ECTI Transactions on Computer and Information Technology
Journal homepage: https://ph01.tci-thaijo.org/index.php/ecticit/
Published by the ECTI Association, Thailand, ISSN: 2286-9131

A Performance of AFIRO among Asynchronous Iteration Strategy Metaheuristic Algorithms

Tasiransurini Ab Rahman[1], Nor Azlina Ab. Aziz[2] and Zuwairie Ibrahim[3]

ABSTRACT

Asynchronous Finite Impulse Response Optimizer (AFIRO) is a metaheuristic algorithm that has been developed as a population-based solution with an asynchronous update mechanism. AFIRO is inspired by the Ultimate Unbiased Finite Impulse Response filter framework. AFIRO works with a group of agents where each agent performs the iteration update asynchronously. In the original paper, AFIRO was compared with the Particle Swarm Optimisation algorithm, Genetic Algorithm, and Grey Wolf Optimizer. Although AFIRO shows a better performance, the comparison seems unfair since the iteration strategy of AFIRO is different from those compared algorithms. Hence, this article further investigates the potential of AFIRO against three existent metaheuristic algorithms with the same iteration strategy, namely Asynchronous PSO (A-PSO), Asynchronous Gravitational Search Algorithm (A-GSA), and Asynchronous Simulated Kalman Filter (A-SKF). The CEC2014 test suite was applied to evaluate the performance, where the results revealed that AFIRO leads 18 out of 30 functions. The Holm post hoc showed that AFIRO performs significantly better than A-SKF and A-GSA while having the same performance as A-PSO. Moreover, the Friedman test disclosed that AFIRO has the highest ranking than A-PSO, A-SKF, and A-GSA. Therefore, it can be concluded that AFIRO performs well in the same iteration strategy category.

Article information:

Keywords: Asynchronous Iteration Strategy, Finite Impulse Response, Metaheuristic Algorithm, Population-Based

Article history:
Received: March 3, 2023
Revised: June 3, 2023
Accepted: July 3, 2023
Published: July 22, 2023
(Online)

DOI: 10.37936/ecti-cit.2023173.251829

1. INTRODUCTION

The iteration strategy is important for population-based metaheuristic algorithms. It specifies the sequence of search steps for agents, considering each other in updating a solution. The iteration strategy can be classified into synchronous and asynchronous updates [1]–[3]. The synchronous update is commonly implemented in most population-based algorithms such as the Honey Badger Algorithm [4], Golden Eagle Optimizer [5], Arithmetic Optimisation Algorithm [6], Carnivorous Plant Algorithm [7], and Black Widow Optimisation Algorithm [8]. Pseudocode 1 shows a general procedure of population-based metaheuristic algorithms. To note, the iteration strategy distinguishes steps 2 and 3.

Pseudocode 1

1 : Randomly initialize possible solutions
2 : Evaluate the current solution
3 : Generate the next possible solution
4 : **While** not stopping conditions **do**
Repeat steps no.2 and no.3
End the algorithm
5 : Return the best-found solution

In a synchronous update, both steps are executed as a group (population) where step 2 needs to be completed by the entire members first before step 3 can be performed. All agents consider the performance of all members before improving (updating) the solution. As an advantage, the agents from a synchronous update are good in exploitation [9] in which the agents

[1] The author is with Computational, Signal, Imaging and Intelligence Focus Group, Department of Electronic Engineering, Faculty of Electrical and Electronic Engineering, Universiti Tun Hussein Onn Malaysia, 86400, Johor, Malaysia, E-mail: surini@uthm.edu.my
[2] The author is with Center for Engineering Computational Intelligence, Faculty of Engineering and Technology, Multimedia University, 75450, Melaka, Malaysia, E-mail: azlina.aziz@mmu.edu.my
[3] The author is with College of Engineering, Universiti Malaysia Pahang, 26600 Pahang, Malaysia, E-mail: zuwairie@ump.edu.my

are dragged to the same point of reference. However, in this condition, the agents are inferior in the exploration.

In contrast, for an asynchronous update, both steps 2 and 3 are carried out as individual tasks without considering the synchronicity with other agents. Unlike a synchronous update, the agents in an asynchronous update improve the solution independently. The agent determines the point of reference immediately after completing its fitness evaluation. As an advantage, the agents from an asynchronous update are good in the exploration.

Asynchronous Particle Swarm Optimisation (A-PSO) algorithm [10]–[18], Asynchronous Gravitational Search Algorithm (A-GSA) [9], [19], [20], Asynchronous Simulated Kalman Filter (A-SKF) [21], [22], and asynchronous Genetic Algorithm [23] are among metaheuristic algorithms that had been successfully modified from the original iteration strategy (synchronous update) to asynchronous update.

The Particle Swarm Optimisation (PSO) algorithm's search strategy is inspired by the social behaviour perceived in nature, such as a flock of birds seeking food. A swarm of particles (agents) searches for the optimal solution by updating its velocity and position. In PSO, the solution is represented by the particle's position. In the original PSO (synchronous PSO), the best value ($gBest$) is updated after the fitness of all particles is evaluated. Then, the particles' velocity and position are updated. Thus, the particles have complete information on the swarm's fitness before updating the $gBest$. This condition allows for a better choice of $gbest$ that influences a faster convergence and stronger exploitation for PSO [12], [15].

However, a faster convergence contributes to premature convergence, which is to be avoided in a metaheuristic algorithm. Therefore, the A-PSO algorithm was proposed in [16] as another option for the original PSO. In A-PSO, a particle updates its $gBest$ right after the fitness evaluation step and immediately updates its velocity and position, one after another. Thus, $gbest$ is updated with incomplete information about the swarm's fitness. In one iteration, $gbest$ is allowed to be updated more than once, thus encouraging the exploration of particles. As studied by Ab. Aziz and Ibrahim in [10], an asynchronous update of PSO is a more practical strategy for swarm robotics search problems that allow robots to move continuously with information available at that moment. The asynchronous update is also ideal for a parallel implementation of PSO which improves the robot processing capabilities [10], [24].

On a different note, the search for an optimal solution in Gravitational Search Algorithm (GSA) is based on the law of gravity and the interaction of mass in the universe. The particles (agents) are represented as objects and the performance of these objects is assessed through their masses. The heaviest mass in the search space is presumed as an optimal solution [25]. In a metaheuristic algorithm, the search space refers to the area of searching near-optimal solutions that represent the range of possible solutions. Similar to PSO, GSA was originally developed as a synchronous algorithm. In synchronous GSA (S-GSA), the velocity and position of the entire population are only updated after the performance of every agent is assessed [19].

In 2013, Ab. Aziz et al. [20] introduced an asynchronous iteration strategy for GSA as a mechanism to avoid premature convergence. In A-GSA, the velocity and position of the agent are updated immediately after the assessment of its performance without waiting for all agents to be assessed. Hence, the search depends on the information on the best and worst agents in the current and previous iteration. The information of the previous iteration is obtained from the agents that have yet to be updated. The lack of synchronous information on the best and worst agents in searching encourages exploration ability [19].

The Simulated Kalman filter (SKF) was originally developed as a synchronous update, as both PSO and GSA. SKF starts with a random initialization of agents to generate an initial solution within the search space. The fitness of each agent is then evaluated synchronously. Next, the best agent for the group is determined and assigned as $\boldsymbol{X}best(t)$. The best solution found so far ($\boldsymbol{X}true$) is updated if a better solution is found. Next, all agents execute the Kalman filter procedure: predict, measure, and estimate, together. The steps are repeated until the maximum number of iterations is reached.

Subsequently, A-SKF [21] was introduced as another variant of SKF. In A-SKF, the fitness of an agent is compared to $\boldsymbol{X}true$ immediately after the fitness evaluation. $\boldsymbol{X}true$ is updated if a better solution is found. Shortly after, an agent performs the Kalman filter procedure. Unlike the original SKF, $\boldsymbol{X}best(t)$ is not involved in A-SKF. The same steps are repeated for all agents, one after another.

The selection of an iteration strategy for a population-based metaheuristic algorithm influences the overall efficiency of the algorithm. The asynchronous update iteration strategy has shown good performance in the Genetic Algorithm (GA) [23], PSO algorithm [16], and SKF algorithm [21]. The success of the asynchronous update has motivated the development of another metaheuristic optimisation algorithm with asynchronous updates iteration strategy, named Asynchronous Finite Impulse Response Optimizer (AFIRO) [26].

AFIRO is inspired by the estimation procedure in the Ultimate Iterative Unbiased Finite Impulse Response (UFIR) filter. The search strategy inspired by the UFIR filter concept has been investigated earlier as a single solution-based metaheuristic algorithm.

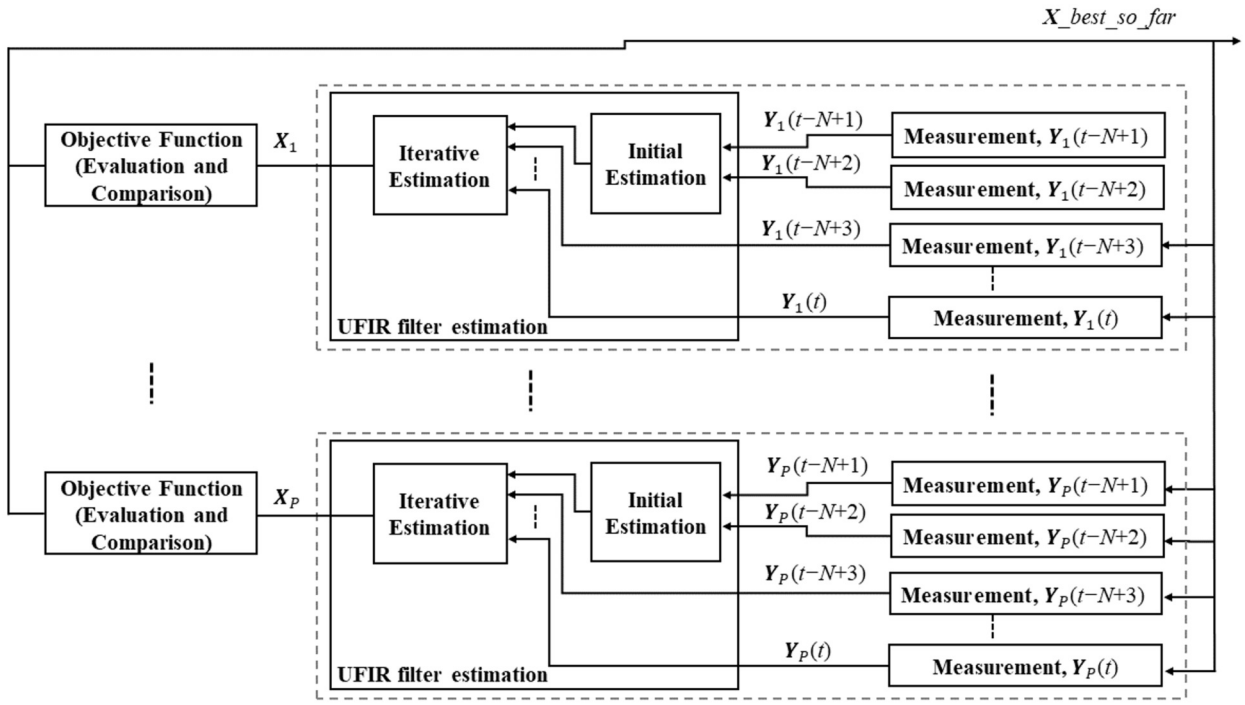

Fig.1: *A Diagram of AFIRO.*

The algorithm, recognized as Single-agent Finite Impulse Response Optimizer (SAFIRO), is as detailed in [27]–[30]. Unlike SAFIRO, AFIRO works with a group of agents to find an optimal or near-optimal solution in solving numerical single-objective optimisation problems. The transformation process from an estimator to an optimiser can be further read in [27].

A good outcome of AFIRO in [26] provides a strong motivation to conduct this study to compare its performance with three metaheuristic algorithms with the same iteration strategy: A-PSO, A-GSA, and A-SKF. In the original paper, AFIRO was compared to PSO, GA, and Grey Wolf Optimizer (GWO). All of the compared algorithms have a synchronous iteration strategy. The comparison seems unfair as AFIRO has a different iteration strategy.

Therefore, this article further compares AFIRO with metaheuristic algorithms that have the same iteration strategy: A-PSO, A-GSA, and A-SKF. The performance was tested by using the IEEE Congress on Evolutionary Computation (CEC) 2014 test suite [31]. The experimental results revealed that the proposed AFIRO can significantly outperform A-GSA and A-SKF while having an equivalent performance with A-PSO. The Friedman test showed that AFIRO is at the highest rank, followed by A-PSO, A-SKF, and A-GSA.

The remainder of this paper is divided as follows: section 2 explains the details of AFIRO, followed by the experimental setup in section 3. Then, section 4 presents the results and discussion, followed by the conclusion in section 5.

2. MATERIALS AND METHODS

2.1 Asynchronous Finite Impulse Response Optimizer (AFIRO)

UFIR filter is one of the popular estimators that is frequently used in estimation problems due to its simple form of mathematical modelling and strong structure, in which it is more robust and stable compared to other estimators [32]–[36]. As aforementioned, the UFIR filter framework is used as the inspirational source of the search strategy in AFIRO. AFIRO is developed as a population-based metaheuristic algorithm that has an asynchronous update iteration strategy.

A set of agents iteratively find a solution asynchronously by using a standard UFIR filter procedure: measurement and estimation. The initialization, measurement and estimation of the solution, fitness evaluation, and update $\boldsymbol{X}\_best\_so\_far$ are four phases in AFIRO. **Fig.1** illustrates how the UFIR filter procedures are adopted in AFIRO.

Every agent in AFIRO serves as an individual UFIR filter that needs an N number of measurements, to perform the estimation in each iteration, t. The search for a solution is performed by P agents. For each sequence, in each iteration, a new measurement, $\boldsymbol{Y}_i(t)$, is simulated by the agent.

The measurement is improved during the estimation of the solution, $\boldsymbol{X}_i(t)$. Each agent performs the initial estimation and iterative estimation during the estimation phase. The estimation of solution for i^{th} agent, $\boldsymbol{X}_i(t)$, at iteration t is defined as $\boldsymbol{X}_i(t) = \{x_i^1(t), x_i^2(t), \ldots, x_i^d(t), \ldots, x_i^D(t)\}$, where x_i^d

indicates the estimated solution of the i^{th} agent in the d^{th} dimension, and D represents the maximum number of dimension.

2.2 Procedure of AFIRO

Pseudocode 2 describes the procedures of AFIRO. Line 01 is the initialization phase; lines 04 to 15 are for the measurement and estimation; line 16 is a fitness evaluation; meanwhile, line 17 updates $\boldsymbol{X}\_best\_so\_far$.

AFIRO commences the optimisation process at line 1, where all agents synchronously perform the initialization phase. The iteration strategy of AFIRO begins at line 03 which is conducted according to the asynchronous update mechanism. Here, the asynchronous update mechanism is related to the sequence of execution by agents for the measurement, $\boldsymbol{Y}_i(t)$, and estimation phase, $\boldsymbol{X}_i(t)$, fitness evaluation phase, and $\boldsymbol{X}\_best\_so\_far$ update.

In each iteration, the agent executes the measurement and estimation, fitness evaluation, and $\boldsymbol{X}\_best\_so\_far$ update one after another, individually. This asynchronous sequence of agents is repeated until all agents complete the procedures.

As shown in Pseudocode 2, initially, the first agent executes the measurement and estimation (lines 04 - 15). Then, the agent immediately assesses its fitness (line 16). Subsequently, the agent updates $\boldsymbol{X}\_best\_so\_far$ (line 17) if the fitness of the estimated solution, $\boldsymbol{X}_i(t)$ itself is better than the fitness of the current $\boldsymbol{X}\_best\_so\_far$.

Next, the second agent executes the same procedures as the first agent. The procedure from lines 04 - 17 is repeated until all agents complete the task. In each sequence, the agent updates $\boldsymbol{X}\_best\_so\_far$ if a better solution is found. Since AFIRO applied an asynchronous update mechanism, $\boldsymbol{X}\_best\_so\_far$ is probably updated more than once in one iteration. Lines 03 to 20 of the procedures are repeated until the halting condition, which is the maximum function evaluation, $maxFES$ is met.

Finally,$\boldsymbol{X}\_best\_so\_far$ is assigned as the optimal solution to the specified optimisation problem. Details of each phase of AFIRO are elaborated in the next sub-sections.

Pseudocode 2 Procedures of AFIRO

```
Algorithm: AFIRO for a minimization problem.
Requirement: horizon length, N and coefficient value, β.
01:     Initialization phase
02:     While not maximum iteration do
03:         for agent=1: P
04:             generate a random value for each
                dimension of a new measurement
05:             if the dimension has a random value ≤ 0.5
06:                 assign a new measurement as
                    Equation 2
07:             else
08:                 a new measurement using Equation
                    3
09:             end
10:                 at sub-iteration, k = 1 and k = 2
11:                     generate a random initial estimation
12:             for k = 3 : N
13:                 an iterative estimation using Equation 4
14:             end
15:             X_i(t) = X̄_i(k)
16:             Evaluate the fitness of agents
17:             Update X_best_so_far
18:             agent+1
19:         end agent
20:         t ⟵ t + 1
21:     end while
22:     Return X_best_so_far
```

2.2.1 Initialization Phase

Similar to the UFIR filter, AFIRO requires N recent measurements to start the optimisation process. All AFIRO's agents conduct the initialization synchronously during this phase. Therefore, N initial measurements, $\boldsymbol{Y}_i(t-3)$, $\boldsymbol{Y}_i(t-2)$, $\boldsymbol{Y}_i(t-1)$, and $\boldsymbol{Y}_i(t)$ are randomly produced by each AFIRO's agent using Equation (1).

$$\boldsymbol{Y}_i(t) = (U[X_{\min}, X_{\max}]) \quad \text{for } t = 0, \ldots, N-1 \tag{1}$$

The subscript, $(_i)$, refers to the agent's number. The $X_{\min}$ and $X_{\max}$ indicate the lower limit and upper limit of the search space, respectively, which depend on the given optimisation problem. Random values are used to explore the search space intensively to find a solution.

The best fitness value is determined by comparing the fitness of initial measurements for all i^{th} agents. For the minimization problem, the initial measurement that has the lowest fitness value is selected as the initial $\boldsymbol{X}\_best\_so\_far$. In contrast, for a maximization problem, the initial measurement that has the highest fitness value is selected as the initial $\boldsymbol{X}\_best\_so\_far$. Besides the initial $\boldsymbol{X}\_best\_so\_far$, $maxFES$ is also defined during the initialization phase.

2.2.2 Measurement and Estimation Phase

After the initial $\boldsymbol{X}\_best\_so\_far$ is obtained from the initialization phase, the agent of AFIRO executes the measurement and estimation of the solution. Dissimilar to the UFIR filter that uses a sensor to get the measurement value, AFIRO produces the measurement using a random mutation of $\boldsymbol{X}\_best\_so\_far$ with a local neighbourhood method.

The agent of AFIRO consists of several dimensions. A uniformly distributed random number between [0,1] is assigned to each dimension. The dimension signifies the difficulty of the problem. The larger the dimension value, the more difficult the problem to be solved.

A random mutation of $\boldsymbol{X}\_best\_so\_far$ is applied in AFIRO to promote the exploration activity. Meanwhile, a shrinking local neighbourhood method is ap-

plied to promote exploitation activity. To note, the mutation process is not applied to all dimensions of measurement value in AFIRO.

Dimensions with random values less than 0.5 are not selected for the mutation process. Thus, the measurement value for these dimensions holds the value of $\boldsymbol{X}\_best\_so\_far$, as shown in Equation (2). Subscript, $(_i)$, represents the agent whereas d indicates the number of dimensions.

$$Y_i^d(t) = \boldsymbol{X}\_best\_so\_far_d \tag{2}$$

On the other hand, dimensions with random values greater than 0.5 undergo the mutation process. The value of 0.5 is selected to give an equal chance for a dimension to mutate. The mutation process in AFIRO is handled within the shrinking local neighbourhood around $\boldsymbol{X}\_best\_so\_far$. Hence, the measurement value for dimensions that performed the mutation is added with the $\boldsymbol{X}\_best\_so\_far$ value, as shown in Equation (3).

$$Y_i^d(t) = \boldsymbol{X}\_best\_so\_far_d + rand(U[-\delta, \delta]) \tag{3}$$

The equation to compute the radius of the local neighbourhood, δ, is shown in Equation (4). A coefficient value, β is used to control the reduction speed of the neighbourhood's size. AFIRO uses $\beta = 10$ for a moderate transition from the exploration to the exploitation phase, as shown in Figure 6 of [27].

$$\delta = e^{-\beta \times \frac{t}{T}} \times \frac{X_{\max} - X_{\min}}{2} \tag{4}$$

After a new measurement, $\boldsymbol{Y}_i(t)$, is obtained, AFIRO's agent calculates an estimation of a solution, $\boldsymbol{X}_i(t)$. AFIRO's agent requires N recent measurements to estimate the solution. The agent estimates the solution in a finite length according to the value of N. Each iteration, t in AFIRO consists of sub-iteration, k, as illustrated in Figure 2.

The number of k must be equal to the value of N. The estimation stage divides into two parts: the initial estimation and the iterative estimation. The former is used to randomly generate the initial estimation of the solution, $\bar{\boldsymbol{X}}_i(k=2)$, whereas the latter is to improve the estimated solution iteratively, starting from $k = 3$ to $k = N$.

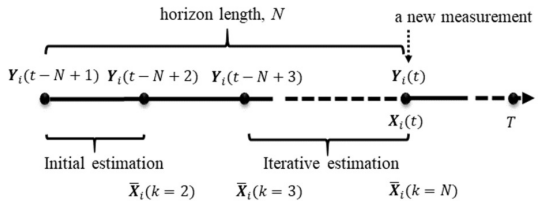

Fig.2: *A graphical representation of the estimation operation in AFIRO.*

In AFIRO, the first two points in sub-iteration, k, are used to generate the initial estimation, $\bar{\boldsymbol{X}}_i(k=2)$ randomly within the [lower limit, upper limit] of the search space, as shown in Pseudocode 3. The random element which is uniformly distributed within (0,1) is applied for the stochastic aspect of AFIRO.

Pseudocode 3 Procedures of initial estimation, $\bar{\boldsymbol{X}}_i(k=2)$, in AFIRO

```
if Y_i(t–N+1) < Y_i(t–N+2)
    _i(k=2) = rand(U[Y_i(t–N+1), Y_i(t–N+2)])
else
    _i(k=2) = rand(U[Y_i(t–N+2), Y_i(t–N+1)])
end
```

Then, the initial estimation is improved iteratively by Equation (5), starting from $k = 3$ to $k = N$.$\bar{\boldsymbol{X}}_i(k)$ is the estimated value for the present point, whereas $\bar{\boldsymbol{X}}_i(k-1)$ is the estimated value for the most recent sub-iteration point.

$$\bar{\boldsymbol{X}}_i(k) = \bar{\boldsymbol{X}}(k-1) + K(k)(\boldsymbol{Y}(t-N+k) - \bar{\boldsymbol{X}}(k-1)) \quad \text{for } k = 3, \ldots, k = N \tag{5}$$

Similar to SAFIRO, the improvement in AFIRO is assisted by the measurement value, $\boldsymbol{Y}(t–N+k)$, and the Kalman-like gain, $K(k)$. The Kalman-like gain, K, can be calculated as in Equation (6). The value in $K(k)$ decreases as sub-iteration, k increases. The operation of sub-iteration stops when $k = N$.

$$K(k) = \frac{1}{k} \tag{6}$$

The final value of $\bar{\boldsymbol{X}}_i(k)$ is then assigned to $\boldsymbol{X}_i(t)$. The estimated solution,$\boldsymbol{X}_i(t)$ represents the candidate solution for that corresponding iteration. The candidate solutions are then evaluated during the fitness evaluation phase.

2.2.3 Fitness evaluation and *X best so far* update

In AFIRO, the estimation phase is conducted immediately after the agent performed the measurement phase. Then, $\boldsymbol{X}\_best\_so\_far$ is updated as soon as the agent evaluates its fitness. During the fitness evaluation phase, the fitness level of the estimated solution, $\boldsymbol{X}_i(t)$ for i^{th} agent is evaluated according to an objective function (fitness function).

An objective function is a function of a given optimisation problem, either minimized or maximized. In a minimization problem, $\boldsymbol{X}\_best\_so\_far$ is updated when the fitness of the current agent is smaller than the fitness of $\boldsymbol{X}\_best\_so\_far$. In contrast, in a maximization problem, $\boldsymbol{X}\_best\_so\_far$ is updated when the fitness of the current agent is larger than the fitness of $\boldsymbol{X}\_best\_so\_far$.

Therefore, $\boldsymbol{X}\_best\_so\_far$ is determined depending on the estimated value of the current agent. If the agent has found a better solution, then

$\boldsymbol{X}\_best\_so\_far$ is updated according to the estimated value of the agent.

The sequence of AFIRO continues for the next agent until all agents finished performing their tasks, consecutively. The procedures of AFIRO are then repeated for the next iteration until $maxFES$ is reached. Once $maxFES$ is reached, $\boldsymbol{X}\_best\_so\_far$ is returned as the optimal solution for the given optimisation problem.

3. EXPERIMENTAL SETUP

As aforementioned, three existent metaheuristic algorithms with asynchronous update iteration strategies were used to benchmark the results of AFIRO: A-PSO, A-GSA, and A-SKF. The experimental settings, parameter settings, as well as data of mean fitness value for A-PSO, A-GSA, and A-SKF were referred from [37]. Table 1 shows the parameter settings for those algorithms. All of the algorithms were implemented using MATLAB.

Table 1: *The Parameter Settings for A-PSO, A-GSA, A-SKF, and AFIRO.*

Algorithm	Parameter	Value
A-PSO	Initial inertia weight, W_1	0.9
	Final inertia weight, W_2	0.4
	Cognitive acceleration factor, C_1	2
	Social acceleration factor, C_2	2
A-GSA	Initial gravitational constant, G_0	100
	Coefficient, β	20
A-SKF	Process noise, Q	0.5
	Measurement noise, R	0.5
	Initial error covariant, $\boldsymbol{P}_i(0)$	1000
AFIRO	Horizon length, N	4
	Coefficient, β	10

The ability of those algorithms in solving the optimisation problems is evaluated by using the CEC2014 test suite for a single-objective real-parameter numerical optimisation. The solution is represented in the fitness form where each function has its ideal fitness value that represents the optimal solution. The MATLAB codes for CEC2014's functions can be obtained from `https://github.com/P-N-Suganthan?tab=repositories`.

In terms of complexity criterion, the performance of metaheuristic algorithms can be fairly compared by using the same function evaluation [38]. Like many others, the complexity of AFIRO is mainly influenced by the complexity of the fitness function rather than the algorithm, as the algorithm only involves basic mathematical operations.

The experimental settings for AFIRO were set the same as all of the referred algorithms. The stopping condition was set to the maximum number of function evaluations ($maxFES$), where $maxFES$ = 10,000 iterations $\times$ dimensions, D[31]. The complexity of the problem was set as 30 dimensions. All experiments were run 30 times with 100 agents on each test function. Thus, the amounts of computation per iteration were significantly the same for all algorithms, where the evaluations were based on the mean fitness value of over 30 runs time with 300,000 $maxFES$. The search space within [-100,100] was used for all dimensions in all functions.

A nonparametric statistical analysis was carried out due to its suitability for multi-problem analysis, where more than one problem (functions) is considered in comparing the performances of algorithms. All algorithms were compared with each other. Thus, the Friedman statistical test for multiple algorithms comparison (with a significant level, α=0.05) was applied as a statistical analysis tool, to rank the performances over a set of results. The null hypothesis of the Friedman test defines that all performances of the tested algorithms are equivalent to each other [39].

The post hoc analysis using Holm's method (with a tolerance level α=0.05) was also applied in this experiment since the Friedman statistic value shows a significant difference exists between the algorithms. Holm's method can detect the differences, where the null hypothesis is rejected if the statistical value is smaller than the p-value [39]. Both Friedman and Holm's post hoc tests were handled using the KEEL Suite 3.0 software (`http://www.keel.es`).

4. RESULTS AND DISCUSSION

This section discusses the performances of A-PSO, A-SKF, A-GSA, and AFIRO in solving 30 benchmark functions in the CEC2014 test suite. The test suite contains three rotated unimodal functions, thirteen simple multimodal problems, six hybrid functions, and eight composition functions. Each of the functions has different properties that can test the algorithm to solve various cases of problems. The properties for all functions are available in [31].

The values of mean fitness and mean error for all algorithms are recorded in Table 2. The functions in the CEC2014 test suite are for minimization optimisation problems. Thus, the smaller value of mean fitness represents a better result. The smallest reading of the mean fitness value for each function is marked in bold, which represents the best performance of each function (Fn).

The ability of AFIRO to solve unimodal optimisation problems is tested by solving Function 1 (Fn1) - Function 3 (Fn3). Table 2 clearly shows that all algorithms are unable to obtain the optimal solution for Fn1 and Fn2. However, AFIRO's performance is far better than the other algorithms for Fn1. The optimal solution for Fn1 is 100. As for Fn2, AFIRO managed to obtain the second-best performance behind A-PSO with a mean fitness error of 1.19E+04. The optimal solution for Fn2 is 200. Noticeably, AFIRO is the only algorithm that can reach the optimal solution (300) for Fn3. The ability to exploit the search space can be benchmarked through unimodal func-

tions [40]. Thus, these results indicate that AFIRO performs well in exploiting the search space which leads to the success of solving unimodal functions.

The simple multimodal functions contain thirteen optimisation problems, where the functions are related to either shifting or, shifting and rotation problems. The capability of AFIRO to handle simple multimodal optimisation problems was evaluated by solving Fn4 - Fn16. Table 2 depicts that, AFIRO shows superior performance by leading in seven of thirteen functions, with a very small fitness error for each function. The functions are Fn4, Fn5, Fn6, Fn11, Fn12, Fn13, and Fn15. In Fn7, Fn14, and Fn16, AFIRO acquired second-best results behind A-PSO. However, the solutions are very close to A-PSO. Those solutions are very near to the optimal. Apart from that, AFIRO obtained the third-best results for Fn8, Fn9, and Fn10 behind A-SKF and A-PSO. Notably, larger mean fitness error values were recorded for all algorithms for Fn10 and Fn11. Both functions contain a huge number of local optima, and their second better local optimum is far from the global optimum. Thus, they are difficult to solve. The ability to explore the search space can be measured through simple multi-modal functions [40]. Hence, the results indicate that other than exploitations, AFIRO performs well in the exploration which contributes to the effectiveness of handling multimodal functions.

The competency of AFIRO to conduct hybrid optimisation problems was assessed by tackling Fn17 - Fn22. The readings in Table 2 show AFIRO performs well in solving hybrid functions. AFIRO acquired better performance than others in four of six functions, namely Fn17, Fn18, Fn20, and Fn21. A-PSO obtained a better result than AFIRO in Fn19. Nevertheless, both results are very near to the optimal solution with the mean fitness error values of 7.42 and 9.25, respectively. The optimal solution for Fn19 is 1900. With the same trend, although A-PSO achieved a better result for Fn22, the solution of AFIRO is not too far from A-PSO. Therefore, the competency of AFIRO in completing hybrid functions has proven good through these findings.

The composition of a function is the most challenging to solve as it contains a combination of unimodal, multimodal, and hybrid functions with local optima trap at the origin. The algorithm's ability in exploring and exploiting can be evaluated simultaneously through composite functions. It is due to the property of many local optima in the functions [40]. The proficiency of AFIRO to solve composition optimisation problems was tested by completing Fn23 - Fn30. The readings in Table 2 show that AFIRO is competent enough to balance between the exploration and exploitation phases by yielding better performance than other algorithms in five of eight functions, with acceptable mean fitness error values. The functions are Fn23, Fn24, Fn25, Fn27, and Fn28. Besides, AFIRO also delivered a competitive solution by achieving the second smallest reading of mean fitness value for Fn26. Although A-GSA obtained a better solution in Fn26, the result is quite close to AFIRO. The same pattern goes for Fn29. Meanwhile, for Fn30, AFIRO records the third-best result behind A-PSO and A-SKF.

Overall, AFIRO shows convincing results by outperforming eighteen of thirty functions in solving the CEC2014 test suite. Another twelve results are contributed by A-SKF (six best performances), A-PSO (five best performances), and A-GSA (one best performance).

4.1 Statistical Analysis

The performance of AFIRO against A-PSO, A-GSA, and A-SKF is measured using a Friedman test. A multiple ($N \times N$) Friedman test was conducted by considering the reduction performance distributed according to a chi-square value of 53.56, with 3 degrees of freedom (DOF). In this statistical analysis, the performance was ranked based on the mean fitness values obtained by the algorithms over the total number of 30 runs, for all 30 functions.

The mean rank is calculated for each algorithm where the smaller value of rank indicates better algorithm performance. As shown in Table 3, statistically, AFIRO has the highest rank, followed by A-PSO, A-SKF, and lastly A-GSA.

Significant differences between the algorithms have been detected by the Friedman test in this experiment. It means the null hypothesis is rejected. Thus, an additional analysis by the Holm post hoc test [39] was performed to find better-performing algorithms. Table 4 tabulates the results of the Holm post hoc test. In this condition, Holm's procedure rejects those hypotheses that have a p-value smaller than 0.025 (denoted in bold form). Based on Table 3 and Table 4, the rejection of hypotheses indicates AFIRO has significantly better performance than A-SKF and A-GSA. Meanwhile, AFIRO performs on par with A-PSO, as the p-value for the algorithms is larger than 0.025.

5. CONCLUSIONS

In the original paper, AFIRO was compared to the PSO, GA, and GWO. The comparison seems unfair as AFIRO has a different iteration strategy from the compared algorithms. Thus, this paper further investigates the performance of AFIRO among three existing metaheuristic algorithms with the same asynchronous iteration strategy, namely A-PSO, A-GSA, and A-SKF. The CEC2014 test suite had been applied as a benchmark function to evaluate the performance of the algorithms. Overall, AFIRO shows a convincing performance by leading eighteen of thirty functions. Besides, the Friedman rank shows AFIRO

Table 2: *Results of A-PSO, A-GSA, A-SKF, and AFIRO for 30 Functions.*

Fn	Ideal fitness	A-PSO		A-GSA		A-SKF		AFIRO	
		Mean fitness error	Mean fitness	Mean fitness error	Mean fitness	Mean fitness error	Mean fitness	Mean fitness error	Mean fitness
1	100	5.20E+06	5.20E+06	7.11E+08	7.11E+08	1.10E+07	1.10E+07	3.60E+05	**3.60E+05**
2	200	138.90	**338.90**	5.94E+10	5.94E+10	1.29E+06	1.29E+06	1.19E+04	1.21E+04
3	300	294.50	594.50	9.77E+04	9.80E+04	9901.00	1.02E+04	0.00	**300.00**
4	400	160.80	560.80	1.01E+04	1.05E+04	117.70	517.70	41.04	**441.04**
5	500	20.86	520.86	20.95	520.95	20.01	520.01	20.00	**520.00**
6	600	10.71	610.71	38.95	638.95	18.17	618.17	5.87	**605.87**
7	700	0.01	**700.01**	543.90	1243.90	0.08	700.08	0.03	700.03
8	800	18.57	818.57	328.50	1128.50	5.47	**805.47**	91.73	891.73
9	900	68.79	968.79	378.10	1278.10	75.26	**975.26**	89.65	989.65
10	1000	609.00	1609.00	7018.00	8018.00	162.00	**1162.00**	2535.79	3535.79
11	1100	2839.00	3939.00	7155.00	8255.00	2585.00	3685.00	2548.17	**3648.17**
12	1200	1.66	1201.66	2.45	1202.45	0.21	1200.21	0.06	**1200.06**
13	1300	0.44	1300.44	6.15	1306.15	0.36	1300.36	0.35	**1300.35**
14	1400	0.35	1400.35	175.10	1575.10	0.23	**1400.23**	0.29	1400.29
15	1500	7.25	1507.25	3.47E+05	3.49E+05	16.40	1516.40	4.70	**1504.70**
16	1600	11.22	1611.22	13.09	1613.09	10.67	**1610.67**	11.03	1611.03
17	1700	6.34E+05	6.36E+05	1.84E+07	1.84E+07	1.17E+06	1.17E+06	1.66E+04	**1.83E+04**
18	1800	4828.00	6628.00	9.81E+08	9.81E+08	8.56E+06	8.56E+06	3635.74	**5435.74**
19	1900	7.42	**1907.42**	292.40	2192.40	19.85	1919.85	9.25	1909.25
20	2000	520.90	2520.90	7.10E+04	7.30E+04	2.42E+04	2.62E+04	238.75	**2238.75**
21	2100	1.66E+05	1.68E+05	4.76E+06	4.76E+06	5.55E+05	5.57E+05	1.62E+04	**1.83E+04**
22	2200	229.40	**2429.40**	1300.00	3500.00	497.30	2697.30	304.69	2504.69
23	2300	315.90	2615.90	669.70	2969.70	316.10	2616.10	315.25	**2615.25**
24	2400	229.30	2629.30	272.60	2672.60	229.20	2629.20	221.31	**2621.31**
25	2500	209.10	2709.10	224.90	2724.90	214.30	2714.30	204.90	**2704.90**
26	2600	107.10	2707.10	106.40	**2706.40**	120.40	2720.40	109.90	2709.90
27	2700	555.60	3255.60	829.30	3529.30	547.60	3247.60	416.95	**3116.95**
28	2800	1142.00	3942.00	4703.00	7503.00	1610.00	4410.00	1073.11	**3873.11**
29	2900	1.60E+06	1.60E+06	1.17E+08	1.17E+08	1189.00	**4089.00**	9210.72	1.21E+04
30	3000	3391.00	**6391.00**	7.47E+05	7.50E+05	3848.00	6848.00	5934.07	8934.07

Table 3: *The Average Ranking of the Algorithms.*

Algorithm	Ranking
AFIRO	1.58
A-PSO	2.13
A-SKF	2.40
A-GSA	3.90

Table 4: *Holm Post Hoc Results.*

Algorithms	p	Holm
A-GSA vs AFIRO	**0.000000**	0.008333
A-PSO vs A-GSA	**0.000000**	0.010000
A-GSA vs A-SKF	**0.000007**	0.012500
A-SKF vs AFIRO	**0.012419**	0.016667
A-PSO vs AFIRO	0.089131	0.025000
A-PSO vs A-SKF	0.423711	0.050000

has better performance than others by achieving the first ranking, followed by A-PSO, A-SKF, and A-GSA. A Holm post hoc reveals that AFIRO has the same performance as A-PSO and is significantly better than A-SKF and A-GSA. In light of this evidence, it is crystal clear that the AFIRO can perform well in the same iteration strategy category.

ACKNOWLEDGMENT

This research is fully supported by the FUNDAMENTAL RESEARCH GRANT SCHEME (FRGS) awarded by the Ministry of Higher Education of Malaysia (MOHE) to Multimedia University, grant number FRGS/1/2019/ICT02/MMU/02/15. The authors would like to thank the anonymous reviewers for their constructive comments, as well as MOHE, Universiti Tun Hussein Onn Malaysia, Multimedia University, and Universiti Malaysia Pahang for their logistics support.

References

[1] A. Rodríguez et al., "Group-based synchronous-asynchronous Grey Wolf Optimizer," *Applied Mathematical Modelling*, vol. 93, pp. 226–243, 2021.

[2] N. A. A. Aziz, N. H. A. Aziz, A. A. Aziz, T. A. Rahman, W. Z. W. Ismail, and Z. Ibrahim, "Iteration Strategy and its Effect towards the Performance of Population Based Metaheuristics," *2020 IEEE 8th Conference on Systems,*

Process and Control (ICSPC), Melaka, Malaysia, pp. 58–63, 2020.

[3] D. Wu and H. Gao, "Study on Asynchronous Update Mechanism in Particle Swarm Optimization," *2014 14th International Symposium on Communications and Information Technologies (ISCIT)*, Incheon, pp. 90–93, 2015.

[4] F. A. Hashim, E. H. Houssein, K. Hussain, M. S. Mabrouk, and W. Al-Atabany, "Honey Badger Algorithm: New metaheuristic algorithm for solving optimization problems," *Mathematics and Computers in Simulation*, vol. 192, pp. 84–110, 2022.

[5] A. Mohammadi-Balani, M. Dehghan Nayeri, A. Azar, and M. Taghizadeh-Yazdi, "Golden eagle optimizer: A nature-inspired metaheuristic algorithm," *Computers & Industrial Engineering*, vol. 152, no. 107050, pp. 1–59, 2021.

[6] L. Abualigah, A. Diabat, S. Mirjalili, M. Abd Elaziz, and A. H. Gandomi, "The Arithmetic Optimization Algorithm," *Computer Methods in Applied Mechanics and Engineering*, vol. 376, no. 113609, pp. 1–38, 2021.

[7] K. M. Ong, P. Ong, and C. K. Sia, "A Carnivorous Plant Algorithm for Solving Global Optimization Problems," *Applied Soft Computing*, vol. 98, no. 106833, 2021.

[8] V. Hayyolalam, A. Asghar, and P. Kazem, "Black Widow Optimization Algorithm: A novel meta-heuristic approach for solving engineering optimization problems ⋆," *Engineering Applications of Artificial Intelligence*, vol. 87, pp. 952–1976, 2020.

[9] N. A. Ab Aziz, Z. Ibrahim, M. Mubin, and S. Sudin, "Adaptive Switching Gravitational Search Algorithm: An Attempt to Improve Diversity of Gravitational Search Algorithm Through Its Iteration Strategy," *Sadhana*, vol. 42, no. 7, pp. 1103–1121, 2017.

[10] N. A. Ab Aziz and Z. Ibrahim, "Asynchronous Particle Swarm Optimization for Swarm Robotics," *Procedia Engineering*, vol. 41, pp. 951–957, 2012.

[11] N. A. A. Aziz, S. Sudin, M. Mubin, S. W. Nawawi, and Z. Ibrahim, "A Random Synchronous- Asynchronous Particle Swarm Optimization Algorithm with a New Iteration Strategy," *ARPN Journal of Engineering and Applied Sciences*, vol. 10, no. 21, pp. 9937–9942, 2015, [Online]. Available: `https://umexpert.um.edu.my/file/publication/00005798\_131223.pdf`.

[12] N. A. Ab Aziz, M. Mubin, M. S. Mohamad, and K. Ab Aziz, "A Synchronous-asynchronous Particle Swarm Optimisation Algorithm.," *The Scientific World Journal*, vol. 2014, no. 123019, 2014.

[13] J. Rada-Vilela, M. Zhang, and W. Seah, "A Performance Study on Synchronicity and Neighborhood Size in Particle Swarm Optimization," Soft Comput., vol. 17, no. 6, pp. 1019–1030, 2013, doi: 10.1007/s00500-013-1015-9.

[14] J. Rada-Vilela, M. Zhang, and W. Seah, "A Performance Study on Synchronous and Asynchronous Updates in Particle Swarm Optimization," *Proceedings of the 13th annual conference on Genetic and evolutionary computation*, pp. 21-28, 2011.

[15] J. Rada-Vilela, M. Zhang, and W. Seah, "Random Asynchronous PSO," *The 5th International Conference on Automation, Robotics and Applications*, Wellington, New Zealand, pp. 220–225, 2011.

[16] A. Carlisle and G. Dozier, "An Off-The-Shelf PSO," Popul. English Ed., vol. 1, pp. 1–6, 2001, [Online]. Available: `http://antho.huntingdon.edu/publications/Off-The-Shelf\_PSO.pdf`.

[17] N. A. Ab. Aziz, N. H. Abd Aziz, T. Ab Rahman, N. Mokhtar, and M. Mubin, "Simultaneous Model Order and Parameter Estimation (SMOPE) based on Random Asynchronous Particle Swarm Optimization," *Mekatronika*, vol. 01, no. 02, pp. 66–71, 2019.

[18] N. A. Ab. Aziz, N. H. Abd Aziz, T. Ab Rahman, N. Mokhtar, and M. Mubin, "Random Synchronous Asynchronous PSO – A Particle Swarm Optimization Algorithm with a New Iteration Strategy," *Mekatronika*, vol. 1, no. 2, pp. 81–92, 2019.

[19] N. A. Ab Aziz, Z. Ibrahim, S. W. Nawawi, S. Sudin, M. Mubin, and K. Ab. Aziz, "Synchronous Gravitational Search Algorithm vs Asynchronous Gravitational Search Algorithm: A Statistical Analysis," *New Trends Softw. Methodol. Tools Tech.*, vol. 265, pp. 160–169, 2014.

[20] N. A. Ab Aziz, S. W. Nawawi, Z. Ibrahim, I. Ibrahim, M. T. Mohd Zaidi, and M. Mubin, "Synchronous vs Asynchronous Gravitational Search Algorithm," *2013 1st International Conference on Artificial Intelligence, Modelling and Simulation*, Kota Kinabalu, Malaysia, pp. 37–42, 2014.

[21] N. A. Ab. Aziz, Z. Ibrahim, N. H. Abdul Aziz, and T. Ab Rahman, "Asynchronous Simulated Kalman Filter Optimization Algorithm," *International Journal of Engineering & Technology*, vol. 7, no. 4.27, pp. 44–49, 2018.

[22] N. A. Ab. Aziz, T. Ab Rahman, and N. H. Abdul Aziz, "Fitness-evaluated Adaptive Switching Simulated Kalman Filter Algorithm with Randomness," *Mekatronika*, vol. 1, no. 2, pp. 45–65, 2019.

[23] V. Coleman, "The DEME Model: An Asyncronous Genetic Algorithm.," 1989. [Online].

Available: https://web.cs.umass.edu/publication/docs/1989/UM-CS-1989-035.pdf.

[24] S. Xue, J. Zhang, and J. Zeng, "Parallel Asynchronous Control Strategy for Target Search with Swarm Robots," *International Journal of Bio-Inspired Computation*, vol. 1, no. 3, pp. 151–163, 2009.

[25] N. Mohd Sabri, M. Puteh, and M. R. Mahmood, "A Review of Gravitational Search Algorithm," *Int. J. Adv. Soft Comput. Appl.*, vol. 5, no. 3, pp. 1–39, 2013, [Online]. Available: http://home.ijasca.com/data/documents/ijasc08_published.pdf.

[26] T. Ab Rahman, "Finite Impulse Response Optimizers for Solving Optimization Problems," *Universiti Malaysia Pahang*, 2019.

[27] T. Ab Rahman, Z. Ibrahim, N. A. Ab Aziz, S. Zhao, and N. H. Abdul Aziz, "Single-Agent Finite Impulse Response Optimizer for Numerical Optimization Problems," in *IEEE Access*, vol. 6, pp. 9358-9374, 2018.

[28] T. Ab Rahman et al., "A Study on the Effect of Local Neighbourhood Parameter towards the Performance of SAFIRO," *International Journal of Engineering & Technology*, vol. 7, no. 4.27, pp. 30–37, 2018.

[29] T. Ab Rahman, Z. Ibrahim, N. A. Ab. Aziz, N. H. Abdul Aziz, M. S. Mohamad, and M. I. Shapiai, "Evaluation of Different Horizon Lengths in Single-agent Finite Impulse Response Optimizer," *2019 International Conference on Computer and Information Sciences (ICCIS)*, Sakaka, Saudi Arabia, pp. 1-7, 2019.

[30] T. Ab Rahman, N. A. Ab. Aziz, and N. H. Abdul Aziz, "Single-agent Finite Impulse Response Optimizer vs Simulated Kalman Filter Optimizer," *Mekatronika*, vol. 01, no. 02, pp. 15–22, 2019.

[31] J. J. Liang, B. Y. Qu, and P. N. Suganthan, "Problem Definitions and Evaluation Criteria for the CEC 2014 Special Session and Competition on Single Objective Real-Parameter Numerical Optimization," 2013. [Online]. Available: http://web.mysites.ntu.edu.sg/epnsugan/PublicSite/SharedDocuments/Forms/AllItems.aspx?RootFolder=\%2Fepnsugan\%2FPublicSite\%2FSharedDocuments\%2FCEC-2014&FolderCTID=\&View=\%7BDAF31868-97D8-4779-AE49-9CEC4DC3F310\%7D.

[32] Y. S. Shmaliy, S. Khan, and S. Zhao, "Ultimate Iterative UFIR Filtering Algorithm," *Measurement*, vol. 92, pp. 236–242, 2016.

[33] C. Lastre-Dominguez, Y. S. Shmaliy, O. Ibarra-Manzano, M. Vazquez-Olguin, and L. J. Morales-Mendoza, "Unbiased FIR Denoising of ECG Data for Features Extraction," 2017.

[34] V. M. Olguin, Y. S. Shmaliy, C. Ki Ahn, and O. G. Ibarra Manzano, "Blind Robust Estimation With Missing Data for Smart Sensors Using UFIR Filtering," in *IEEE Sensors Journal*, vol. 17, no. 6, pp. 1819–1827, 2017.

[35] K. Uribe-Murcia, Y. S. Shmaliy, and J. A. Andrade-lucio, "UFIR Filtering for GPS-Based Tracking over WSNs with Delayed and Missing Data," *Journal of Electrical and Computer Engineering*, vol. 2018, 7456010, 2018.

[36] M. Vazquez-Olguin, Y. Semenovic Shmaliy, O. Ibarra-Manzano, and L. J. Lastre-Dominguez, Carlos Morales-Mendoza, "Design of an Unbiased Finite Impulse Response Filter for a Smart Sensor to Estimate State of CO Concentration," *2017 IEEE International Autumn Meeting on Power, Electronics and Computing (ROPEC)*, Ixtapa, Mexico, pp. 1-6, 2017.

[37] N. A. Ab. Aziz, "An Adaptively Switching Iteration Strategy for Population Based Metaheuristics," *University of Malaya*, 2016.

[38] N. A. Ab. Aziz, M. Mubin, Z. Ibrahim, and S. W. Nawawi, "Statistical Analysis for Swarm Intelligence - Simplified," *International Journal of Future Computer and Communication*, vol. 4, no. 3, pp. 193–197, 2015.

[39] J. Derrac, S. García, D. Molina, and F. Herrera, "A Practical Tutorial on the use of Nonparametric Statistical Tests as A Methodology for Comparing Evolutionary and Swarm Intelligence Algorithms," *Swarm and Evolutionary Computation*, vol. 1, no. 1, pp. 3–18, 2011.

[40] S. Mirjalili, S. M. Mirjalili, and A. Lewis, "Grey Wolf Optimizer," *Advances in Engineering Software*, vol. 69, pp. 46–61, 2014.

Tasiransurini Ab Rahman holds a certificate and diploma in electronic engineering from the Polytechnic of Sultan Haji Ahmad Shah (POLISAS), Pahang, Malaysia. She obtained her B.Eng. (Hons) in electrical engineering from Kolej Universiti Tun Hussein Onn (KUiTTHO), Johor, Malaysia, M.Eng. degree in electrical engineering (communication and computer network) from Universiti Kebangsaan Malaysia (UKM), Bangi, Malaysia, and her PhD degree in electronic engineering from Universiti Malaysia Pahang (UMP), Malaysia. She is currently a senior lecturer in the Department of Electronic Engineering, Faculty of Electrical and Electronic Engineering (FKEE), Universiti Tun Hussein Onn Malaysia (UTHM). She is also a member of FKEE's Computational, Signal, Imaging and Intelligence Focus Group. She is one of the inventors of a single-agent finite impulse response optimizer (SAFIRO), which is an optimization algorithm based on the iterative unbiased finite impulse response (UFIR) filter framework. Her current research interests include computational intelligence and its application in engineering.

Nor Azlina Ab. Aziz graduated with a B.Eng. (Hons) in electronics, majoring in computers and M.Eng.Sc from Multimedia University (MMU), Malaysia and a PhD degree from the University of Malaya (UM), Malaysia. Both her doctoral and master research works were in the field of computational intelligence. She is an associate professor in the Faculty of Engineering & Technology, MMU, Melaka, Malaysia. She is currently a deputy director of MMU's Research Management Centre and a member of the Engineering Computational Intelligence Special Interest Group. She is one of the inventors of SAFIRO and the simulated Kalman filter algorithm (SKF), which is an optimization algorithm based on the Kalman filter framework. Her research interests include the fundamental aspects and applications of computational intelligence in engineering.

Zuwairie Ibrahim received his B.Eng. in electrical engineering and an M.Eng. degree in image processing from Universiti Teknologi Malaysia, Johor, Malaysia, in 2000 and 2003, respectively. In 2006, he received his PhD degree in DNA computing from Meiji University, Tokyo, Japan. From 2002 to 2008, he was a lecturer at Universiti Teknologi Malaysia, Johor, Malaysia. He was then promoted to senior lecturer in 2008 and served Universiti Teknologi Malaysia until 2012. In 2012, he transferred to Universiti Malaysia Pahang, Pahang, Malaysia, to work as an associate professor. He is one of the inventors of the SKF algorithm and SAFIRO. His research interests include the fundamentals and applications of computational intelligence, specifically, particle swarm optimization, ant colony optimization, gravitational search algorithm, and black hole algorithms.