Enhancing Global Optimization Performance of Arithmetic Optimization Algorithm with a Modified Population Initialization Scheme

Tin Chang Ting¹, Hameedur Rahman², Meng Choung Chiong¹, Mohamed Khan Aftab Ahamed Khan³, Cik Suhana Hassan¹, Farah Adilah Jamaludin¹, Sew Sun Tiang¹, Wei Hong Lim¹

¹Faculty of Engineering, Technology and Built Environment, UCSI University, Kuala Lumpur 56000, Malaysia
²Faculty of Computing and Artificial Intelligence, Air University, Islamabad Capital Territory 44000, Pakistan

Email: 1002058096@ucsiuniversity.edu.my, rameedur@gmail.com, ChiongMC@ucsiuniversity.edu.my, mohamedkhan@ucsiuniversity.edu.my, suhana@ucsiuniversity.edu.my, FarahAdilah@ucsiuniversity.edu.my, tiangss@ucsiuniversity.edu.my, limwh@ucsiuniversity.edu.my

Abstract

Arithmetic Optimization Algorithm (AOA) is widely used to solve global optimization problems. However, it often faces premature convergence challenges in complex optimization scenarios. A key factor affecting AOA’s performance is the solution quality of the initial population. The conventional initialization scheme, despite its prevalence, lacks reliability in ensuring high-quality solutions due to inherent stochastic processes. To address this issue, we propose a modified initialization scheme that improves initial population quality by integrating chaotic maps and oppositional-based learning. Through extensive simulation studies, we demonstrate that the enhanced AOA, equipped with this new initialization scheme, exhibits superior performance in solving a range of benchmark functions with improved accuracy.

Keywords: Arithmetic optimization algorithm, Population initialization, Chaotic map, Oppositional-based learning

1. Introduction

Optimization is pivotal in real-life engineering design, where it seeks the most effective solutions while accommodating various stakeholders' criteria. Accurate problem definition and modeling are essential for resolving these design challenges. This process includes establishing clear objectives, identifying both technical and non-technical constraints, and optimizing decision variables. Nonetheless, real-world engineering design problems frequently present complexity and challenges due to factors like high dimensionality, numerous constraints, conflicting objectives, and data uncertainty.

In the era of Industrial Revolution 4.0 (IR 4.0), engineering systems have become more intricate, often involving non-differentiable, nonlinear, multimodal, and non-continuous functions. While traditional optimization methods such as Newton’s method are prevalent, they face limitations in addressing these complex, real-world engineering design challenges. These methods often depend heavily on initial solutions and are typically suited to specific problem types, limiting their scalability for diverse, complicated optimization tasks [1]. Most traditional methods also rely on gradient information to identify optimal solutions, which is not always feasible in real-world scenarios that frequently present as black box functions [2]. Furthermore, the limited global search capability of traditional optimization methods increases the risk of premature convergence.

Recognizing the limitations of traditional optimization methods, there is an urgent need in the Industrial Revolution 4.0 era to develop more intelligent and robust optimization algorithms. These algorithms must be capable of providing efficient solutions to a broad spectrum of increasingly complex optimization problems. Metaheuristic Search Algorithms (MSAs) have emerged as effective solutions, drawing on search mechanisms inspired by natural phenomena. Existing MSAs fall into four categories based on their natural inspirations [3]: evolutionary algorithms, swarm intelligence, human-based algorithms, and physics-based algorithms. Compared to traditional methods, MSAs offer several advantages, including potent global search capabilities, straightforward implementation, and enhanced scalability. They exploit the unique strengths of their respective inspirations, enabling them to effectively address a variety of complex optimization challenges as outlined in [4], [5], [6], [7], [8], [9], [10], [11], [12], [13].

The Arithmetic Optimization Algorithm (AOA) [14], introduced in 2021, is a physics-based algorithm with search mechanisms inspired by the four basic arithmetic operations: division, multiplication, addition, and subtraction. These operations vary in their exploration and exploitation strengths, essential for solving optimization problems. Like other MSAs, achieving a proper balance between exploration and exploitation is crucial for enhancing AOA’s performance. Consequently, several enhancement schemes, including hybridization
The quality of the initial population is a key factor in determining the performance of MSAs. Despite numerous variants of the AOA being introduced in recent years, many still rely on conventional methods to randomly generate their initial populations. This conventional approach, while simple to implement, does not intelligently leverage environmental information around the solution regions during initialization [3]. As a result, it often mistakenly places some solutions in local or suboptimal regions, leading to premature convergence. Additionally, if initial solutions are far from the global optimum, the algorithm’s convergence speed may be compromised. These drawbacks can significantly impact the robustness and effectiveness of AOA in addressing complex real-world optimization challenges.

This paper presents the Multi-Chaotic Dynamic Oppositional Learning (MCDOL) module as an enhancement to the AOA’s population initialization scheme, culminating in a new variant named MCDOL- AOA. The MDCOL module is designed to produce an initial population with enhanced solution quality, both in terms of accuracy and convergence rate, by harnessing the strengths of multiple chaotic maps and the dynamic oppositional-based learning (DOL) mechanism. Notably, the diverse ergodic properties of various chaotic maps are employed for a comprehensive solution space search, enhancing population diversity and mitigating the risk of initializing solutions in local optima. Additionally, the inclusion of DOL in MCDOL module promotes more rapid global optimum identification by expanding the exploration of solution regions. The effectiveness of MCDOL-AOA is evaluated through benchmarking against the original AOA across a range of benchmark functions with varying characteristics.

2. Methodology

2.1. Proposed MCDOL module

In this subsection, we detail how the proposed MCDOL module generates an initial population of superior quality. The MCDOL module leverages multiple chaotic maps to create a diverse set of initial solutions, forming a chaotic population. This approach contrasts with conventional initialization schemes, which often result in poorly distributed initial solutions. The deterministic nature of chaotic maps, coupled with their ability to exhibit stochastic behavior through ergodic properties, allows for a more comprehensive exploration of the solution space. Unlike some prior studies [26], [27], [28] that rely on a single chaotic map, this paper explores the benefits of using multiple chaotic maps for population initialization. We hypothesize that each map’s unique ergodic characteristics can enhance the algorithm’s performance for specific optimization problems. Therefore, the synergistic integration of multiple chaotic maps promises to significantly improve the algorithm’s robustness in addressing various complex problems.

The MCDOL module incorporates five distinct chaotic maps — Circle, Logistic, Piecewise, Sine, and Tent — for generating a chaotic population. Each chaotic map is randomly chosen to generate different dimensions of the initial solutions within this population. To facilitate this, define \( \vartheta_t \) as the output of a chaotic variable at the \( t \)-th iteration, where \( t = 1, \ldots, T \). The population initialization in the MCDOL module, using these multiple chaotic maps, follows five criteria: (a) the Circle map is used if \( 0 \leq \vartheta_0 < 0.2 \), (b) the Logistic map for \( 0.2 \leq \vartheta_0 < 0.4 \), (c) the Piecewise map for \( 0.4 \leq \vartheta_0 < 0.6 \), (d) the Sine map for \( 0.6 \leq \vartheta_0 < 0.8 \), and (e) the Tent map for \( 0.8 \leq \vartheta_0 \leq 1.0 \). The specific equations for these five chaotic maps, Circle, Logistic, Piecewise, Sine, and Tent, are detailed in Eqs. (1) to (5).

\[
\vartheta_{t+1} = \text{mod} \left( \vartheta_t + 0.2 - \frac{0.5}{2\pi} \sin(2\pi \vartheta_t), 1 \right) \quad (1)
\]

\[
\vartheta_{t+1} = 4 \vartheta_t (1 - \vartheta_t) \quad (2)
\]

\[
\vartheta_{t+1} = \begin{cases} 
\frac{\vartheta_t}{p}, & 0 \leq \vartheta_t < p \\
\frac{\vartheta_t - p}{1 - p - \vartheta_t}, & 0.5 \leq \vartheta_t < 1 - p \\
\frac{1 - \vartheta_t}{1 - p}, & 1 - p \leq \vartheta_t < 1 
\end{cases} \quad (3)
\]

\[
\vartheta_{t+1} = \sin (\pi \vartheta_t) \quad (4)
\]

\[
\vartheta_{t+1} = \begin{cases} 
\vartheta_t, & \vartheta_t < 0.7 \\
10 \vartheta_t, & 0.7 \leq \vartheta_t \\
\frac{3}{10}(1 - \vartheta_t), & \vartheta_t \geq 0.7 
\end{cases} \quad (5)
\]

Let \( X^U_d \) and \( X^L_d \) denote the lower and upper bounds of the \( d \)-th dimensional decision variable, respectively, for \( d = 1, \ldots, D \). At the final iteration \( t = T \), a chaotic value \( \vartheta_T \) is generated from a randomly selected chaotic map. This value initializes the \( d \)-th dimension of each \( n \)-th chaotic solution, as outlined in Eq. (6). The resulting chaotic population, denoted as \( P^C = [X^C_{1}, \ldots, X^C_{D}] \), encompasses all solution members formed using multiple chaotic maps.

\[
X^C_{n,d} = X^U_d + \vartheta_T (X^U_d - X^L_d) \quad (6)
\]

While various chaotic maps exhibit differing levels of robustness to local optima, they may still generate initial solutions distant from the global optimum, potentially slowing the algorithm’s convergence. To address this issue, a DOL operator is applied to the chaotic population \( P^C \), generating opposite solutions for each \( n \)-th chaotic solution. This DOL operator is expected to broaden the initial population’s coverage of the solution space.
Enhancing Global Optimization Performance

The proposed MCDOL-AOA algorithm, denoted as \( P^0 \), is determined using Eq. (5), corresponding to \( X^0_{n,d} \), where \( r_1, r_2 \in [0, 1] \). Consequently, this generates an opposition population, \( P^o = [X^U_1, ..., X^U_n, ..., X^U_N] \).

\[
X^o_{n,d} = X^C_{n,d} + r_1 \left[ r_2(X^U_d + X^d_d) - X^C_{n,d} \right]
\] (7)

The two populations, one generated using multiple chaotic maps \( P^c \) and the other using the DOL operator \( P^0 \), are merged to form a combined population set \( P^c \cup P^0 \) with a total size of \( 2N \). The fitness of each solution member in this merged population set is evaluated according to the predefined objective function. These solution members are then ordered based on their fitness values, from the best to worst performing ones. The top \( N \) solution members from this ordered population set \( P^c \cup P^0 \) are chosen as the initial population for the proposed MCDOL-AOA algorithm, represented as \( P = [X_1, ..., X_n, ..., X_N] \).

2.2. Iterative Search Processes of MCDOL-AOA

After generating an initial population \( P \) of superior quality using the MCDOL module, each \( n \)-th solution in MCDOL-AOA is iteratively updated through search mechanisms similar to those in the original AOA.

During each iteration, the Math Optimizer Accelerated (MOA) function value is adjusted in MCDOL-AOA to alternate between exploration and exploitation phases:

\[
MOA(C_{iter}) = \text{Min} + C_{iter} \frac{\text{Max} - \text{Min}}{M_{iter}}
\] (8)

where \( C_{iter} \) and \( M_{iter} \) represent the current and maximum iteration counts; \( \text{Min} \) and \( \text{Max} \) are the minimum and maximum values of MOA. Simultaneously, the Math Optimizer Probability (MP) function, which dictates the search range for each solution and is influenced by the critical parameter \( \theta \) for exploitation efficiency, is updated as:

\[
MP(C_{iter}) = 1 - \left( \frac{C_{iter}}{M_{iter}} \right)^{1/\theta}
\] (9)

A random number \( \text{rand1} \) determines the search strategy (exploration or exploitation) at each iteration for updating the \( d \)-th dimension of the \( n \)-th solution, \( X_{n,d} \). During the exploration phase (\( \text{rand1} > \text{MOA} \)), either the Multiplication or Division operator is selected:

\[
X_{n,d}(C_{iter} + 1) = \begin{cases} 
\text{best}_d + (\text{MP} + \varepsilon) \times [(X^C_d - X^d_d)\mu + X^U_d], & \text{rand2} < 0.5 \\
\text{best}_d \times \text{MP} \times [(X^C_d - X^d_d)\mu + X^U_d], & \text{Otherwise}
\end{cases}
\] (10)

where \( \text{rand2} \) is a random number between 0 and 1; \( \text{best}_d \) denotes the \( d \)-th dimension of the current best solution; \( \varepsilon \) is a small positive number to prevent division by zero, and \( \mu \) is a control parameter.

In the exploitation phase (\( \text{rand1} \leq \text{MOA} \)), either the Addition or Subtraction operator is used to update \( X_{n,d} \):

\[
X_{n,d}(C_{iter} + 1) = \begin{cases} 
\text{best}_d - \text{MP} \times [(X^C_d - X^d_d)\mu + X^U_d], & \text{rand2} < 0.5 \\
\text{best}_d + \text{MP} \times [(X^C_d - X^d_d)\mu + X^U_d], & \text{Otherwise}
\end{cases}
\] (11)

As illustrated in Fig. 1, MCDOL-AOA continues this iterative search, following Eqs. (8) to (11), until pre-set termination criteria are fulfilled. Upon completion, optimal decision variables in the best solution are decoded to solve the specific optimization problems.

**MCDOL-AOA for Global Optimization**

**Inputs:** \( D, N, M_{iter}, T, \text{Max, Min, } \theta \)

01: Initialize \( P^c \to \emptyset \), \( P^0 \to \emptyset \) and \( C_{iter} \to 0 \);  
02: for each \( n \)-th solution do  
03: \quad for \( d \)-th dimension do  
04: \quad \quad Randomly initialize \( \theta_d \in [0,1] \), where \( t = 0 \);  
05: \quad \quad if \( 0 \leq \theta_d < 0.2 \) then  
06: \quad \quad \quad Select Circle map in Eq. (1);  
07: \quad \quad elseif \( 0.2 \leq \theta_d < 0.4 \) then  
08: \quad \quad \quad Select Logistic map in Eq. (2);  
09: \quad \quad elseif \( 0.4 \leq \theta_d < 0.6 \) then  
10: \quad \quad \quad Select Piecewise map in Eq. (3);  
11: \quad \quad elseif \( 0.6 \leq \theta_d < 0.8 \) then  
12: \quad \quad \quad Select Sine map in Eq. (4);  
13: \quad \quad elseif \( 0.8 \leq \theta_d \leq 1.0 \) then  
14: \quad \quad \quad Select Tent map in Eq. (5);  
15: \quad \quad end if  
16: \quad \quad while \( t \leq T \) do  
17: \quad \quad \quad Update \( \theta_d \) with the selected chaotic map;  
18: \quad \quad \quad \( t \leftarrow t + 1 \);  
19: \quad \quad \quad end while  
20: \quad \quad Generate \( X^C_{n,d} \) using Eq. (6);  
21: \quad \quad Generate \( X^U_{n,d} \) using Eq. (7);  
22: \quad \quad end for  
23: \quad \quad Update \( P^c \leftarrow P^c \cup X^C_{n,d} \) and \( P^0 \leftarrow P^0 \cup X^U_{n,d} \);  
24: \quad end for  
25: \quad Merge the two population sets as \( P^c \cup P^0 \);  
26: \quad Evaluate the fitness values of all solutions stored within \( P^c \cup P^0 \);  
27: \quad Rearrange the solutions stored within \( P^c \cup P^0 \) from best to worst based on their fitness values;  
28: \quad Select the top \( N \) solutions from the sorted \( P^c \cup P^0 \) as the initial population, i.e., \( P = [X_1, ..., X_n, ..., X_N] \);  
29: \quad Assign the first solution of \( P \) and its fitness as \( \text{best} \) and \( f(\text{best}) \), respectively;  
30: \quad while \( C_{iter} \leq M_{iter} \) do  
31: \quad \quad Update \( \text{MOA} \) and \( \text{MP} \) with Eqs. (8) and (9);  
32: \quad \quad for each \( n \)-th solution do  
33: \quad \quad \quad if \( \text{rand1} > \text{MOA} \) then \(^*/\text{Exploration}*/\)  
34: \quad \quad \quad \quad Update \( X_{n,d}(C_{iter} + 1) \) with Eq. (10);  
35: \quad \quad \quad elseif \(^*/\text{Exploration}*/\)  
36: \quad \quad \quad \quad Update \( X_{n,d}(C_{iter} + 1) \) with Eq. (11);  
37: \quad \quad \quad end if  
38: \quad \quad \quad Fitness evaluation of \( X_{n}(C_{iter} + 1) \);  
39: \quad \quad \quad Update the \( X_n, f(X_n) \), \( \text{best} \) and \( f(\text{best}) \), with greedy selection method;  
40: \quad \quad \quad end while  
41: \quad \quad \quad C_{iter} \leftarrow C_{iter} + 1;  
42: \quad \quad end for  
43: \quad end while  
Output: \( f(\text{best}) \).

Fig.1 Workflow of proposed MCDOL-AOA in solving the global optimization problems.
3. Results and Discussions

3.1. Simulations settings

In this section, we compare the performance of MCDOL-AOA with the original AOA using 23 benchmark functions, each with distinct characteristics as outlined in [17]. Functions F1 to F7 are scalable unimodal functions, while F8 to F13 are scalable multimodal functions, all set at a dimension size of $D = 100$. Functions F14 to F23, on the other hand, are fixed-dimension multimodal functions with $D$ ranging from 2 to 6. Both MCDOL-AOA and the original AOA were implemented on MATLAB 2021a, running on a personal computer equipped with an Intel® Core™ i7-HQ CPU at 2.50 GHz and 16 GB RAM. For both algorithms, the population size and maximum number of iterations are set at 30 and 1000, respectively.

3.2. Performance analysis

The evaluation of MCDOL-AOA's performance, and its comparison with the original AOA across all 23 benchmark functions, is detailed in Table 1. We employ two performance metrics: mean error ($E_{\text{mean}}$) and standard deviation (SD), to assess the algorithms' accuracy and consistency in solving these functions. Lower values of $E_{\text{mean}}$ and SD are preferable, indicating the algorithm's consistent and accurate resolution of the given benchmark functions.

<table>
<thead>
<tr>
<th>Fun</th>
<th>Original AOA</th>
<th>MCDOL-AOA</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$E_{\text{mean}}$</td>
<td>$SD$</td>
</tr>
<tr>
<td>F1</td>
<td>2.048e-04</td>
<td>5.662e-05</td>
</tr>
<tr>
<td>F2</td>
<td>0.0112</td>
<td>0.0115</td>
</tr>
<tr>
<td>F3</td>
<td>0.0915</td>
<td>0.0229</td>
</tr>
<tr>
<td>F4</td>
<td>0.0523</td>
<td>0.0076</td>
</tr>
<tr>
<td>F5</td>
<td>98.090</td>
<td>0.0899</td>
</tr>
<tr>
<td>F6</td>
<td>15.131</td>
<td>0.7747</td>
</tr>
<tr>
<td>F7</td>
<td>2.609e-05</td>
<td>3.060e-05</td>
</tr>
<tr>
<td>F8</td>
<td>-1.53e-04</td>
<td>646.631</td>
</tr>
<tr>
<td>F9</td>
<td>6.244e-05</td>
<td>1.478e-05</td>
</tr>
<tr>
<td>F10</td>
<td>0.0016</td>
<td>1.452e-04</td>
</tr>
<tr>
<td>F11</td>
<td>0.0085</td>
<td>0.0249</td>
</tr>
<tr>
<td>F12</td>
<td>0.891</td>
<td>0.0655</td>
</tr>
<tr>
<td>F13</td>
<td>9.906</td>
<td>0.0030</td>
</tr>
<tr>
<td>F14</td>
<td>9.864</td>
<td>4.317</td>
</tr>
<tr>
<td>F15</td>
<td>0.0055</td>
<td>0.0125</td>
</tr>
<tr>
<td>F16</td>
<td>-1.032</td>
<td>5.375e-12</td>
</tr>
<tr>
<td>F17</td>
<td>0.398</td>
<td>5.620e-07</td>
</tr>
<tr>
<td>F18</td>
<td>18.3</td>
<td>28.080</td>
</tr>
<tr>
<td>F19</td>
<td>-3.863</td>
<td>5.441e-05</td>
</tr>
<tr>
<td>F20</td>
<td>-3.294</td>
<td>0.0511</td>
</tr>
<tr>
<td>F21</td>
<td>-8.554</td>
<td>2.522</td>
</tr>
<tr>
<td>F22</td>
<td>-8.077</td>
<td>3.155</td>
</tr>
</tbody>
</table>

The results in Table 1 showcase MCDOL-AOA's impressive accuracy, outperforming or matching the original AOA in 16 of the 23 benchmark functions, as evidenced by lower $E_{\text{mean}}$ values. In the scalable unimodal function's category (F1 to F7), MCDOL-AOA outperforms the original AOA in 5 of the 7 functions based on $E_{\text{mean}}$. For the 6 scalable multimodal functions (F8 to F13), MCDOL-AOA surpasses the original AOA in 4 functions (F8, F9, F12, and F13) and equals the original AOA in one function (F10). When dealing with the fixed-dimension multimodal functions (F14 to F23), MCDOL-AOA outperforms the original AOA in 4 functions (F15, F18, F22, and F23) and equals its performance in 3 other functions (F16, F17, and F19).

Table 1: Performance comparison of MCDOL-AOA and original AOA using 23 benchmark functions with different characteristics.

Analysis of the simulation results in Table 1 indicates that MCDOL-AOA significantly outperforms the original AOA in solving larger-scale benchmark functions (F1 to F13) with $D = 100$, evidenced by lower $E_{\text{mean}}$ values in 9 out of 13 functions. However, the performance of MCDOL-AOA is relatively on par with the original AOA in fixed-dimension problems (F14 to F23) with smaller dimensions ($D = 2, 3, 4, \text{ and } 6$). These findings suggest that the MCDOL module in MCDOL-AOA effectively produces initial populations with superior fitness and diversity compared to the original AOA, yielding enhanced optimization results. The benefits of the MCDOL module are particularly notable in larger dimensional sizes (e.g., $D = 100$), as utilized in this study. The multiple chaotic maps’ non-repetitive and ergodic properties within the MCDOL module facilitate a more exhaustive exploration of the solution space across various problem types, thereby minimizing the risk of local optima entrapment and premature convergence. Additionally, the DOL mechanism within the MCDOL module contributes to faster algorithmic convergence by broadening solution space exploration through the generation of opposite solutions from those initiated by the chaotic maps.

4. Conclusion

In this paper, we introduce an enhanced version of the AOA, termed MCDOL-AOA, designed to solve complex global optimization problems with improved accuracy. The novelty of MCDOL-AOA resides in its integration of multiple chaotic maps and DOL mechanisms into a modified initialization scheme, the MCDOL module. This module aims to generate an initial population of superior quality, focusing on fitness and diversity. Simulation results demonstrate that MCDOL-AOA, benefiting from the enhanced initial population quality provided by the MCDOL module, surpasses the original AOA in solving 23 distinct benchmark functions. Notably, the performance gains of MCDOL-AOA are more pronounced in solving functions with larger dimensions, specifically $D = 100$. This underscores the efficacy of the multiple chaotic maps and DOL mechanisms within the MCDOL module in boosting the algorithm’s robustness against premature convergence and enhancing its convergence speed. Future work will explore the application of MCDOL-AOA to real-world engineering optimization problems, including machine

©The 2024 International Conference on Artificial Life and Robotics (ICAROB2024), J:COM HorutoHall, Oita, Japan, 2024
learning model training and scheduling optimization, to assess its practicality and effectiveness.

Acknowledgements

This work was supported by UCSI University’s Research Excellence & Innovation Grant (REIG) with project code of REIG-FETBE-2022/038 and Billion Prima Sdn. Bhd.’s Industry Research Grant with project code of IND-FETBE-2023/006.

References

6. E. S. Ghith and F. A. A. Tolba, Real-time implementation of an enhanced PID controller based on marine predator algorithm (MPA) for micro-robotics system, 2022 3rd International Conference on Artificial Intelligence, Robotics and Control (AIRC), Cairo, Egypt, 2022.
### Authors Introduction

<table>
<thead>
<tr>
<th>Author</th>
<th>Bio</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mr. Tin Chang Ting</td>
<td>He is currently pursuing the Bachelor of Mechatronics Engineering with Honours in Faculty of Engineering, Technology and Built Environment, UCSI University, Malaysia. His research interests are machine learning, deep learning, and optimization algorithm.</td>
</tr>
<tr>
<td>Dr. Hameedur Rahman</td>
<td>He is an Associate Professor in Faculty of Computing and AI at AIR University in Pakistan. He received his PhD in Computer Science from Universiti Kebangsaan Malaysia in 2018. His research interests are Virual/Augmented Reality, Image Processing, Data Mining, Artificial Intelligence, Natural Language Processing and CyberSecurity.</td>
</tr>
<tr>
<td>Dr. Meng Choung Chiong</td>
<td>He is an Assistant Professor in Faculty of Engineering at UCSI University in Malaysia. He received his PhD in Chemical Engineering from Universiti Teknologi Malaysia. His research interests are artificial intelligence, mechatronics and alternative fuels combustion.</td>
</tr>
<tr>
<td>Dr. Mohamed Khan Afthab Ahmed Khan</td>
<td>He is an Assistant Professor in Faculty of Engineering at UCSI University in Malaysia. His research interests are artificial intelligence, robotics, control, medical rehabilitation and power electronics.</td>
</tr>
<tr>
<td>Dr. Cik Suhana Hassan</td>
<td>She is an Assistant Professor in Faculty of Engineering at UCSI University in Malaysia. She obtained her PhD in Mechanical Engineering from Universiti Putra Malaysia. Her research interests are artificial intelligence, bio composite and finite element analysis.</td>
</tr>
<tr>
<td>Dr. Farah Adilah Binti Jamaluddin</td>
<td>She is an Assistant Professor in Faculty of Engineering at UCSI University in Malaysia. She obtained her PhD in Electrical Power Engineering from Universiti Putra Malaysia. Her research interests are artificial intelligence, electrical insulation and high voltage engineering.</td>
</tr>
</tbody>
</table>

Dr. Sew Sun Tiang                       | She is an Assistant Professor in Faculty of Engineering at UCSI University in Malaysia. She received her PhD in Electrical and Electronic Engineering from Universiti Sains Malaysia in 2014. Her research interests are optimization and antenna design. |

Dr. Wei Hong Lim                        | He is an Associate Professor in Faculty of Engineering at UCSI University in Malaysia. He received his PhD in Computational Intelligence from Universiti Sains Malaysia in 2014. His research interests are optimization and artificial intelligence. |