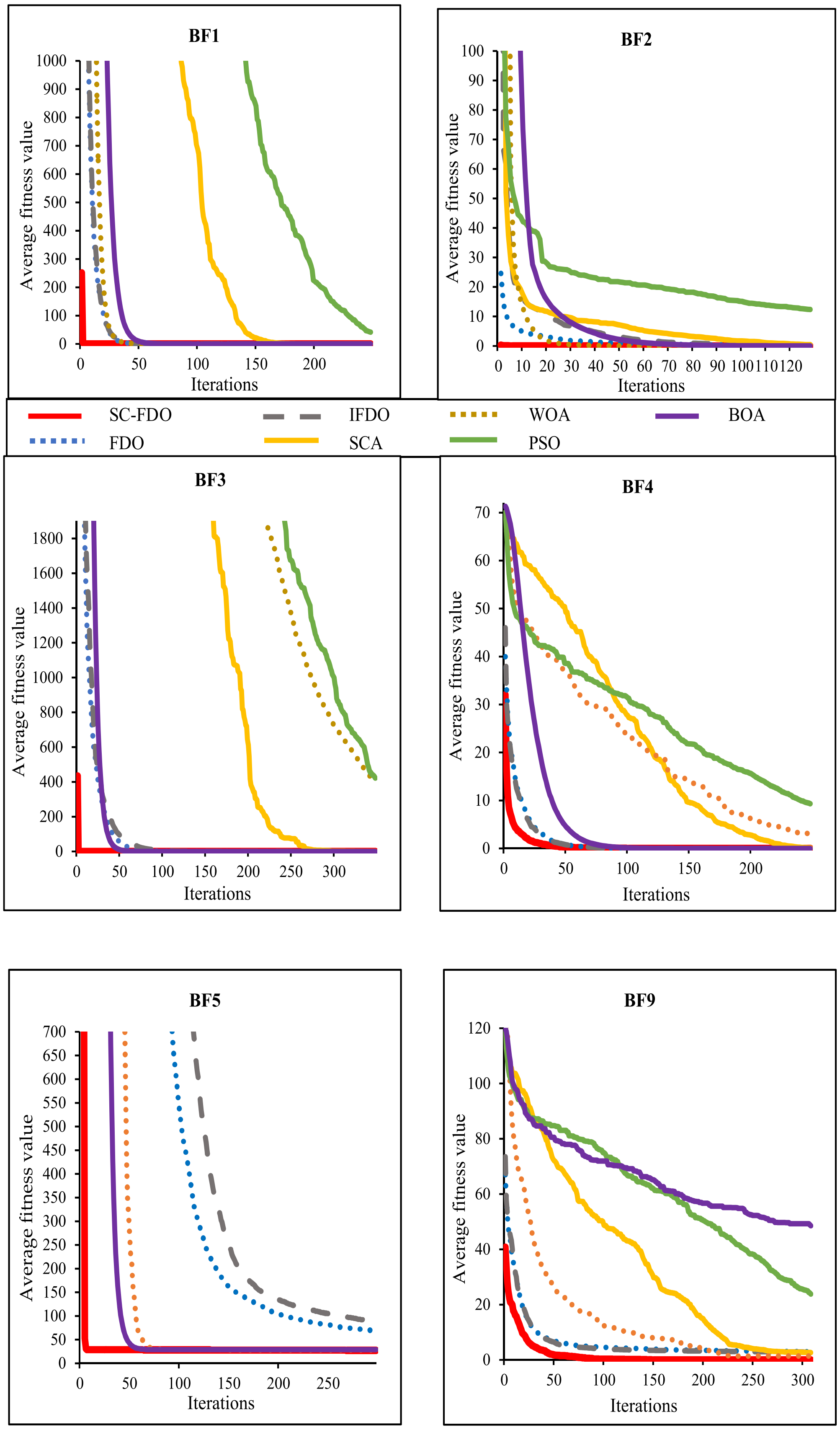


FIGURE 6. Convergence curves of the SC-FDO and other algorithms on ten representative test functions.

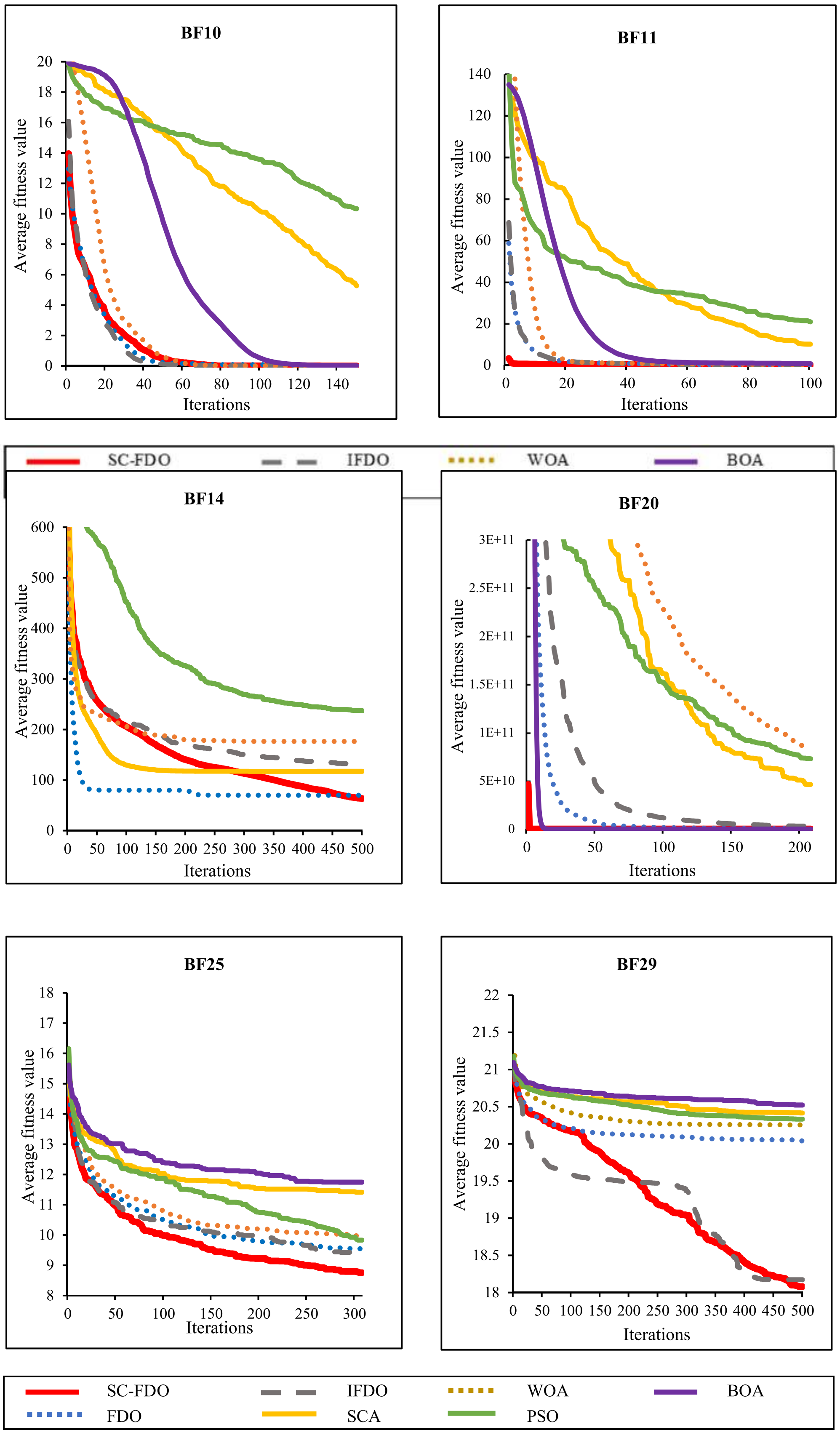


FIGURE 6. ***(Continued.)*** **Convergence curves of the SC-FDO and other algorithms on ten representative test functions.**

TABLE 7. The P-value of Wilcoxon rank sum test between the proposed SC-FDO and other algorithms.

Function	SC-FDO vs.					
	FDO	IFDO	SCA	WOA	PSO	BOA
BF1	7.69E-12	7.69E-12	7.69E-12	7.69E-12	7.68E-12	7.69E-12
BF2	7.687e-12	7.687e-12	7.69E-12	7.69E-12	7.69E-12	7.68E-12
BF3	1.54E-11	1.54E-11	1.54E-11	1.54E-11	1.54E-11	1.54E-11
BF4	1.21E-12	1.21E-12	1.21E-12	1.21E-12	1.21E-12	1.21E-12
BF5	0.1453	9.05E-02	3.02E-11	4.222e-9,	3.02E-11	3.02E-11
BF6	3.02E-11	3.02E-11	3.34E-11	1.34E-05	3.02E-11	3.02E-11
BF7	7.39E-11	4.97E-11	1.09E-10	2.15E-10	1.29E-06	4.08E-11
BF8	8.24E-02	4.13E-02	8.99E-11	9.23E-01	9.94E-01	4.08E-11
BF9	7.69E-12	7.52E-12	7.69E-12	4.32E-13	7.69E-12	7.69E-12
BF10	8.19E-08	7.95E-08	1.29E-11	1.29E-01	1.29E-11	1.29E-11
BF11	7.69E-12	7.68E-12	7.69E-12	1.69E-12	7.69E-12	7.68E-12
BF12	5.55E-10	6.05E-07	3.52E-07	3.20E-09	3.02E-11	3.83E-09
BF13	5.49E-11	5.49E-01	2.57E-07	2.57E-07	3.02E-11	1.78E-10
BF14	0.09202	2.15E-02	3.56E-04	2.34E-01	1.71E-05	1.29E-09
BF15	1.67E-04	2.26E-03	2.20E-08	2.12E-01	6.31E-01	4.43E-03
BF16	2.01E-07	5.46E-06	1.19E-01	1.33E-04	3.15E-02	8.99E-11
BF17	2.14E-03	1.17E-05	7.20E-05	1.12E-01	9.12E-01	9.75E-08
BF18	4.46E-04	3.32E-06	8.32E-03	3.40E-01	1.62E-01	4.68E-02
BF19	1.60E-03	2.55E-03	6.92E-03	4.38E-03	4.76E-06	1.53E-03
BF20	3.02E-11	3.02E-11	3.02E-11	3.02E-11	3.02E-11	3.02E-11
BF21	7.58E-12	5.21E-12	3.02E-11	6.72E-10	1.21E-12	3.02E-11
BF22	1.19E-12	1.19E-12	2.99E-11	6.68E-01	8.64E-02	2.99E-11
BF23	3.02E-11	3.02E-11	3.56E-01	8.15E-05	9.35E-01	3.65E-08
BF24	3.34E-11	3.02E-11	1.17E-09	2.16E-03	9.94E-01	3.34E-11
BF25	7.29E-03	1.63E-02	3.02E-11	4.42E-06	5.56E-04	7.39E-11
BF26	4.46E-01	6.95E-01	3.02E-11	3.34E-11	7.38E-10	3.02E-11
BF27	6.15E-02	1.91E-02	3.69E-11	1.33E-10	1.46E-10	3.02E-11
BF28	3.02E-11	3.02E-11	3.02E-11	6.41E-01	1.11E-06	3.02E-11
BF29	2.06E-01	1.15E-01	2.37E-10	1.20E-08	4.20E-10	3.02E-11

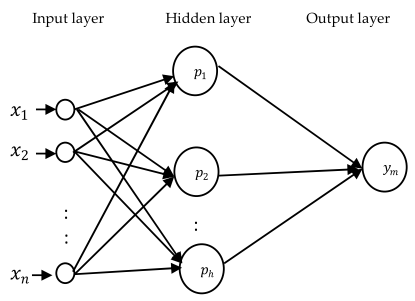


FIGURE 7. Multilayer perceptron (MLP) neural network.

The final output of the output layer is calculated using (20).

$$y_k = \frac{2}{\left(1+\exp\left(-2*\left(\sum_{j=1}^{h}\left(w_{jk}*p_j\right)+b_k\right)\right)\right)-1},\quad k=1,2,3,\ldots,m \tag{20}$$

where y_k is the k^{th} output, w_{jk} is the connection weight from the j^{th} hidden node to the k^{th} output node and b_k is the bias of the k^{th} output node.

Furthermore, the SC-FDO based MLP trainer will optimize the neural network with a set of optimal values for the weights and biases as described in (18)-(20). In the SC-FDO trainer, each variable indicates the total of weights and biases and defined as follows [39]:

$$v=\{W,b\}=\left\{w_{1,1},w_{1,2},\ldots,w_{n,h},\ b_1,\ldots,b_h,\ w_1,w_{h,}b_i\right\} \tag{21}$$

All the weights and biases variables need to converge until the optimum solution is reached that provides the highest prediction accuracy. The evaluation metric of the neural network is the mean square error (MSE), as indicated in (22).

$$MSE=\frac{1}{N}\sum_{i=1}^{N}(y-\tilde{y})^2 \tag{22}$$

where N is the number of outputs,y is the actual value, and $\tilde{y}$ is the predicted value by the SC-FDO based MLP trainer.

A. CASE STUDY: MISSING WEATHER DATA IMPUTATION

This section further verifies the performance of the proposed SC-FDO based MLP trainer by solving real-world application problems. Rainfall data are essential components of the hydrological cycle to assess flood risk [40] and predict rainfall forecasting [41]. However, missing rainfall values in the weather dataset reduces the accuracy and robustness of the hydrological data analysis. In the real-world, the data could go missing on more than 50% missing rates of the variable (s) in the dataset due to the equipment malfunctioned and measurement errors. Therefore, this section attempts to compare the predictive ability of the proposed SC-FDO based MLP trainer in handling high missing rates on the time series dataset with benchmark approaches. In addition, the results of the proposed SC-FDO based MLP with random weight factor, abbreviated as SC-FDO and SC-FDO with fixed weight factor, abbreviated as SC-FDO (fixed wf) were compared to evaluate the effect of the proposed random weight factor in imputation.

1) DATASET

The dataset used in this study was historical weather data for Basel, Switzerland and downloaded from meteoblue website [42]. This study analyzed the daily historical weather data from January 1985 to September 2020 with no missing attribute values. We performed the principal component analysis method to find the most important features. The results showed that the most important features are the rainfall, average soil moisture, minimum soil moisture, max soil moisture, minimum temperature, maximum wind speed, low cloud cover low, medium cloud cover, high cloud cover, total cloud cover, relative humidity, maximum relative humidity, and minimum relative humidity. Furthermore, the daily weather data were split into training and testing datasets. The size of each training and testing dataset is 80% and 20% of the daily weather dataset, respectively. The data were randomly removed from the testing dataset in nine missing rates: 10%, 20%, 30%, 40%, 50%, 60%, 70%, 80% and 90% [43], [44]. The missing values were categorized as missing completely at random (MCAR) [45], [46] because the presence of missing values is not affected by the other variable values in the dataset.

2) EXPERIMENT SETTINGS

The experiments were conducted using MATLAB R2018a. The computer settings were set the same as the sub-section of Benchmark Test Function settings at Section IV Numerical Experiment and Results. All the experiments were executed for 30 independent runs over each missing rate. The population size was fixed to 30, the number of hidden neurons was 15, the maximum number of iterations (t_{max}) was 1100, and the maximum number of epochs was 1000 on all simulations. The parameter settings of the algorithms were presented in Table 8.

TABLE 8. Parameter settings for algorithms.

Algorithms	Parameter	Value of the parameter
SC-FDO	a	2
	wf	[0,1]
SC-FDO (fixed wf)	a	2
	wf	1
FDO	wf	1
IFDO	wf	[0,1]

3) PERFORMANCE MEASURES

The performances of the SC-FDO and the benchmark approaches were measured as follows:

- Mean absolute error (MAE)

$$MAE = \frac{1}{N}\sum\nolimits_{i=1}^{N} |O_i - T_i| \tag{23}$$

- Root mean square error (RMSE)

$$RMSE = \sqrt{\frac{\sum_{i=1}^{N} (O_i - T_i)^2}{N}} \tag{24}$$

- Correlation coefficient (R)

$$R = \frac{\sum_{i=1}^{N} \left(T_i - \bar{T}\right)\left(O_i - \bar{O}\right)}{\sqrt{\sum_{i=1}^{N} \left(T_i - \bar{T}\right)^2 \left(O_i - \bar{O}\right)^2}} \tag{25}$$

where N is the total number of observations, 0 is the actual values of observations and $T$is the imputed values.

4) RESULTS AND DISCUSSIONS

The effects of missing data on the imputation ability of the SC-FDO based MLP trainer and the benchmark approaches, SC-FDO (fixed wf), FDO, IFDO based MLP trainers are shown in Fig. 8. The boxplots in Fig. 8 show a summary of the distribution of imputation results based on minimum, first quartile (Q1), median, third quartile (Q3) and maximum values. First, the missing data imputation methods were evaluated for the low proportion of missing values, from 10% until 40% missing rates, as depicted in Fig. 8(a)-8(c).

The FDO imputation method was the most sensitive to the percentage of missing rates. The results indicated that FDO has the lowest performance in the presence of low missing values. The highest median of MAE in mm (10%: 0.179, 20%: 0.3904, 30%: 0.6324 and 40%: 0.8731) and the highest median of RMSE in mm (10%: 0.7715, 20%: 1.3814, 30%: 1.6969 and 40%: 2.3070) but the lowest median of R (10%: 0.9889, 20%: 0.9630, 30%: 0.9442 and 40%: 0.8946) were observed for the FDO imputation.

The IFDO imputation slightly performed better than the FDO imputation. The MAE results indicated that IFDO was less sensitive to the amount of low missingness than the FDO imputation, in which the IFDO has the second-highest median of MAE and RMSE. For the SC-FDO (fixed wf) method, the MAE, RMSE, and R results show better model performance than IFDO and FDO for all the low missing cases.

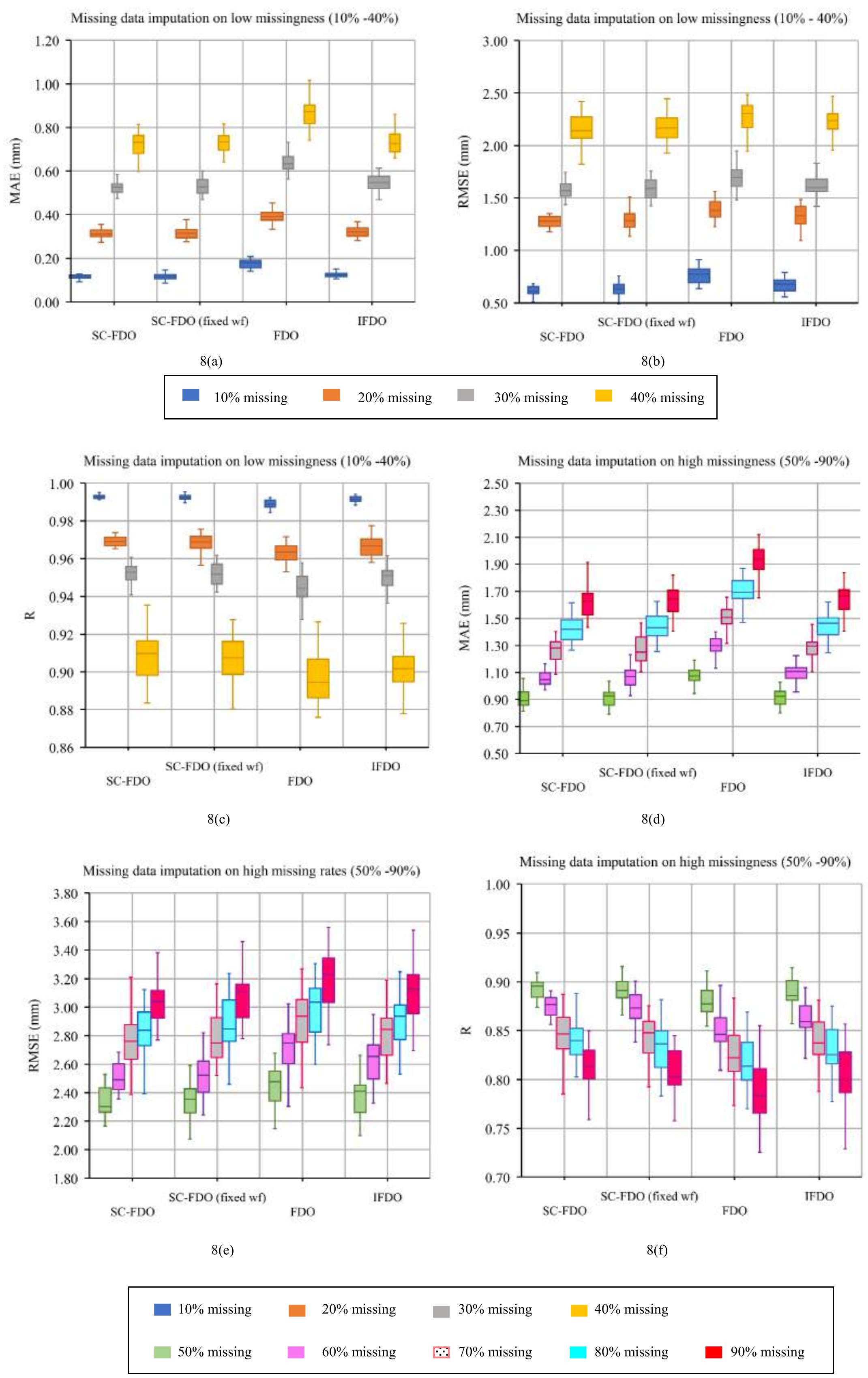


FIGURE 8. Missing data imputation on high-low missingness.

In addition, the results showed that the SC-FDO imputation achieved the lowest median of MAE (10%: 0.1149 mm, 20%: 0.3098 mm, 30%: 0.5253 mm and 40%: 0.7324 mm), the lowest median of RMSE (10%: 0.6130 mm, 20%: 1.2784 mm, 30%: 1.5685 mm and 40%: 2.1431 mm), however the highest median of R (10%: 0.9929, 20%: 0.9691, 30%: 0.9527 and 40%: 0.9105) for the low proportion of missingness cases. With the implementation of the random weight factor (*wf*) and global fitness weight parameter (*fw**), the imputation results of SC-FDO showed improvements in the three performance measures compared to the SC-FDO (fixed *wf*), FDO, and IFDO imputation. The shorter distributions of MAE, RMSE, and R in SC-FDO, indicating that the SC-FDO is substantially better than SC-FDO (fixed *wf*), FDO, and IFDO imputation. Thus, the SC-FDO imputation is the preferred method in the presence of low missingness compared to the SC-FDO (fixed *wf*), FDO, and IFDO.

Furthermore, this study revealed the effects of missing data imputation for high missingness from 50% to 90%, as shown in Fig. 8(d)-8(f). For the large proportion of missing data, two imputation methods stood out as the median R values of the SC-FDO and SC-FDO (fixed *wf*) were higher than the other two imputation methods. The SC-FDO (fixed *wf*) was the highest median R (R = 0.8481 mm) for the missing rates of 70%. Meanwhile, the SC-FDO obtained the highest median of R values for the high missingness of 50%, 60%, 80%, and 90%. The median R values indicated that overall, the SC-FDO has higher accuracy than the SC-FDO (fixed *wf*), IFDO, and FDO imputation. The underlying reason is the SC-FDO enables the scout bees to converge more accurately than the SC-FDO (fixed *wf*), FDO, and IFDO compared to the equations (10)-(13) with (2) respectively.

In addition, the FDO imputation generates the highest median of MAE and RMSE values for all the high missingness, with the MAE values, range between 1.0737 mm to 1.9363 mm and the RMSE values range between 2.4760 mm and 3.2274 mm, respectively. On the other hand, the SC-FDO demonstrated the best performances compared to the FDO imputation. The median MAE and RMSE values of the SC-FDO decreased by an average range between 0.18 mm and 0.31 mm, and an average range between 0.17 and 0.25 mm, respectively. Meanwhile, the median MAE and RMSE of SC-FDO (fixed *wf*) and IFDO imputation laid between the SC-FDO and FDO imputations.

Overall, the SC-FDO imputation outperformed the three imputation methods with the highest average accuracy of 90% when treating the low and high missingness in the dataset.

Furthermore, a comparison of average execution time for SC-FDO, SC-FDO (fixed *wf*), FDO, and IFDO based MLP trainers is plotted in Fig. 9. The proposed SC-FDO optimizer trainer has the lowest average execution time. Meanwhile, the IFDO optimizer trainer took the longest average execution time to perform missing data estimation for all the missingness. Overall, the average execution time of the proposed SC-FDO was slightly less time than the SC-FDO (fixed *wf*). However, the FDO and IFDO optimizer trainers took more computational time to perform missing data estimation than the proposed SC-FDO. The SC-FDO optimizer trainer reduced the computational time up to an average of 77% and 87% compared to the original FDO and IFDO, respectively.

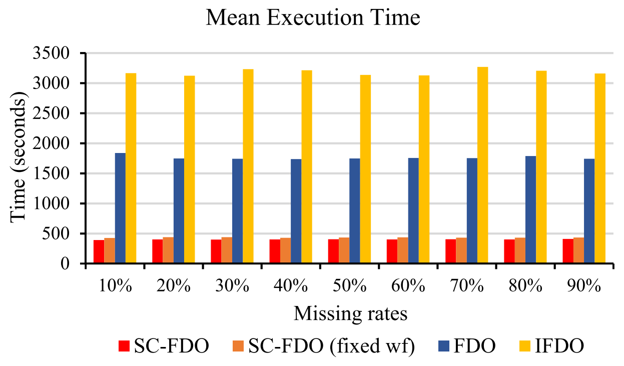


FIGURE 9. **Comparison of average execution time (seconds) for SC-FDO based MLP trainer and the benchmark approaches at different missing rates.**

5) ANALYSIS OF IMPUTATION RESULTS

The proposed SC-FDO imputation-based MLP trainer demonstrated the best performance for most levels of missingness than the other three optimizer trainers. This study revealed that the proposed SC-FDO imputation method achieved an improvement in prediction accuracy than the SC-FDO (fixed *wf*), FDO, and IFDO optimizer trainers. The adaptation of the random (*wf*) and global fitness weight (*fw**) strategy improved the performance of the SC-FDO imputation. The global fitness weight (*fw**) parameter helped the SC-FDO finds appropriate random *wf* over the iterations for stable search. Without the global fitness weight (*fw**) strategy, the small value of *wf* results in less exploration, whereas the high value of *wf* may result in premature convergence. Additionally, the two strategies (the modified pace-updating equation and the conversion parameter) in the proposed SC-FDO also enhanced the balance of exploratory and exploitative characteristics of the original FDO. Consequently, the proposed SC-FDO imputation produces consistently good imputation results than the SC-FDO (fixed *wf*), FDO, and IFDO optimizer trainers.

In addition to that, the proposed SC-FDO also improved the efficiency of the original FDO and IFDO imputation. The SC-FDO has significantly shortened the computational time of the FDO and IFDO, approximately 77% and 87%, respectively. The main reason is the proposed best solution-updating function in the SC-FDO could positively reduce the time taken to find the best search region by periodically updating the position around the global best solution during optimization.

Furthermore, this study found that the performances of the four imputation methods decreased as the missing rates increased. The level of imputation sensitivity depends on the percentages of missingness and the imputation models. The FDO imputation was the most sensitive for the growing ratios of missingness in the dataset among the four methods. The accuracy of the FDO imputation was reduced to 78% when the missing rate is 90%. Similar distributions are also observed for the SC-FDO (fixed *wf*) and IFDO imputations. However, the proposed SC-FDO imputation is less sensitive as the fraction of missing data increased. The proposed SC-FDO obtained an accuracy of 81% at the missing rate of 90%. Our findings are consistent with the work done by Gill *et al.* [47], Kim *et al.* [48], and Chiu *et al.* [49] that the effect of missingness is significant when the fraction of missing data grows larger. Therefore, the proposed SC-FDO was the best method for the low and high proportions of missingness.

VI. CONCLUSION

This study demonstrated the effect of the modified pace-updating equation, the random weight factor (*wf*) and global fitness weight (fw^*) strategy, the conversion parameter strategy, and the best solution-updating strategy in the proposed SC-FDO. The benchmark test results revealed that the proposed SC-FDO performed better than the existing FDO and several well-known optimization algorithms in terms of convergence precision and speed. The SC-FDO has significantly obtained theoretically or approximately optimal solutions for most of the benchmarking test cases. The results also proved that the proposed SC-FDO has successfully balanced the FDO's exploitation and exploration, improved the convergence speed, avoided the local optima, and moved towards optimality. Furthermore, the Wilcoxon rank-sum test results proved that the proposed SC-FDO was systematically different from the benchmark algorithms at the 0.05 significance level. Additionally, the proposed SC-FDO based MLP trainer demonstrated encouraging results than the SC-FDO (fixed *wf*), FDO and IFDO based MLP trainers in solving the problems of low and high missingness in the time series dataset. The missing value cases were refined as optimization problems, where the four optimizer trainers were used to predict missing values in the time series datasets at different missing rates from 10% to 90%. The SC-FDO trainer outperformed the other three optimizer trainers with the highest average accuracy of 90% for all the missing rates. The SC-FDO trainer also obtained a computational time reduction of 77% and 87% in missing data estimation compared to the FDO and IFDO optimizer trainers, respectively. Therefore, the findings of the proposed SC-FDO support its use to optimize the real-world missing data problems.

VII. LIMITATION AND FUTURE WORKS

The SC-FDO requires parameter tuning for constant a in the conversion parameter strategy. Since parameter tuning is typically computationally expensive, particularly for real-world applications, an automatic parameter tuning method needs to further explore in the future. The researchers can also investigate multi-objective parameter tuning and cost effectiveness on the SC-FDO. In addition, the hybridization of the SC-FDO with other metaheuristic algorithms such as coot algorithm [20] and colony predation algorithm [21] could be of interest to the researchers.

ACKNOWLEDGMENT

The authors are grateful for the support of student Sebastien Mambou in consultations regarding application aspects.

REFERENCES

[1] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Proc. Int. Conf. Neural Netw. (ICNN)*, vol. 4, Nov./Dec. 1995, pp. 1942–1948.
[2] R. Storn and K. Price, "Differential evolution-a simple and efficient heuristic for global optimization over continuous spaces," *J. Global Optim.*, vol. 11, no. 4, pp. 341–359, 1997.
[3] J. H. Holland, "Genetic algorithms," *Sci. Amer.*, vol. 267, no. 1, pp. 66–73, 1992.
[4] S. Mirjalili, "SCA: A sine cosine algorithm for solving optimization problems," *Knowl.-Based Syst.*, vol. 96, pp. 120–133, Mar. 2016.
[5] H. Huang, X. Feng, A. A. Heidari, Y. Xu, M. Wang, G. Liang, H. Chen, and X. Cai, "Rationalized sine cosine optimization with efficient searching patterns," *IEEE Access*, vol. 8, pp. 61471–61490, 2020.
[6] J. M. Abdullah and T. A. Rashid, "Fitness dependent optimizer: Inspired by the bee swarming reproductive process," *IEEE Access*, vol. 7, pp. 43473–43486, Mar. 2019.
[7] N. Covic and B. Lacevic, "Wingsuit flying search—A novel global optimization algorithm," *IEEE Access*, vol. 8, pp. 53883–53900, Mar. 2020.
[8] S. Mirjalili and A. Lewis, "The whale optimization algorithm," *Adv. Eng. Softw.*, vol. 95, pp. 51–67, May 2016.
[9] J. Tu, H. Chen, J. Liu, A. A. Heidari, X. Zhang, M. Wang, R. Ruby, and Q.-V. Pham, "Evolutionary biogeography-based whale optimization methods with communication structure: Towards measuring the balance," *Knowl.-Based Syst.*, vol. 212, Jan. 2021, Art. no. 106642.
[10] S. Arora and S. Singh, "Butterfly optimization algorithm: A novel approach for global optimization," *Soft Comput.*, vol. 23, no. 3, pp. 715–734, 2019.
[11] M. Tubishat, M. Alswaitti, S. Mirjalili, M. A. Al-Garadi, M. T. Alrashdan, and T. A. Rana, "Dynamic butterfly optimization algorithm for feature selection," *IEEE Access*, vol. 8, pp. 194303–194314, 2020.
[12] S. Mirjalili, "Dragonfly algorithm: A new meta-heuristic optimization technique for solving single-objective, discrete, and multi-objective problems," *Neural Comput. Appl.*, vol. 27, no. 4, pp. 1053–1073, 2016.
[13] X. Cui, Y. Li, J. Fan, T. Wang, and Y. Zheng, "A hybrid improved dragonfly algorithm for feature selection," *IEEE Access*, vol. 8, pp. 155619–155629, 2020.
[14] S. Mirjalili, S. M. Mirjalili, and A. Lewis, "Grey wolf optimizer," *Adv. Eng. Softw.*, vol. 69, pp. 46–61, Mar. 2014.
[15] S. Mirjalili, "Moth-flame optimization algorithm: A novel nature-inspired heuristic paradigm," *Knowl.-Based Syst.*, vol. 89, pp. 228–249, Nov. 2015.
[16] C. Chen, X. Wang, H. Yu, M. Wang, and H. Chen, "Dealing with multi-modality using synthesis of moth-flame optimizer with sine cosine mechanisms," *Math. Comput. Simul.*, vol. 188, pp. 291–318, Oct. 2021.
[17] D. Oliva, S. Esquivel-Torres, S. Hinojosa, M. Pérez-Cisneros, V. Osuna-Enciso, N. Ortega-Sánchez, G. Dhiman, and A. A. Heidari, "An opposition-based moth swarm algorithm for global optimization," *Expert Syst. Appl.*, vol. 184, Dec. 2021, Art. no. 115481.
[18] W. Shan, Z. Qiao, A. A. Heidari, H. Chen, H. Turabieh, and Y. Teng, "Double adaptive weights for stabilization of moth flame optimizer: Balance analysis, engineering cases, and medical diagnosis," *Knowl.-Based Syst.*, vol. 214, Feb. 2021, Art. no. 106728.
[19] X. Wang, D. Cong, Z. Yang, and J. Han, "Root based optimization algorithm for task-oriented structural design of a multi-axial road test rig," *IEEE Access*, vol. 8, pp. 168061–168078, 2020.
[20] I. Naruei and F. Keynia, "A new optimization method based on COOT bird natural life model," *Expert Syst. Appl.*, vol. 183, Nov. 2021, Art. no. 115352.

[21] J. Tu, H. Chen, M. Wang, and A. H. Gandomi, "The colony predation algorithm," *J. Bionic Eng.*, vol. 18, no. 3, pp. 674–710, May 2021.

[22] A. Daraz, S. Abdullah, H. Mokhlis, I. U. Haq, G. Fareed, and N. N. Mansor, "Fitness dependent optimizer-based automatic generation control of multi-source interconnected power system with non-linearities," *IEEE Access*, vol. 8, pp. 100989–101003, 2020.

[23] A. Daraz, S. A. Malik, I. U. Haq, K. B. Khan, G. F. Laghari, and F. Zafar, "Modified PID controller for automatic generation control of multi-source interconnected power system using fitness dependent optimizer algorithm," *PLoS ONE*, vol. 15, no. 11, Nov. 2020, Art. no. e0242428.

[24] D. S. Abdul-Minaam, W. M. E. S. Al-Mutairi, M. A. Awad, and W. H. El-Ashmawi, "An adaptive fitness-dependent optimizer for the one-dimensional bin packing problem," *IEEE Access*, vol. 8, pp. 97959–97974, 2020.

[25] D. A. Muhammed, S. A. M. Saeed, and T. A. Rashid, "Improved fitness-dependent optimizer algorithm," *IEEE Access*, vol. 8, pp. 19074–19088, 2020.

[26] A. Daraz, S. A. Malik, H. Mokhlis, I. U. Haq, F. Zafar, and N. N. Mansor, "Improved-fitness dependent optimizer based FOI-PD controller for automatic generation control of multi-source interconnected power system in deregulated environment," *IEEE Access*, vol. 8, pp. 197757–197775, 2020.

[27] H. M. Mohammed and T. A. Rashid, "Chaotic fitness dependent optimizer for planning and engineering design," Soft Comput., Tech. Rep., Aug. 2021.

[28] D. H. Wolper and W. G. Macready, "No free lunch theorems for optimization," *IEEE Trans. Evol. Comput.*, vol. 1, no. 1, pp. 67–82, Apr. 1997.

[29] W, Long, T. Wu, X. Liang, and S. Xu, "Solving high-dimensional global optimization problems using an improved sine cosine algorithm," *Expert Syst. Appl.*, vol. 123, pp. 108–126, Jun. 2019.

[30] S. Gupta, K. Deep, H. Moayedi, L. K. Foong, and A. Assad, "Sine cosine grey wolf optimizer to solve engineering design problems," *Eng. Comput.*, pp. 1–27, Feb. 2020.

[31] H. M. Mohammed, S. U. Umar, and T. A. Rashid, "A systematic and meta-analysis survey of whale optimization algorithm," *Comput. Intell. Neurosci.*, vol. 2019, pp. 1–25, Apr. 2019.

[32] C. Qu, Z. Zeng, J. Dai, Z. Yi, and W. He, "A modified sine-cosine algorithm based on neighborhood search and greedy levy mutation," *Comput. Intell. Neurosci.*, vol. 2018, pp. 1–19, Jul. 2018.

[33] M. Issa, A. E. Hassanien, D. Oliva, A. Helmi, I. Ziedan, and A. Alzohairy, "ASCA-PSO: Adaptive sine cosine optimization algorithm integrated with particle swarm for pairwise local sequence alignment," *Expert Syst. Appl.*, vol. 99, pp. 56–70, Jun. 2018.

[34] S. N. Chegini, A. Bagheri, and F. Najafi, "PSOSCALF: A new hybrid PSO based on sine cosine algorithm and Levy flight for solving optimization problems," *Appl. Soft Comput.*, vol. 73, pp. 697–726, Dec. 2018.

[35] X.-S. Yang, *Nature-Inspired Metaheuristic Algorithms*. London, U.K.: Luniver Press, 2010.

[36] K. V. Price, N. H. Awad, M. Z. Ali, and P. N. Suganthan, "The 100-digit challenge: Problem definitions and evaluation criteria for the 100-digit challenge special session and competition on single objective numerical optimization," Nanyang Technol. Univ., Singapore, Tech. Rep., Nov. 2018.

[37] Statistics Kingdom. *Mann Whitney U Test Calculator (Wilcoxon Rank-Sum), Non-Parametric Test*. Accessed: Jul. 1, 2021. [Online]. Available: https://www.statskingdom.com/170median_mann_whitney.html

[38] K. K. Kuok, S. Harun, S. M. Shamsuddin, and P. C. Chiu, "Evaluation of daily rainfall-runoff model using multilayer perceptron and particle swarm optimization feed forward neural networks," *J. Environ. Hydrol.*, vol. 18, no. 10, pp. 1–6, 2010.

[39] S. Mirjalili, "How effective is the Grey Wolf optimizer in training multi-layer perceptrons," *Appl. Intell.*, vol. 43, no. 1, pp. 150–161, 2015.

[40] Y. Shen, M. M. Morsy, C. Huxley, N. Tahvildari, and J. L. Goodall, "Flood risk assessment and increased resilience for coastal urban watersheds under the combined impact of storm tide and heavy rainfall," *J. Hydrol.*, vol. 579, Dec. 2019, Art. no. 124159.

[41] K. K. Kuok, S. Harun, and C. P. Chan, "Hourly runoff forecast at different leadtime for a small watershed using artificial neural networks," *Int. J. Advance Soft Comput. Appl*, vol. 3, no. 1, pp. 68–86, 2011.

[42] Meteoblue. Canton of Basel-City. Switzerland. *Weather History Download Basel*. Accessed: Aug. 1, 2020. https://www.meteoblue.com/en/weather/archive/export/basel_switzerland_2661604

[43] P. C. Chiu, A. Selamat, and O. Krejcar, "Infilling missing rainfall and runoff data for Sarawak, Malaysia using Gaussian mixture model based K-nearest neighbor imputation," in *Proc. Int. Conf. Ind., Eng. Other Appl. Appl. Intell. Syst.* Cham, Switzerland: Springer, Jul. 2019, pp. 27–38.

[44] P. Zhu, Q. Xu, Q. Hu, C. Zhang, and H. Zhang, "Multi-label feature selection with missing labels," *Pattern Recognit.*, vol. 74, pp. 488–502, Feb. 2018.

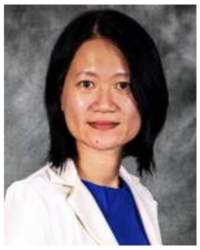
[45] R. J. Little and D. B. Rubin, *Statistical Analysis With Missing Data*. Hoboken, NJ, USA: Wiley, Aug. 2014.

[46] P. C. Chiu, A. Selamat, O. Krejcar, and K. K. Kuok, "Missing rainfall data estimation using artificial neural network and nearest neighbor imputation," in *Proc. Advancing Technol. Industrialization Through Intell. Softw. Methodol., Tools Techn.: 18th Int. Conf. New Trends Intell. Softw. Methodol., Tools Techn. (SoMeT)*, vol. 318. Amsterdam, The Netherlands: IOS Press, Sep. 2019, pp. 132–143.

[47] M. K. Gill, T. Asefa, Y. Kaheil, and M. McKee, "Effect of missing data on performance of learning algorithms for hydrologic predictions: Implications to an imputation technique," *Water Resour. Res.*, vol. 43, no. 7, Jul. 2007.

[48] T. Kim, W. Ko, and J. Kim, "Analysis and impact evaluation of missing data imputation in day-ahead PV generation forecasting," *Appl. Sci.*, vol. 9, no. 1, p. 204, Jan. 2019.

[49] P. C. Chiu, A. Selamat, O. Krejcar, and K. K. Kuok, "Imputation of rainfall data using improved neural network algorithm," in *Proc. Int. Conf. Pattern Recognit. (ICPR) Int. Workshops Challenges*, in Lecture Notes in Computer Science, vol. 12664, A. Del Bimbo, Ed. Cham, Switzerland: Springer, Jan. 2021, pp. 390–406.



PO CHAN CHIU received the M.Sc. degree in information technology from Universiti Malaysia Sarawak (UNIMAS), in 2010. She is currently pursuing the Ph.D. degree in computer science with Universiti Teknologi Malaysia (UTM). She started her career as a Software Engineer, for three years. She is currently serving as a Lecturer for UNIMAS. She worked on several consultancy projects and developed software solutions to meet the needs of the woodworking industry. Her research interests include artificial intelligence, data analytics, optimization, and neural networks.

ALI SELAMAT (Member, IEEE) has been the Dean of Malaysia Japan International Institute of Technology (MJIIT), Universiti Teknologi Malaysia (UTM), Malaysia, since 2018. He is currently a Full Professor with UTM. He is also a Professor with the Department of Software Engineering, School of Computing, UTM. An academic institution was established under the cooperation of the Japanese International Cooperation Agency (JICA) and the Ministry of Education Malaysia (MOE) to provide the Japanese style of education in Malaysia. He has published more than 120 research articles with IF JCR, with more than 2400 citations received in the Web of Science and H-index 26. His research interests include software engineering, software process improvement, software agents, web engineering, information retrievals, pattern recognition, genetic algorithms, neural networks, soft computing, collective computational intelligence, strategic management, key performance indicator, and knowledge management. He is the Chair of the IEEE Computer Society Malaysia Section. He is on the Editorial Board of the journal *Knowledge-Based Systems* (Elsevier).

ONDREJ KREJCAR is currently a Full Professor of systems engineering and informatics with the University of Hradec Kralove (UHK), Czech Republic. He is also the Vice Dean for science and research at the Faculty of Informatics and Management, UHK. He is the Director of the Center for Basic and Applied Research, University of Hradec Kralove. At UHK, he is a Guarantee of the Doctoral Study Programme in Applied Informatics, where he is focusing on lecturing on smart approaches to the development of information systems and applications in ubiquitous computing environments. He has been the Vice Leader and a Management Committee Member of the Project COST CA17136 at WG4, since 2018. He has also been a Management Committee Member substitute of the Project COST CA16226, since 2017. He has a number of collaborations throughout the world, namely Malaysia, Spain, U.K., Ireland, Ethiopia, Latvia, and Brazil. His H-index is 20 (according to Web of Science), with more than 1500 citations received in the Web of Science. He has published more than 110 research articles with IF JCR. His research interests include control systems, smart sensors, ubiquitous computing, manufacturing, wireless technology, portable devices, biomedicine, image segmentation and recognition, biometrics, technical cybernetics, and ubiquitous computing. His second area of interest is in biomedicine (image analysis), as well as biotelemetric system architecture (portable device architecture and wireless biosensors), and the development of applications for mobile devices with use of remote or embedded biomedical sensors.

Dr. Krejcar has been the Deputy Chairman of the Panel 7, namely Processing Industry, Robotics, and Electrical Engineering of the Epsilon Program, Technological Agency of the Czech Republic, since 2014. Since 2019, he has been the Chairman of the Program Committee of the KAPPA Program, Technological Agency of the Czech Republic, and a Regulator of the EEA/Norwegian Financial Mechanism in the Czech Republic, for the period 2019–2024. He is on the Editorial Board of *Sensors* (MDPI) with JCR Index and several other ESCI indexed journals. In 2018, he was the 14th Top-Peer Reviewer in Multidisciplinary in the World according to Publons.

KING KUOK KUOK received the M.Eng. degree from UNIMAS, in 2004, and the Ph.D. degree from UTM, in 2010. He was a Field Engineer with the Hydrological and Water Resources Branch, Department of Irrigation and Drainage, Sarawak, Malaysia, from 2002 to 2009, and the Road, Civil, and Structural Design Engineer at private companies, for more than ten years. He is currently a Senior Lecturer with Swinburne University of Technology Sarawak Campus. His research interests include water resources, water supply, hydrology, artificial intelligence, and building information modeling.