Applied Soft Computing Journal 154 (2024) 111328

Contents lists available at ScienceDirect

Applied Soft Computing

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

An effective wavelet neural network approach for solving first and second order ordinary differential equations

Lee Sen Tan [a], Zarita Zainuddin [a], Pauline Ong [b,*], Farah Aini Abdullah [a]

[a] *School of Mathematical Sciences, Universiti Sains Malaysia, 11800 USM, Penang, Malaysia*
[b] *Faculty of Mechanical and Manufacturing Engineering, Universiti Tun Hussein Onn Malaysia, Batu Pahat, 86400 Parit Raja, Johor, Malaysia*

HIGHLIGHTS

- Wavelet neural networks (WNNs) is used to solve ordinary differential equations.
- We propose an improved butterfly optimization algorithm (IBOA) to train the WNNs.
- The accuracy of the testing solutions is improved by the employment of the IBOA.
- The proposed WNNs with IBOA outperformed others.

ARTICLE INFO

Keywords:
Wavelet neural networks
Improved butterfly optimization algorithm
Ordinary differential equations, Initial value problems, Boundary value problems

ABSTRACT

The development of efficient numerical methods for obtaining numerical solutions of first and second order ordinary differential equations (ODEs) is of paramount importance, given the widespread utilization of ODEs as a means of characterizing the behavior in various scientific and engineering disciplines. While various artificial neural networks (ANNs) approaches have recently emerged as potential solutions for approximating ODEs, the limited accuracy of existing models necessitates further advancements. Hence, this study presents a stochastic model utilizing wavelet neural networks (WNNs) to approximate ODEs. Leveraging the compact structure and fast learning speed of WNNs, an improved butterfly optimization algorithm (IBOA) is employed to optimize the adjustable weights, facilitating more effective convergence towards the global optimum. The proposed WNNs approach is then rigorously evaluated by solving first and second order ODEs, including initial value problems, singularly perturbed boundary value problems, and a Lane–Emden type equation. Comparative analyses against alternative training methods, other existing ANNs, and numerical techniques demonstrate the superior performance of the proposed method, affirming its efficiency and accuracy in approximating ODE solutions.

1. Introduction

Ordinary differential equations (ODEs) have widespread applications in engineering problems and epidemic models, often obtained through reduction from partial differential equations (PDEs). Traditionally, classical numerical methods, such as Runge-Kutta and shooting methods, have been employed to approximate ODE solutions for initial value problems (IVPs) and boundary value problems (BVPs), respectively. Alternatively, artificial neural networks (ANNs) offer an appealing alternative, transforming the task of solving ODEs into an optimization problem. ANNs have demonstrated advantages over classical numerical methods, including the ability to provide closed analytic solutions without requiring additional interpolation procedures for testing points within the computational interval. Furthermore, ANNs offer enhanced flexibility by accommodating the solutions of both IVPs and BVPs.

The initial exploration of using ANNs for approximating ODEs involved multilayer perceptrons (MLPs) [1]. Subsequently, several studies utilized MLPs to solve various ODEs, including IVPs and BVPs [2–4]. However, the MLPs suffered from issues such as susceptibility to local minima and slow learning rates [5]. To address these limitations, a performance enhancement was achieved by employing single-layer functional link ANNs, where the hidden layer was replaced with a functional expansion block based on orthogonal polynomials. In a previous work [6], Legendre neural networks (LENNs) were proposed for solving Lane–Emden equations, and the results demonstrated that the

* Corresponding author.
E-mail address: ongp@uthm.edu.my (P. Ong).

https://doi.org/10.1016/j.asoc.2024.111328
Received 10 July 2023; Received in revised form 29 December 2023; Accepted 25 January 2024
Available online 1 February 2024

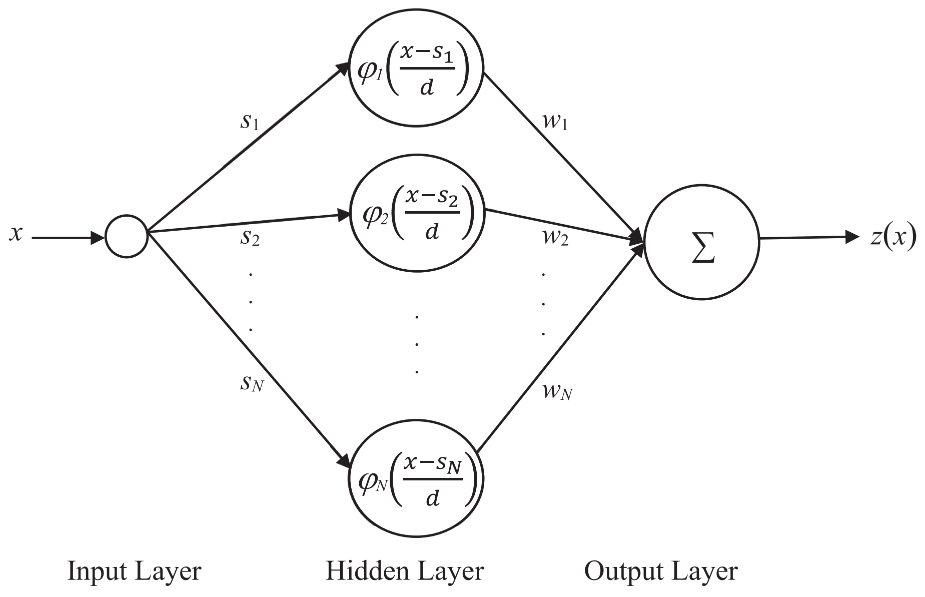

x
s_1
s_2
s_N
$\varphi_1\left(\frac{x-s_1}{d}\right)$
$\varphi_2\left(\frac{x-s_2}{d}\right)$
$\varphi_N\left(\frac{x-s_N}{d}\right)$
w_1
w_2
w_N
Σ
$z(x)$
Input Layer
Hidden Layer
Output Layer

Fig. 1. Architecture of a WNN with input x, output z. This architecture has one input layer with one input node, one hidden layer with N hidden nodes and one output layer. s_n and d are the translation and dilation of the wavelet activation functions φ_n, respectively, for $n = 1, \ldots, N$. The hidden nodes and output node are linked by synaptic weights w_n.

LENNs provided more accurate solutions with reduced computational efforts compared to the MLPs. Nevertheless, the LENNs require a large number of inputs corresponding to Legendre polynomials, resulting in increased computational effort with additional polynomials and inputs [7]. The limitations inherent in the ANNs have spurred the exploration of wavelet neural networks (WNNs), which offer rapid learning capabilities while preserving the universal approximation property exhibited by MLPs [5]. Moreover, WNNs hold promise as a computational model for solving ODEs due to their compact structure resulting from the localized characteristics of wavelet activation functions in the hidden layer [8].

The selection of an appropriate training method has a significant impact on the efficiency and approximation accuracy of ANNs, in addition to the choice of ANN model type. This is primarily because the training method plays a crucial role in determining a high-quality set of optimal weights for ANNs. Traditionally, backpropagation (BP), a gradient-based training algorithm, has demonstrated satisfactory performance in training various ANN models, including MLPs, radial basis function networks (RBFNs), LENNs, and Chebyshev neural networks (CHNNs), for solving ODEs and PDEs [6, 9–13]. The study conducted in [9] demonstrated that the solutions obtained using the CHNNs method for ODEs exhibited a high level of accuracy, matching the exact solutions up to 2–3 decimal places. This outcome highlights the effectiveness of the BP training algorithm in achieving satisfactory solution accuracy. In addition to the BP, the Quasi-Newton Broyden--Fletcher-Goldfarb-Shanno (BFGS) algorithm has shown promising results in training ANNs for solving ODEs, PDEs, fractional differential equations (DEs), and fuzzy DEs [1,14,15]. Local optimization methods, specifically sequential quadratic programming (SQP) and interior point algorithm (IPA), are other gradient-based algorithms utilized for training ANNs models in solving DEs [16–21]. In [17], the SQP algorithm was employed to train ANNs for solving the first Painlevé equations. The achieved mean absolute error (MAE) was in the order of 10^{-07}, indicating that the SQP algorithm has the potential to enhance accuracy in ANNs models. Despite the effectiveness of existing gradient-based training algorithms in optimizing ANN weights, a notable concern is their reliance on gradient information.

To address the limitations of gradient-based training algorithms, the introduction of gradient-free training algorithms has eliminated the need for derivative computations. An example is the extreme learning machine (ELM) algorithm, which has been utilized for determining the weights of Bernstein neural networks (BENNs) and LENNs for solving ODEs and PDEs [22–24]. Additionally, the ELM algorithm has been incorporated into the computation of weights for MLPs and RBFNs in ODE and PDE solutions [25,26]. These studies have focused on adjusting only the synaptic weights. However, the efficacy of the ELM algorithm heavily relies on the predetermined weight values between the input and hidden layers of ANNs, despite its usefulness and efficient computational time. Other gradient-free algorithms employed for adjusting weight parameters of ANNs in the context of solving DEs include singular value decomposition and Nelder-Mead techniques [27–29].

Metaheuristic algorithms, which possess exploration and exploitation capabilities, have also been extensively employed for training ANNs in solving DEs. While metaheuristic algorithms demonstrate strong optimization abilities, recent research has predominantly focused on hybrid algorithms that combine a metaheuristic method with a local optimization algorithm to enhance the performance of ANNs in solving DEs [30–33]. Typically, a metaheuristic algorithm is utilized to obtain improved initial weights, which are subsequently refined by a local optimization algorithm. For instance, in [30], a combination of particle swarm optimization (PSO) and SQP was employed to enhance the efficacy of ANNs in solving two-dimensional nonlinear Bratu's equations. The results indicated that the hybrid PSO-SQP training method outperformed standalone SQP, IPA, PSO, genetic algorithm (GA), as well as the hybrid GA-IPA method. However, it is important to note that the hybridization of two iterative-based training algorithms increased computational time due to a large number of function evaluations.

Recently, an improved version of the butterfly optimization algorithm (IBOA) [34], which is a metaheuristic-based training algorithm, has been proposed and utilized to enhance the training process of WNNs for solving PDEs. Numerical results demonstrated that WNNs trained with IBOA exhibited superior approximation abilities compared to those trained with PSO, butterfly optimization algorithm (BOA), and momentum backpropagation (MBP). By utilizing the IBOA, candidate solutions can effectively escape from local optima and converge towards the search space where near-optimal solutions exist.

Inspired by the effectiveness of the WNNIBOA in approximating PDE solutions [34], this study contributes by extending the utilization of the IBOA to optimize the weight parameters of WNNs for achieving more precise approximations of ODEs. Notably, to the best of the author's

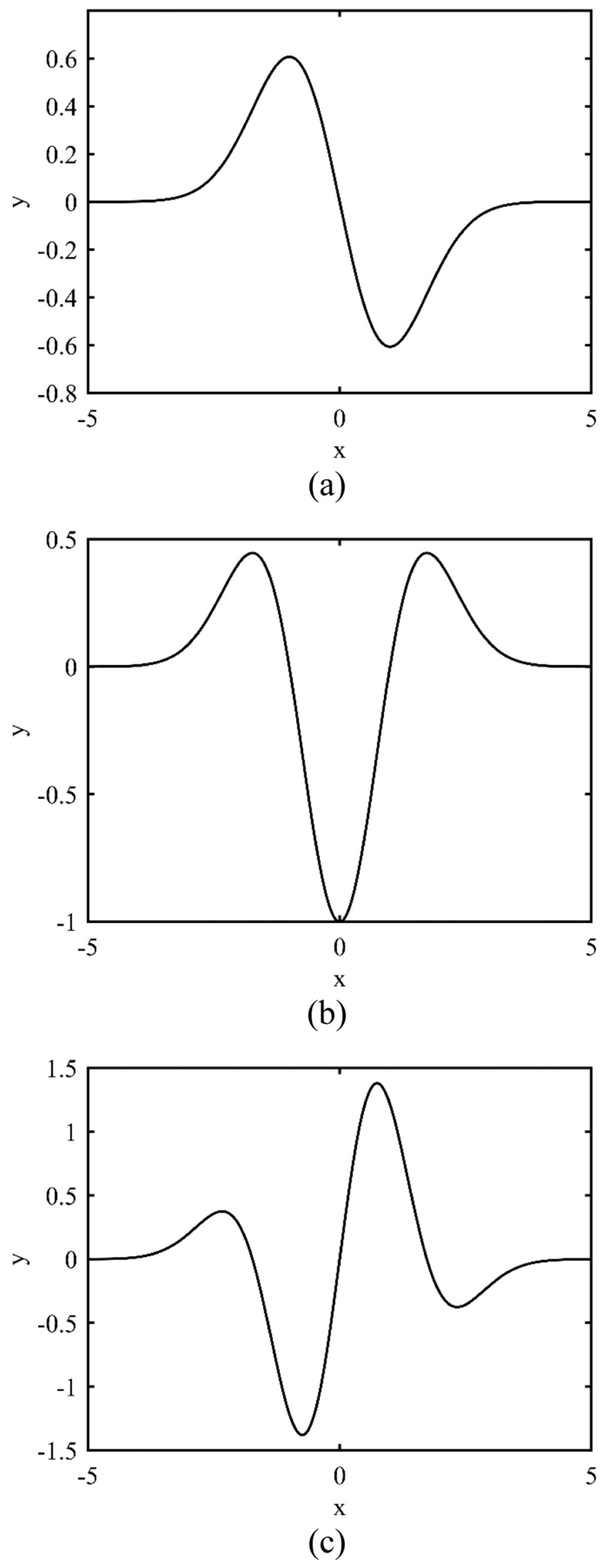


Fig. 2. The illustrations of (a) GW activation function; (b) the first order derivatives of GW activation function; and (c) the second order derivatives of GW activation function.

knowledge, prior research has not explored the applications of WNNs for ODE solutions, making this study a pioneering endeavor. The primary objective of this study is to employ the intelligent computational approach, termed the wavelet neural networks with improved butterfly optimization algorithm (WNNIBOA), for ODE solutions. Furthermore, a comprehensive comparative analysis is performed, evaluating the accuracy of the WNNIBOA against other training methods, alternative ANNs and numerical methods.

2. Wavelet neural networks

WNNs model is a type of feedforward neural network that combines wavelet theory and ANNs [35]. It consists of a three-layer architecture, as depicted in Fig. 1.

In Fig. 1, both the input and output layers of the WNNs model consist of a single node, reflecting the one-dimensional nature of the ODEs being addressed. The input layer receives input values, while the output layer generates the corresponding outputs. The distinctive feature of the WNNs model lies in its hidden layer, which comprises wavelet activation functions φ along with translation parameter s and dilation parameter d. These elements contribute to the WNNs' compact topology compared to other ANNs approaches, resulting in faster training speed [8]. Meanwhile, the output layer of the WNNs operates similarly to that of MLPs, combining the outputs of the hidden nodes through a weighted sum. Overall, the mathematical formulation defining the output of the WNNs can be expressed as:

$$z(x) = \sum_{n=1}^{N} w_n \varphi_n \left(\frac{x - s_n}{d} \right), \tag{1}$$

where x is the input vector, φ_n is the wavelet activation function of hidden nodes, $s_n = [s_1, s_2,., s_N]$ is the translation vector, d is the dilation parameter, $w_n = [w_1, w_2,., w_N]$ is the synaptic weights, and N is the number of hidden nodes.

In this study, the hidden layer of the WNNs for solving ODEs incorporates the Gaussian wavelet (GW) activation function. Specifically, the solution of the ODEs, as well as its first and second order derivatives, are represented in terms of the GW activation function and its corresponding first and second order derivatives. These expressions are provided in Eqs. (2)–(4).

$$\varphi(x) = -xe\left(-\frac{x^2}{2}\right) \tag{2}$$

$$\varphi'(x) = \left(x^2 - 1\right)e\left(-\frac{x^2}{2}\right) \tag{3}$$

$$\varphi''(x) = -x\left(x^2 - 3\right)e\left(-\frac{x^2}{2}\right) \tag{4}$$

Fig. 2 presents the graphical representations of the GW activation function and its corresponding derivatives. In order to align with the shape of the target functions effectively [36], the dilation and translation parameters are maintained as constants. Consequently, the learning process focuses solely on determining the optimal values for the synaptic weights, while the fixed dilation and translation parameters are not subject to adjustment. The appropriate selection of the dilation and translation values for the WNNs in the case of ODE examples is addressed in Section 7.

3. Mathematical formulations for solving ODEs

The problems of finding solutions to ODEs accompanied by initial conditions or boundary conditions are referred to as IVPs and BVPs, respectively. In both the IVPs and BVPs, the solutions of the ODE must not only satisfy the ODE itself but also meet the specified associated conditions. In this study, in order to satisfy the associated conditions of an ODE, the solution is represented as a trial solution, as originally proposed by Lagaris *et al.* [1].

A first order ODE can be formulated as:

$$\frac{dy}{dx} = h\left(x, y\right) \tag{5}$$

The considered ODE represents an IVP with a specified initial condition of $y(a) = A$. In order to address the IVP, the trial solution and its corresponding first-order derivatives can be expressed in the following form:

$$y(x) = A + (x - a)z(x), \tag{6}$$

$$\frac{dy}{dx} = z(x) + \left(x - a\right)\frac{dz}{dx}, \tag{7}$$

where $z(x)$ is the output of the WNNs, and dz/dx is the first order derivative of the output of the WNNs with respect to inputs x.

In general, the trial solution in the proposed approach is designed to satisfy the given conditions without the need for adjustable parameters, except for the final term which incorporates the adjustable output of the WNNs, denoted as $z(x)$. In Eq. (6), the initial condition solution, represented by the term A, ensures that when the initial condition is evaluated at $x = a$, the mathematical expression (x - a) in the final term yields a value of zero, resulting in an exact match between the trial solution $y(x)$ and the given initial condition value A. For other training points within the computational domain interval, the adjustable output of the WNNs is trained using a set of weights that minimize the discrepancy between the trial solution and the first-order ODE, as expressed in Eq. (5). To facilitate the training process, the first-order derivative of the trial solution, necessary for substitution in the first-order derivative term of Eq. (5), is derived and provided in Eq. (7). Thus, for an IVP with initial conditions $y(0) = A$ and $y'(0) = B$, the trial solution can be expressed as:

$$y(x) = A + Bx + x^2 z(x). \tag{8}$$

Eq. (8) differs from Eq. (6) primarily due to the inclusion of an additional term Bx, which ensures that the value B is obtained for $y'(0)$.

Let us consider a BVP associated with a second-order ODE given by:

$$\frac{d^2y}{dx^2} = h\left(x, y, \frac{dy}{dx}\right), \tag{9}$$

with two points Dirichlet boundary conditions, $y(0) = A$ and $y(1) = B$. The trial solution and its corresponding first and second order derivatives for the BVP can be expressed as:

$$y(x) = A(1 - x) + Bx + x(1 - x)z(x), \tag{10}$$

$$\frac{dy}{dx} = -A + B + \left(1 - 2x\right)z(x) + \left(x - x^2\right)\frac{dz}{dx}, \tag{11}$$

$$\frac{d^2y}{dx^2} = -2z(x) + 2\left(1 - 2x\right)\frac{dz}{dx} + \left(x - x^2\right)\frac{d^2z}{dx^2}, \tag{12}$$

where $z(x)$ is the output of the WNNs, and dz/dx and d^2z/dx^2 are the first and second order derivatives of the output of the WNNs with respect to inputs x, respectively.

To obtain the approximated values of the trial function and its derivatives, the training of WNNs is conducted to determine the optimal values of the parameter w for $z(x)$. The obtained $z(x)$ is subsequently differentiated to compute the derivatives of the WNN's output with respect to the inputs x, yielding the following expressions:

$$\frac{dz}{dx} = \sum_{n=1}^{N} w_n \frac{d\varphi}{dx}, \tag{13}$$

$$\frac{d^2z}{dx^2} = \sum_{n=1}^{N} w_n \frac{d^2\varphi}{dx^2}, \tag{14}$$

where $d\varphi/dx$ and $d^2\varphi/dx^2$ are the first and second order derivatives of the wavelet activation functions with respect to inputs x, respectively. After approximating the trial solution and its derivatives, they are incorporated into the fitness function, which is constructed by evaluating the mean square error (MSE) of the residual of the approximated ODE. Specifically, the equations for MSE corresponding to the first and second order ODEs can be represented as Eq. (15) and Eq. (16), respectively, given by:

$$MSE = \frac{1}{NI}\sum_{n=1}^{NI}\left\{\frac{dy_n}{dx} - h\left(x, y_n\right)\right\}^2, \tag{15}$$

$$MSE = \frac{1}{NI}\sum_{n=1}^{NI}\left\{\frac{d^2y_n}{dx^2} - h\left(x, y_n, \frac{dy_n}{dx}\right)\right\}^2, \tag{16}$$

where NI is the total number of training points, y_n is the n-th trial solution, and dy_n/dx and d^2y_n/dx^2 are the first and second order derivatives of the n-th trial solution with respect to inputs x, respectively. In light of this, it becomes evident that employing WNNs approaches for solving ODEs leads to an optimization problem, wherein the synaptic weights are determined such that the error is minimized.

4. Improved butterfly optimization algorithm

IBOA [34], an enhanced variant of BOA, is inspired by the foraging behavior of butterflies. BOA is a population-based metaheuristic algorithm that employs a group of butterflies as search agents [37]. The movement of butterflies can be classified into two scenarios: global search and local search. During the global search, a butterfly is attracted to the fragrance emitted by other butterflies, causing it to move towards the source. On the other hand, during the local search, when a butterfly is unable to perceive any fragrance from its surroundings or other butterflies, it resorts to randomization in its movement.

On this basis, the BOA operates based on the level of fragrance perceived from other butterflies. To represent the fragrance, three key parameters are utilized: stimulus intensity (I), power exponent (a), and sensory modality (c). By employing these parameters, the magnitude of the perceived fragrance (f) is expressed as a function of the physical stimulus intensity, given by the following formulation:

$$f = cI^a. \tag{17}$$

The term stimulus intensity pertains to the strength of the physical stimulus that corresponds to the fitness of a butterfly. The fitness is determined by an objective function, which assesses the degree to which a butterfly fulfills the optimization problem's objective. The power exponent represents the exponent to which the stimulus intensity (the base) is raised. Additionally, the sensory modality serves as a coefficient that multiplies the stimulus intensity raised to the power of a. Consequently, its value impacts the convergence speed of the algorithm.

In addition to the three essential parameters mentioned earlier, the standard BOA utilizes a user-defined switch probability p to alternate between the local search and the global search. However, the BOA prioritizes exploration over exploitation by employing a fixed switch probability, p= 0.8 [37]. This setting favors exploration since when a randomly generated number is lower than the switch probability p, it triggers the exploration process. To address this limitation, an improved version called IBOA has been proposed [34]. The IBOA enhances the exploitation capability of the BOA by introducing a dynamic switch probability, modifying the sensory modality function, and updating the local search equation of the BOA. The performance of the IBOA was evaluated through training WNNs for solving PDEs. Simulation results demonstrated the promising potential of the IBOA in achieving better

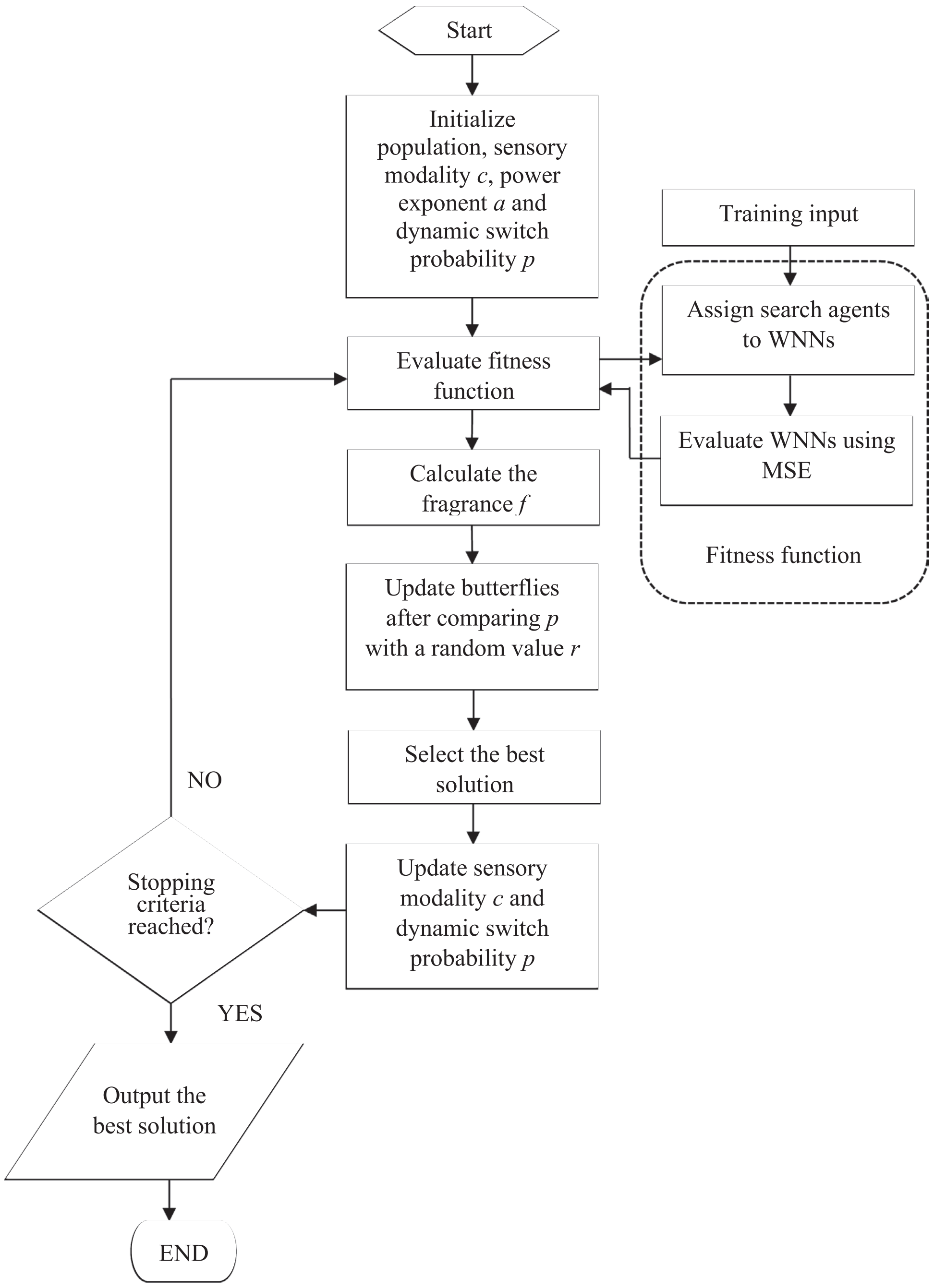

Start
Initialize population, sensory modality c, power exponent a and dynamic switch probability p
Training input
Assign search agents to WNNs
Evaluate fitness function
Evaluate WNNs using MSE
Calculate the fragrance f
Fitness function
Update butterflies after comparing p with a random value r
Select the best solution
NO
Stopping criteria reached?
Update sensory modality c and dynamic switch probability p
YES
Output the best solution
END

Fig. 3. Flowchart of the WNNIBOA.

weight convergence.

The optimization process of the IBOA consists of three stages: initialization stage, iteration stage, and final stage. During the initialization stage, an initial population of butterflies is randomly generated within the specified variable boundary. The fitness and fragrance of each butterfly are then calculated. By comparing the fitness values, the best fitness and its corresponding best solution found so far are stored, allowing the algorithm parameters to be set.

Following the initialization stage, the iteration stage commences by comparing the dynamically determined switch probability, denoted as *p*, with a randomly generated number, *r*, within the range [0,1]. If *p* is greater than *r*, the global search operation is executed, and the solutions undergo updates according to the following procedure [37]:

$$x^{t+1} = x^t + \left(r^2 g - x^t\right) f, r < p, \tag{18}$$

where x^t is the solution vector in iteration number *t* and *g* is the current best solution found among all the butterflies in the current iteration. When the value of the dynamically determined switch probability, *p*, is smaller than or equal to a randomly generated number, *r*, the local search operation is activated. Consequently, the solutions within the IBOA undergo modification and updating using the following procedure [34]:

$$x^{t+1} = x^t + i\left(x_j^t - x_k^t\right) f, r \geq p, \tag{19}$$

where *i* is a random number between 0 and 1, and x_j^t and x_k^t are the *j*-th and *k*-th butterflies, respectively, at iteration *t*, which are selected randomly from the population in the solution space. In contrast to the standard BOA where a random number is multiplied by one of two randomly selected butterflies, the Eq. (19) used in the IBOA introduces randomness by multiplying a random number with the difference between two randomly selected butterflies. This modification helps reduce the chance of solutions clustering near the variable boundaries, as it avoids the accumulation of large step sizes. Additionally, the random number is reduced to the order of one, denoted as *i*, which facilitates the escape from local minima.

After updating the fitness values, the butterflies in the IBOA generate fragrance at their respective positions using Eq. (17). Each new fitness value is compared to its corresponding fitness value. If the new fitness value is smaller, the corresponding fitness value and its solution are replaced with the new fitness value and new solution, respectively. Otherwise, the corresponding fitness value and its solution remain unchanged. Subsequently, the new fitness value of the new solution is compared to the best fitness value found so far. If the best fitness value is larger than the new fitness value, the new fitness value becomes the new best fitness value, and the new solution is recorded as the best solution.

After updating all the butterflies, the sensory modality value of the IBOA is updated as follows:

$$c\left(t+1\right) = c(t) + \frac{0.15c(t)}{MaxI}, \tag{20}$$

where *MaxI* is the maximum number of iterations and *t* is the current iteration number. The adjustment of the sensory modality value in the IBOA differs from the BOA in terms of the coefficient used in the second term. In the IBOA, the coefficient is set to 0.15, which is slightly larger than the value of 0.025 in the standard BOA. Additionally, at the beginning of each iteration, the sensory modality value in the IBOA is initialized to 0.001 instead of 0.01. This lower initialization value aims to reduce the likelihood of butterflies reaching saturation. Meanwhile, the power exponent *a* is fixed and unchanged.

Furthermore, the update of the switch probability value in the IBOA occurs after all butterflies have moved, and it is obtained by subtracting each value of *pp* from unity, as shown in Eq. (21).

$$pp^{t+1} = pp^{t} - e^{\frac{-t}{t+1}} \times \sigma \times pp^{t}, \tag{21}$$

$$p = 1 - pp, \tag{22}$$

Here, the initial value of *pp* is 1 and $\sigma = 0.0002$. The dynamic switch probability *p* starts from 0 at the beginning of the search process. Initially, *p* is small, favoring the execution of local search. This allows potential solutions to be refined while maintaining the algorithm's exploration capability. One iteration is completed during this stage. In subsequent iterations, all butterflies continue to adjust their positions in search of optimal solutions until the stopping criteria are met. In the final stage, the algorithm outputs the optimal solution with the best fitness value. The pseudocode of the IBOA can be found in Algorithm 1.

The IBOA is specifically applied for the training of WNNs. Fig. 3 illustrates the WNNIBOA method's procedure for solving ODEs. Initially, key training parameters, including the number of iterations, stopping criteria, and various IBOA parameters, such as the population size of butterflies, sensory modality (*c*), power exponent (*a*) and dynamic switch probability (*p*), are predefined. Search agents are then assigned to the synaptic weights of WNNs to approximate solutions using a defined set of training points within the ODE domain. The solutions obtained from WNNs are evaluated using either MSE or a fitness function, reflecting the residual of the approximated ODE. This is followed by the iterative phase of IBOA, which involves calculating fragrance, updating butterflies by comparing dynamic switch probability (*p*) and a random value (*r*), evaluating fitness, selecting the optimal solution, and adjusting sensory modality (*c*), and dynamic switch probability (*p*) [34]. This iterative process persists until meeting the stopping criteria, wherein the butterflies undergo continuous training to converge towards synaptic weights that closely satisfy the ODE.

Algorithm 1. Improved Butterfly Optimization Algorithm [34].

```
Objective function $f(\mathbf{x})$, $\mathbf{x} = (x_1, x_2, ..., x_N)$, $N$ = number of dimensions
Generate initial population of $b$ butterflies $\mathbf{x}_m = (m = 1, 2, \ldots, b)$
Stimulus Intensity $I$ at $\mathbf{x}_m$ is determined by $f(\mathbf{x}_m)$
Find the best butterfly in the initial population
Initialize sensory modality $c$, power exponent $a$ and dynamic switch probability $p$
while ($t < MaxI$) or (stopping criterion) do
  for (each butterfly in population) do
    Calculate fragrance for each butterfly using Equation (17)
    if $r < p$ then
      Move towards best butterfly using Equation (18)
    else
      Move randomly using Equation (19)
    end if
    Evaluate the new butterflies
    Update butterflies in population
    Update the best butterfly
  end for
  Update the value of $p$ using Equation (21) and Equation (22)
  Update the value of $c$ using Equation (20)
end while
Output the obtained best solution
```

5. Numerical simulations and discussion

In this section, we analyze the performance of the proposed WNNIBOA method by considering five ODEs: three IVPs and two BVPs. These ODEs have known exact solutions, allowing us to evaluate the effectiveness of the method. All the numerical simulations are conducted using MATLAB.

Table 1
Parameter setting for IBOA, BOA, PSO and MBP.

Parameter	IBOA	BOA	PSO	PSOA	MBP	DEV
Population size	30	30	30	250	-	50
Power exponent	0.1	0.1	-	-	-	-
Initial sensory modality	0.001	0.01	-	-	-	-
Switch probability	dynamic	0.8	-	-	-	-
Momentum	-	-	-	-	0.9	-

5.1. Experimental setup

For each ODE considered in this study, the determination of the number of hidden nodes in the WNNIBOA model is based on the number of training points employed in each example. To evaluate the performance of the WNNIBOA model, five alternative WNNs models are developed, each trained with a different optimization algorithm using the same stopping criteria. Particularly, the optimization algorithms of BOA, PSO, MBP and differential evolution (DEV) are considered in this study. These models are referred to as wavelet neural networks with butterfly optimization algorithm (WNNBOA), wavelet neural networks with particle swarm optimization (WNNPSO and WNNPSOA), wavelet neural networks with momentum backpropagation (WNNMBP), and wavelet neural networks with differential evolution (WNNDEV). The training process stops upon meeting either of two criteria: achieving a fitness limit below The training process is terminated based on two stopping criteria: a fitness limit of less than 1.00e-16 and a maximum of 5000 iterations. All training methods adhere to these criteria except for WNNPSOA, which employs a maximum of 600 iterations, and WNNDEV, limited to 300 iterations. It is worth noting that determining optimal stopping criteria for a stochastic algorithm to converge is challenging due to randomness. Consequently, the choice of stopping criteria relies on empirical evaluations. To ensure robustness, the simulations are repeated 10 times using different sets of synaptic weights.

The variable boundary in both the BOA and IBOA is initially defined as $[-50, 50]$. However, this range of numbers can lead to inefficiency in the PSO algorithm. To address this issue, the variable boundary in the PSO algorithm is adjusted to the range of $[-5, 5]$ in order to optimize the solutions of ODEs effectively. In the PSO algorithm, the inertia weight, personal, and global learning coefficients take values of 1, 1.5, and 2.0, respectively [38]. WNNPSO and WNNPSOA vary not only in their maximum iteration settings but also in population size. Specifically, the WNNPSO is trained with a population size of 30, while the WNNPSOA utilizes a population size of 250. For the MBP algorithm, the learning rate is empirically adjusted to the highest stable value without significant divergence for each example. Specifically, values of 0.1, 0.5, 0.006, 0.002, and 0.3 are set for Examples 1, 2, 3, 4, and 5, respectively. Although not guaranteed as optimal, these values efficiently enhance speed with momentum value assigned to 0.9. The DEV parameters are set as: crossover rate of 0.9, mutation factor of 1, and population size of 50. Table 1 provides an overview of the parameter settings utilized in the IBOA, BOA, PSO, MBP, and DEV algorithms.

In addition to comparing the performance of the WNNs trained using different methods, the WNNIBOA is also compared with other existing ANNs and numerical methods reported in the literature. To facilitate comparison with previous studies, the same training and testing points are used if they are available in the literature. However, if the points are not provided, testing points are randomly selected within the computational interval.

5.2. Performance evaluation

The performance evaluation of all the WNNs models is conducted using two error criteria: the absolute error (AE) and MAE. These criteria are defined as follows:

$$AE = |\hat{y}_n - y_n|, \tag{23}$$

$$MAE = \frac{1}{NI}\sum_{n=1}^{NI}|\hat{y}_n - y_n|, \tag{24}$$

where $\hat{y}_n$ is the n-th exact solution, y_n is the n-th WNNs solution and NI is the total number of training points. The AE values are plotted and tabulated to provide an indication of the accuracy of the approximate solution. In an ideal scenario, the AE values should approach zero and be close to the x-axis in the error graphs, indicating a close approximation to the exact solution. Additionally, the MAE computes the average of all the AE values, providing a measure of the overall performance of the algorithm in each run.

In each independent run, the MAE was computed and recorded for statistical analysis. Specifically, for each training algorithm, we obtained ten sets of best weight vectors from ten independent runs. Among these ten sets, we selected the set with the minimum MAE (referred to as MIN MAE) for performance assessment.

5.3. Numerical experiments

Example 1. Consider the first order linear ODE that has been previously solved by Haweel and Abdelhameed [39] and Li-ying *et al.* [40]. The ODE is defined as:

$$\frac{dy}{dx} = y - x^2 + 1,\ x \in \left[0, 2\right] \tag{25}$$

subject to the initial condition $y(0) = 0.5$. The exact solution to this ODE is given by $\hat{y} = (x+1)^2 - 0.5e^x$, while the trial solution is written as $y(x) = 0.5 + xz(x)$. The domain interval [0,2] was divided into ten equal parts for numerical evaluation. The AE between the WNNs and the exact solution for each training method are presented in Table 2 and depicted in Fig. 4.

Fig. 4 exhibits a close proximity to the x-axis, indicating a negligibly small AE. The comparison of AE values in Table 2 further demonstrates the higher accuracy achieved by integrating the IBOA with the WNNs.

Table 2
Comparison of AE values at training points for Example 1.

x	AE								
	WNNIBOA	WNNBOA	WNNPSO	WNNPSOA	WNNMBP	WNNDEV	PSNNs[39]	CNNs[39]	Heun[39]
0.0000	0	0	0	0	0	0	0	0	0
0.2000	3.72e-09	1.54e-01	5.05e-05	1.64e-04	4.73e-04	1.01e-02	2.99e-04	6.01e-04	2.00e-03
0.4000	2.29e-09	2.94e-01	9.97e-05	2.15e-04	7.12e-04	3.50e-03	4.88e-04	3.88e-04	4.19e-03
0.6000	6.12e-09	4.22e-01	6.67e-06	9.45e-05	4.75e-04	7.78e-03	6.41e-04	2.34e-03	6.84e-03
0.8000	9.51e-09	5.28e-01	1.88e-04	2.53e-04	4.12e-04	1.36e-02	7.30e-04	1.53e-03	9.63e-03
1.0000	9.65e-09	6.06e-01	6.65e-05	6.29e-05	7.53e-04	8.46e-03	8.59e-04	1.74e-03	1.29e-02
1.2000	1.42e-08	6.68e-01	1.54e-04	4.41e-04	1.18e-03	8.34e-04	1.14e-03	4.44e-03	1.64e-02
1.4000	1.23e-08	7.37e-01	5.09e-05	2.93e-04	1.28e-03	5.92e-03	1.30e-03	3.50e-03	2.04e-02
1.6000	3.07e-08	8.32e-01	2.23e-04	1.78e-04	1.17e-03	1.06e-02	1.48e-03	5.48e-03	2.47e-02
1.8000	1.52e-08	9.65e-01	1.41e-04	1.18e-06	1.57e-03	1.20e-02	1.76e-04	4.28e-03	2.94e-02
2.0000	3.51e-07	1.14e+ 00	3.48e-05	6.41e-04	2.47e-03	1.51e-02	7.03e-03	1.90e-02	3.42e-02

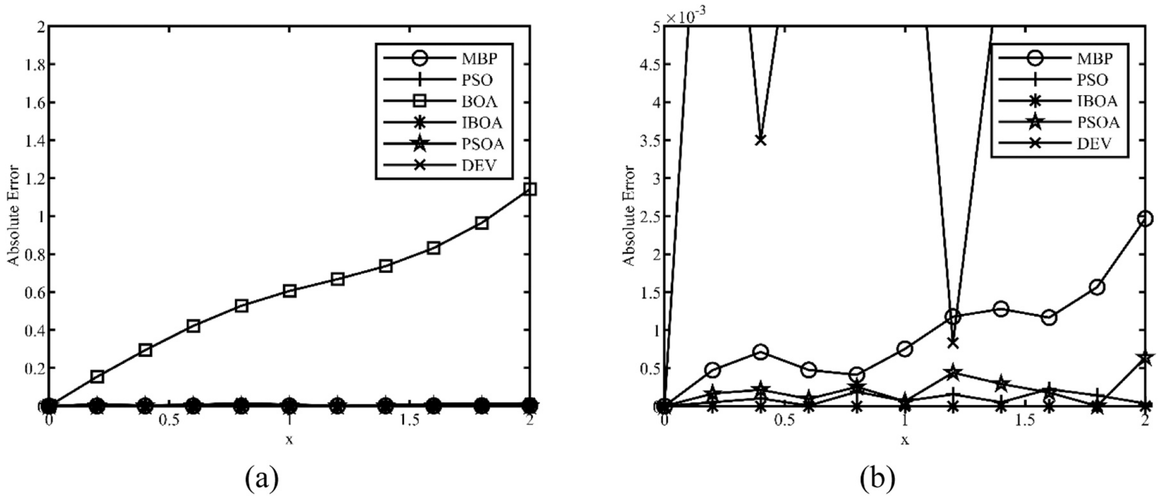


Fig. 4. (a) Plot of AE for WNNs solutions with different training methods (b) Zoom-in of part (a) for Example 1.

Table 3
Comparison of AE values at testing points for Example 1.

x	AE					
	WNNIBOA	WNNBOA	WNNPSO	WNNPSOA	WNNMBP	WNNDEV
0.0111	5.16e-11	9.36e-03	3.76e-06	2.32e-06	8.49e-06	1.03e-03
0.6194	6.54e-09	4.34e-01	2.93e-05	1.26e-04	4.53e-04	8.75e-03
1.1992	1.42e-08	6.68e-01	1.54e-04	4.40e-04	1.18e-03	8.03e-04
1.6100	3.09e-08	8.38e-01	2.30e-04	1.93e-04	1.17e-03	1.09e-02
1.8132	1.42e-08	9.75e-01	1.22e-04	4.91e-05	1.62e-03	1.13e-02

Table 4
Comparison of AE values at training points for Example 2.

x	AE						
	WNNIBOA	WNNBOA	WNNPSO	WNNPSOA	WNNMBP	WNNDEV	ANNs[31]
0	0	0	0	0	0	0	2.10e-10
0.0500	2.91e-10	5.17e-03	3.52e-05	4.54e-05	6.44e-05	3.53e-03	3.39e-08
0.1000	4.46e-10	5.54e-03	2.58e-05	4.71e-05	6.75e-05	4.25e-03	7.82e-08
0.1500	1.18e-10	3.79e-03	1.64e-05	4.84e-06	6.27e-06	3.13e-03	3.37e-09
0.2000	5.67e-11	1.59e-03	5.09e-05	4.52e-05	6.67e-05	2.31e-03	2.27e-09
0.2500	1.09e-10	1.80e-04	5.28e-05	7.11e-05	1.04e-04	2.83e-03	1.50e-08
0.3000	3.97e-10	1.24e-03	2.37e-05	6.06e-05	8.73e-05	4.31e-03	6.33e-08
0.3500	6.38e-10	1.69e-03	1.71e-05	2.14e-05	2.88e-05	5.72e-03	1.81e-07
0.4000	7.83e-10	1.88e-03	4.64e-05	2.76e-05	4.15e-05	6.27e-03	7.22e-09
0.4500	7.94e-10	2.20e-03	5.04e-05	6.59e-05	9.15e-05	5.82e-03	9.35e-10
0.5000	6.39e-10	3.02e-03	2.94e-05	7.99e-05	1.01e-04	4.82e-03	5.18e-09
0.5500	3.98e-10	4.58e-03	3.58e-06	6.69e-05	6.65e-05	3.81e-03	6.69e-08
0.6000	2.50e-10	6.89e-03	3.10e-05	3.49e-05	5.50e-06	3.08e-03	7.42e-09
0.6500	2.96e-10	9.80e-03	3.97e-05	2.24e-06	5.55e-05	2.50e-03	7.80e-09
0.7000	3.80e-10	1.29e-02	2.75e-05	2.99e-05	8.96e-05	1.84e-03	7.75e-08
0.7500	2.19e-10	1.57e-02	4.20e-06	3.92e-05	8.16e-05	1.09e-03	2.58e-09
0.8000	2.00e-10	1.74e-02	1.45e-05	2.99e-05	3.57e-05	5.71e-04	1.95e-08
0.8500	5.01e-10	1.73e-02	1.69e-05	1.07e-05	2.32e-05	7.08e-04	1.28e-08
0.9000	4.76e-10	1.48e-02	4.89e-06	5.38e-06	5.91e-05	1.36e-03	4.92e-08
0.9500	4.41e-10	9.21e-03	5.31e-06	8.38e-06	4.60e-05	1.45e-03	1.78e-08
1.0000	0	0	0	0	0	0	2.30e-10

The AE range for the WNNIBOA model is found to be between 10^{-07} and 10^{-09}, significantly smaller compared to the ranges of 10^{+00} to 10^{-01}, 10^{-03} to 10^{-04}, and 10^{-02} to 10^{-04} obtained by the WNNBOA, WNNMBP, and WNNDEV models, respectively. Examining the performance of PSO and PSOA in Table 2 reveals that both algorithms achieve an AE range between 10^{-04} and 10^{-06}. However, these results are surpassed by WNNIBOA. Hence, it suggests that PSO exhibits consistent behavior while optimizing WNNs for ODEs, despite variations in maximum iteration numbers and population sizes. Furthermore, the adoption of the IBOA training method leads to a remarkable improvement in precision at the endpoint, reducing the value from 1.14e+ 00 (BOA) to 3.51e-07. These findings highlight the substantial impact of the training method selection on the approximation ability of the WNNs.

In addition to comparing with other WNNs solutions, the WNNIBOA solutions were further compared with those obtained from a numerical method and two ANNs methods, namely Heun's method, power series neural networks (PSNNs) and cosine neural networks (CNNs) [39]. PSNNs and CNNs are ANNs that utilize power series and cosine function as activation functions in the hidden nodes, respectively, while Heun's

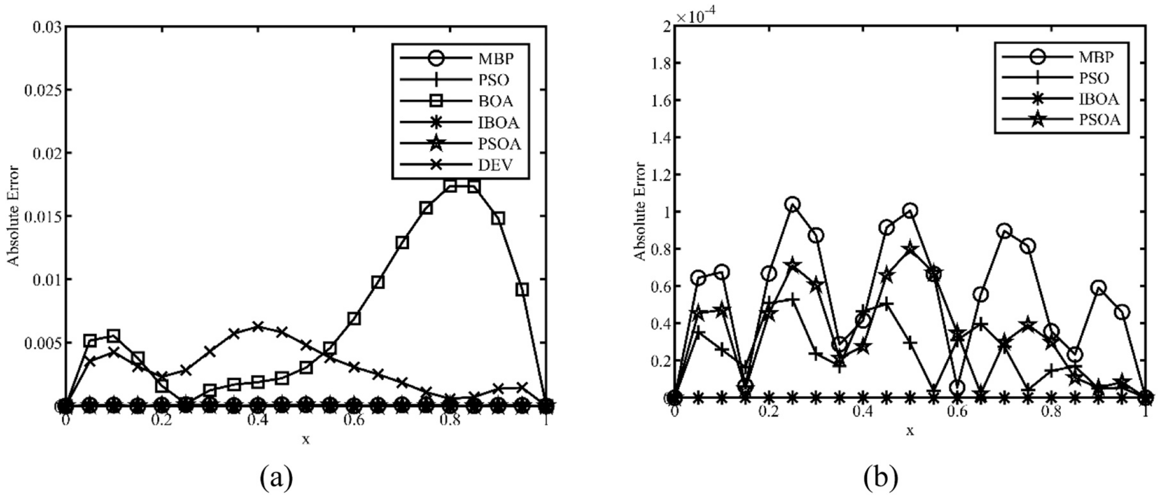


Fig. 5. (a) Plot of AE for WNNs solutions with different training methods (b) Zoom-in of part(a) for Example 2.

Table 5
Comparison of AE values at testing points for Example 2.

x	AE					
	WNNIBOA	WNNBOA	WNNPSO	WNNPSOA	WNNMBP	WNNDEV
0.0910	4.69e-10	5.69e-03	3.10e-05	5.08e-05	7.29e-05	4.32e-03
0.1635	3.55e-11	3.19e-03	2.80e-05	9.52e-06	1.47e-05	2.80e-03
0.3943	7.73e-10	1.86e-03	4.42e-05	2.22e-05	3.40e-05	6.26e-03
0.6776	3.57e-10	1.15e-02	3.52e-05	1.94e-05	7.91e-05	2.16e-03
0.8895	4.95e-10	1.56e-02	7.95e-06	2.87e-06	5.51e-05	1.22e-03

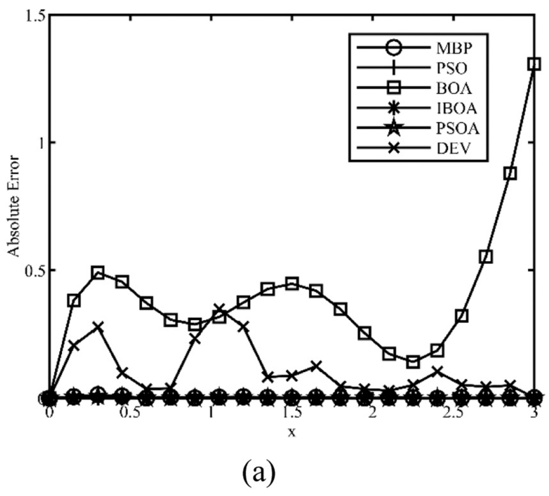
method is a numerical method for solving ODEs with initial conditions. Table 2 presents the AE values for the PSNNs, CNNs, and Heun's method, which range from 10^{-03} to 10^{-04}, 10^{-02} to 10^{-04}, and 10^{-02} to 10^{-03}, respectively. When compared to the Heun's method, only a slight reduction in AE values was observed for the PSNNs and CNNs. However, the proposed WNNIBOA method exhibited the most significant improvement in accuracy when solving the ODE.

Furthermore, compared to the RBFNs method, which utilizes a network architecture with one hidden layer consisting of 20 hidden nodes and 30 equidistant training points [26], the proposed WNNIBOA method achieved higher accuracy solutions with fewer hidden nodes and training points. This is evident from the fact that the maximum AE obtained by the RBFNs method (2.20e-06) was larger than the AE obtained by the proposed WNNIBOA method (3.51e-07).

All WNNs models were subjected to testing using five testing points. Table 3 presents the AE values at these testing points for all the WNNs models. It is evident that the testing solutions obtained by the WNNIBOA exhibited much closer proximity to the exact solution compared to the other WNNs models. Interestingly, it was observed that the testing point at 0.0111, which lies in close proximity to the initial point, consistently yielded the lowest AE value among all the testing solutions across the WNNs models. This finding suggests that the automatic satisfaction of

Table 6
Comparison of AE values at training points for Example 3.

x	AE						
	WNNIBOA	WNNBOA	WNNPSO	WNNPSOA	WNNMBP	WNNDEV	RBFNs[11]
0.0000	0	0	0	0	0	0	0
0.1500	1.94e-07	3.83e-01	4.26e-03	3.96e-03	6.06e-03	2.06e-01	3.35e-04
0.3000	1.14e-07	4.91e-01	8.61e-03	8.60e-03	1.37e-02	2.78e-01	3.97e-04
0.4500	9.81e-08	4.55e-01	4.31e-03	4.78e-03	8.48e-03	9.93e-02	1.01e-04
0.6000	6.96e-08	3.72e-01	1.37e-03	7.14e-04	7.51e-04	3.56e-02	4.16e-04
0.7500	4.66e-08	3.07e-01	2.33e-03	2.00e-03	4.37e-03	3.90e-02	1.10e-03
0.9000	3.22e-08	2.89e-01	8.68e-04	4.93e-04	1.09e-03	2.33e-01	1.39e-03
1.0500	2.35e-08	3.18e-01	3.74e-03	2.90e-03	4.05e-03	3.49e-01	9.82e-04
1.2000	2.71e-08	3.74e-01	3.21e-03	2.65e-03	5.86e-03	2.80e-01	5.99e-04
1.3500	3.27e-08	4.27e-01	1.09e-04	4.75e-04	3.18e-03	8.31e-02	8.12e-04
1.5000	3.05e-08	4.48e-01	2.34e-03	1.22e-03	1.17e-03	8.79e-02	1.12e-03
1.6500	2.46e-08	4.20e-01	1.94e-03	1.06e-03	3.50e-03	1.25e-01	1.09e-03
1.8000	1.66e-08	3.48e-01	4.95e-04	2.19e-04	2.37e-03	4.67e-02	6.07e-04
1.9500	5.45e-09	2.54e-01	2.22e-03	1.01e-03	7.16e-04	3.59e-02	1.06e-04
2.1000	4.24e-10	1.74e-01	1.46e-03	6.39e-04	2.95e-03	2.95e-02	2.78e-04
2.2500	1.37e-09	1.42e-01	7.81e-04	1.47e-04	2.55e-03	5.14e-02	8.27e-04
2.4000	1.17e-08	1.87e-01	1.78e-03	3.94e-04	2.58e-05	1.04e-01	1.23e-03
2.5500	1.30e-08	3.23e-01	2.82e-04	1.18e-04	2.30e-03	5.27e-02	1.48e-03
2.7000	2.57e-09	5.54e-01	1.53e-03	8.06e-06	2.05e-03	4.40e-02	1.78e-03
2.8500	2.61e-08	8.80e-01	4.11e-04	3.69e-05	9.74e-04	5.01e-02	2.49e-03
3.0000	9.59e-08	1.31e+ 00	1.45e-03	1.24e-04	2.23e-03	2.43e-03	2.86e-03



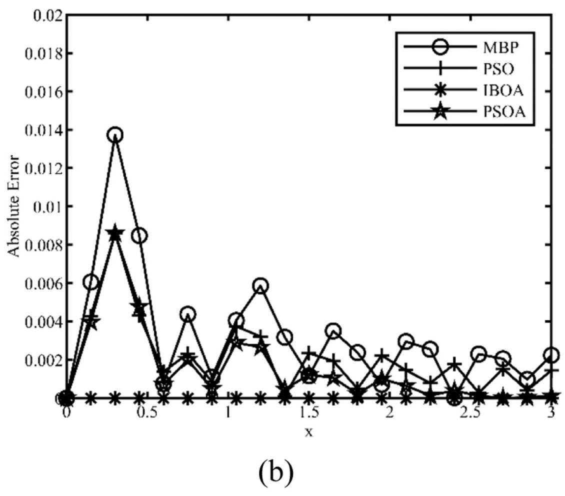


Fig. 6. (a) Plot of AE for WNNs solutions with different training methods (b) Zoom-in of part (a) for Example 3.

Table 7
Comparison of AE values at testing points for Example 3.

x	AE					
	WNNIBOA	WNNBOA	WNNPSO	WNNPSOA	WNNMBP	WNNDEV
0.0500	9.40e-08	1.67e-01	5.34e-04	7.34e-04	1.00e-03	3.50e-02
0.5683	7.62e-08	3.90e-01	4.59e-04	2.01e-04	9.47e-04	2.32e-02
1.1243	2.40e-08	3.44e-01	3.95e-03	3.13e-03	5.55e-03	3.38e-01
1.7566	1.93e-08	3.73e-01	2.67e-04	1.67e-04	3.02e-03	7.55e-02
2.6590	1.10e-09	4.81e-01	1.23e-03	1.06e-05	2.45e-03	2.12e-02

the initial condition through the trial solution confers a notable advantage in terms of accuracy for points located near the initial point.

Example 2. We consider the linear and homogeneous singularly perturbed BVP of the second order ODE as presented by Raja *et al.* [31], given as:

$$-0.1\frac{d^2y}{dx^2}+y=0, x\in\left[0,1\right] \tag{26}$$

The ODE includes boundary conditions $y(0)=1$ and $y(1)=0$. To fulfil the boundary conditions, the trial solution is expressed as $y(x)=1-x+(x-x^2)z(x)$. The exact solution is $\hat{y}=\frac{e^{(1-x)/\sqrt{0.1}}-e^{-(1-x)/\sqrt{0.1}}}{e^{1/\sqrt{0.1}}-e^{-1/\sqrt{0.1}}}$. The computational domain [0,1] was divided into twenty equal segments. The AE values for the WNNs trained with various methods are provided in Table 4 and depicted in Fig. 5.

As indicated in Table 4, the AE values for the WNNMBP, WNNs trained with PSO variants, WNNDEV and WNNBOA models range from 10^{-04} to 10^{-06}, 10^{-05} to 10^{-06}, 10^{-03} to 10^{-04} and 10^{-02} to 10^{-04},

Table 8
Comparison of AE values at training points for Example 4.

x	AE						
	WNNIBOA	WNNBOA	WNNPSO	WNNPSOA	WNNMBP	WNNDEV	ANNs[42]
0	0	0	0	0	0	0	1.80e-07
0.0500	2.76e-12	1.15e-03	1.67e-06	1.83e-06	1.96e-06	4.05e-04	1.70e-08
0.1000	1.87e-11	3.87e-03	9.74e-06	9.96e-06	7.02e-07	1.32e-03	1.10e-06
0.1500	2.17e-11	7.24e-03	1.77e-05	1.73e-05	1.08e-05	2.03e-03	7.80e-07
0.2000	3.14e-12	1.06e-02	1.63e-05	1.40e-05	3.30e-05	2.27e-03	3.20e-07
0.2500	8.60e-12	1.34e-02	2.20e-06	3.72e-06	6.07e-05	2.22e-03	2.90e-07
0.3000	5.96e-12	1.54e-02	2.07e-05	3.18e-05	8.48e-05	2.06e-03	1.90e-07
0.3500	3.21e-11	1.65e-02	4.32e-05	6.00e-05	9.57e-05	1.84e-03	4.40e-07
0.4000	4.08e-11	1.68e-02	5.53e-05	7.63e-05	8.67e-05	1.49e-03	3.30e-07
0.4500	1.97e-11	1.62e-02	5.03e-05	7.13e-05	5.65e-05	1.00e-03	6.60e-07
0.5000	1.74e-11	1.49e-02	2.76e-05	4.19e-05	1.03e-05	5.58e-04	5.90e-07
0.5500	4.72e-11	1.32e-02	6.89e-06	7.12e-06	4.10e-05	4.31e-04	3.80e-07
0.6000	5.62e-11	1.11e-02	4.19e-05	6.38e-05	8.31e-05	7.98e-04	4.40e-07
0.6500	4.48e-11	8.69e-03	6.49e-05	1.12e-04	1.02e-04	1.57e-03	9.80e-08
0.7000	1.86e-11	6.16e-03	6.65e-05	1.38e-04	9.07e-05	2.35e-03	1.90e-07
0.7500	1.46e-11	3.54e-03	4.45e-05	1.32e-04	4.91e-05	2.69e-03	4.50e-07
0.8000	3.82e-11	9.34e-04	6.48e-06	9.61e-05	9.52e-06	2.39e-03	9.10e-08
0.8500	3.10e-11	1.50e-03	3.10e-05	4.55e-05	6.17e-05	1.73e-03	-
0.9000	2.68e-12	3.51e-03	4.85e-05	3.21e-06	8.12e-05	1.35e-03	1.80e-07
0.9500	1.86e-11	4.63e-03	3.43e-05	9.77e-06	5.38e-05	1.61e-03	-
1.0000	5.56e-12	4.16e-03	2.15e-06	5.38e-06	1.75e-06	2.08e-03	2.70e-07

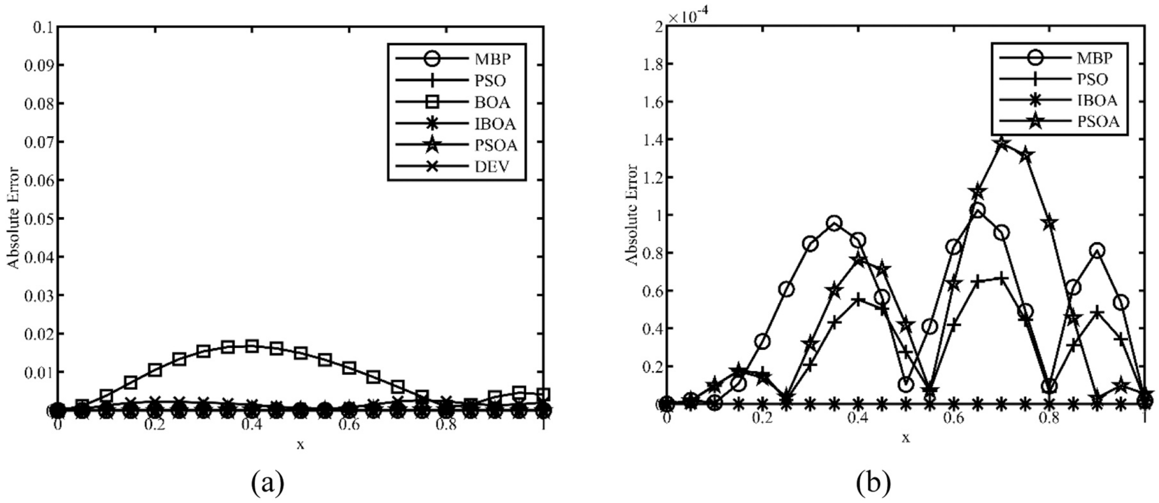


Fig. 7. (a) Plot of AE for WNNs solutions with different training methods (b) Zoom-in of part (a) for Example 4.

Table 9
Comparison of AE values at testing points for Example 4.

x	AE					
	WNNIBOA	WNNBOA	WNNPSO	WNNPSOA	WNNMBP	WNNDEV
0.0200	2.00e-13	2.03e-04	2.79e-08	6.48e-08	5.06e-07	6.13e-05
0.2455	8.57e-12	1.32e-02	3.93e-06	1.59e-06	5.82e-05	2.23e-03
0.4352	2.86e-11	1.64e-02	5.37e-05	7.53e-05	6.75e-05	1.15e-03
0.6888	2.54e-11	6.74e-03	6.82e-05	1.35e-04	9.62e-05	2.20e-03
0.8995	2.35e-12	3.49e-03	4.85e-05	3.51e-06	8.13e-05	1.35e-03

respectively. In contrast, the WNNIBOA model achieves significantly lower AE values in the range of 10^{-10} to 10^{-11}, demonstrating its superior performance compared to the other WNNs models.

Additionally, the proposed WNNIBOA model's accuracy was compared with an existing ANNs method proposed by Raja *et al.* [31], where optimal weights were trained using a two-stage hybrid training approach combining GA with SQP. The MIN, mean, and standard deviation (STD) values of the AE obtained from a sufficiently large number of independent runs were reported in [31]. For comparison purposes, the MIN AE values representing the best solutions were selected and denoted as ANNs in Table 4. These values were compared with the AE values obtained by the WNNIBOA model from the run with the MIN MAE. As shown in Table 4, the ANNs method by Raja *et al.* achieved a range of MIN AE between 10^{-07} and 10^{-10}, while the proposed WNNIBOA model outperforms it with significantly lower AE values. This comparison confirms the effectiveness of the WNNIBOA model in solving ODEs. Notably, unlike the ANNs' hybrid training approach, the WNNs' IBOA training method is relatively straightforward as it does not require an additional combination of an iterative-based local search method.

Table 5 presents the AE values at five testing points. A comparison with other WNNs models reveals that the WNNIBOA model consistently exhibits the most substantial improvement in accuracy across all testing points. Notably, the AE values achieved by the WNNIBOA model at the testing points are within the same range as those obtained at the training points.

Example 3. We consider the first order linear ODE.

Table 10
Comparison of AE values at training points for Example 5.

x	AE					
	WNNIBOA	WNNBOA	WNNPSO	WNNPSOA	WNNMBP	WNNDEV
0	0	0	0	0	0	0
0.0667	3.98e-09	1.27e-03	6.53e-05	1.58e-04	1.07e-04	1.39e-03
0.1333	7.42e-09	2.65e-03	3.65e-05	1.17e-04	4.55e-05	2.74e-03
0.2000	3.31e-09	5.62e-03	4.39e-05	7.98e-05	9.34e-05	3.82e-03
0.2667	3.77e-09	5.01e-03	9.09e-05	2.34e-04	1.51e-04	3.67e-03
0.3333	9.82e-09	8.05e-04	6.96e-05	2.19e-04	8.01e-05	2.43e-03
0.4000	1.39e-08	5.42e-03	2.85e-06	4.99e-05	5.16e-05	1.58e-03
0.4667	1.32e-08	1.15e-02	5.97e-05	1.52e-04	1.39e-04	2.38e-03
0.5333	8.55e-09	1.55e-02	7.77e-05	2.55e-04	1.20e-04	4.53e-03
0.6000	5.82e-09	1.63e-02	4.32e-05	1.98e-04	1.39e-05	6.54e-03
0.6667	5.67e-09	1.40e-02	1.63e-05	2.99e-05	9.90e-05	7.22e-03
0.7333	3.10e-09	9.60e-03	5.65e-05	1.26e-04	1.32e-04	6.80e-03
0.8000	2.59e-09	4.86e-03	4.66e-05	1.59e-04	6.05e-05	6.29e-03
0.8667	9.60e-09	1.44e-03	1.26e-06	6.36e-05	4.95e-05	5.77e-03
0.9333	4.49e-09	1.30e-04	3.06e-05	3.20e-05	8.13e-05	3.88e-03
1.0000	0	0	0	0	0	0

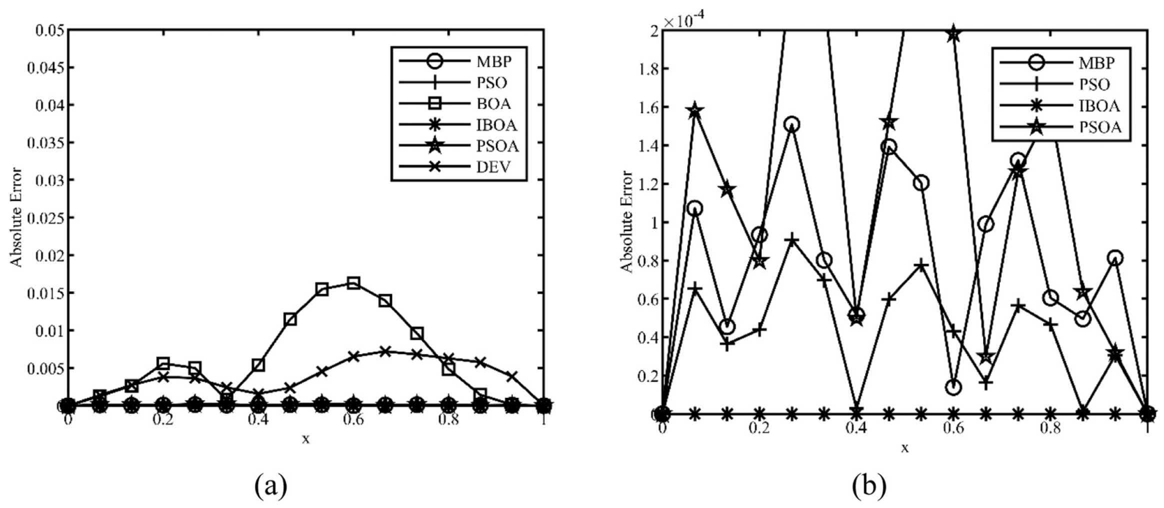


Fig. 8. (a) Plot of AE for WNNs solutions with different training methods (b) Zoom-in of part (a) for Example 5.

Table 11
Comparison of AE values at testing points for Example 5.

x	AE					
	WNNIBOA	WNNBOA	WNNPSO	WNNPSOA	WNNMBP	WNNDEV
0.1318	7.43e-09	2.55e-03	3.81e-05	1.21e-04	4.85e-05	2.71e-03
0.2555	2.62e-09	5.38e-03	8.76e-05	2.18e-04	1.51e-04	3.80e-03
0.4721	1.29e-08	1.19e-02	6.32e-05	1.66e-04	1.42e-04	2.52e-03
0.6384	5.78e-09	1.53e-02	9.18e-06	1.08e-04	5.66e-05	7.11e-03
0.9516	4.46e-10	6.91e-05	2.76e-05	3.68e-05	6.67e-05	2.96e-03

Table 12
Percentage of convergence rate in terms of the MAE for WNNs models.

% Runs with MAE	Training Algorithm	Percentage of convergence rate				
		Example 1	Example 2	Example 3	Example 4	Example 5
10e-03	IBOA	100	100	100	100	100
	BOA	0	40	0	10	40
	PSO	100	100	90	100	100
	PSOA	100	100	90	100	100
	MBP	100	100	100	100	100
	DEV	10	100	0	50	20
10e-05	IBOA	100	100	100	100	100
	BOA	0	0	0	0	0
	PSO	10	80	0	20	60
	PSOA	0	50	0	50	0
	MBP	0	100	0	100	100
	DEV	0	0	0	0	0
10e-07	IBOA	100	100	100	100	100
	BOA	0	0	0	0	0
	PSO	0	0	0	0	0
	PSOA	0	0	0	0	0
	MBP	0	0	0	0	0
	DEV	0	0	0	0	0

$$\frac{dy}{dx}+2y=\cos\left(4x\right),\ x\in\left[0,3\right] \tag{27}$$

with the initial condition $y(0) = 3$. The exact solution for this ODE is $\hat{y} = 0.1(29e^{-2x} + 2\sin(4x) + \cos(4x))$. To satisfy the initial condition, the trial solution is written as $y(x) = 3 + xz(x)$. The computational domain [0,3] was divided into twenty equal parts with a step size of 0.15. Table 6 presents the AE values for the WNNs models trained with different methods. To provide a visual comparison, these AE values are plotted in Fig. 6.

Since the AE values that are far away from zero indicate a low accuracy, it can be inferred from Fig. 6 that the most unfavourable performance is obtained by the WNNBOA and followed by the WNNDEV. Analyzing Table 6 reveals satisfactory results for the WNNMBP, WNNPSO and WNNPSOA, with AE values ranging from 10^{-02} to 10^{-05}, 10^{-03} to 10^{-04} and 10^{-03} to 10^{-06}, respectively. In contrast, the WNNIBOA demonstrates the highest precision, exhibiting an impressive range of AE values between 10^{-07} and 10^{-10}, accompanied by a smooth graph in Fig. 6. Notably, the utilization of the WNNIBOA method leads to a remarkable improvement in accuracy at the endpoint, reducing the AE from 1.31e+ 00 to 9.59e-08 compared to the BOA method. This observation further reinforces the efficacy of the IBOA in achieving optimal weight solutions.

Additionally, the performance of the proposed WNNIBOA model was

Table 13
Statistical MAE results of different WNNs models.

Example	Performance	WNN IBOA	WNN BOA	WNN PSO	WNN PSOA	WNN MBP	WNN DEV	*p*-value
1	MIN	**4.13e-08**	5.77e-01	9.23e-05	2.13e-04	9.53e-04	7.98e-03	3.84e-28
	MEAN	**1.79e-07**	1.50e+00	1.76e-03	2.56e-03	1.11e-03	7.74e-02	
	MAX	**2.80e-07**	2.06e+00	7.07e-03	6.00e-03	1.23e-03	2.32e-01	
	STD	**8.29e-08**	4.21e-01	2.21e-03	2.00e-03	9.07e-05	6.88e-02	
2	MIN	**3.54e-10**	6.42e-03	2.36e-05	3.32e-05	5.34e-05	2.83e-03	3.06e-04
	MEAN	**8.22e-10**	2.57e-02	7.68e-05	1.41e-04	6.42e-05	4.64e-03	
	MAX	**2.26e-09**	1.16e-01	1.57e-04	3.01e-04	7.06e-05	8.18e-03	
	STD	**6.05e-10**	3.34e-02	3.95e-05	9.30e-05	5.08e-06	1.73e-03	
3	MIN	**4.12e-08**	4.02e-01	2.07e-03	1.50e-03	3.26e-03	1.06e-01	1.73e-23
	MEAN	**9.07e-08**	9.66e-01	5.81e-03	6.59e-03	3.45e-03	2.33e-01	
	MAX	**1.64e-07**	1.54e+00	1.29e-02	1.00e-02	3.57e-03	3.67e-01	
	STD	**4.17e-08**	3.28e-01	2.99e-03	2.78e-03	9.13e-05	7.77e-02	
4	MIN	**2.14e-11**	8.26e-03	2.82e-05	4.48e-05	4.84e-05	1.53e-03	4.20e-08
	MEAN	**1.48e-10**	4.34e-02	2.01e-04	2.33e-04	6.26e-05	1.03e-02	
	MAX	**3.13e-10**	1.35e-01	4.05e-04	5.64e-04	8.99e-05	2.67e-02	
	STD	**9.20e-11**	3.73e-02	1.15e-04	1.96e-04	1.26e-05	6.72e-03	
5	MIN	**5.96e-09**	5.88e-03	4.01e-05	1.17e-04	7.66e-05	3.69e-03	4.48e-16
	MEAN	**2.61e-07**	1.36e-02	9.90e-05	2.36e-04	8.15e-05	1.27e+00	
	MAX	**7.55e-07**	2.47e-02	1.66e-04	3.51e-04	9.17e-05	1.59e+00	
	STD	**2.66e-07**	6.69e-03	4.79e-05	7.27e-05	5.17e-06	6.67e-01	

Table 14
Percentage of accuracy improvements for the WNNIBOA model in comparison to the other WNNs models in terms of the MIN MAE.

Example		Approximation Improvement (%)			
	WNNBOA	WNNPSO	WNNPSOA	WNNMBP	WNNDEV
1	100.00	99.96	99.98	100.00	100.00
2	100.00	100.00	100.00	100.00	100.00
3	100.00	100.00	100.00	100.00	100.00
4	100.00	100.00	100.00	100.00	100.00
5	100.00	99.99	99.99	99.99	100.00

compared with that of the existing RBFNs introduced by Rizaner and Rizaner [11]. The RBFNs, consisting of 21 hidden nodes, were trained using the conventional BP algorithm. Their AE values, calculated with available solutions up to four decimal places, are presented in Table 6. It can be observed that the AE values for the RBFNs range from 10^{-03} to 10^{-04}, which are surpassed by the superior range of AE values obtained by the proposed WNNIBOA method, ranging from 10^{-07} to 10^{-10}. Furthermore, a more recent study improved the RBFNs model [26] by incorporating the ELM training method. The WNNIBOA method was compared to this enhanced RBFNs model in terms of the average mean squared error (AMSE) criterion, given by:

$$AMSE = \frac{1}{NI}\sum_{n=1}^{NI}\left(\widehat{y}_n - y_n\right)^2 \tag{28}$$

where $\widehat{y}_n$ is the n-th exact solution, y_n is the n-th WNNs solution and NI is the total number of training points. When considering the same number of training points and hidden nodes, the AMSE achieved by the RBFNs model is 5.85e-12, whereas for the WNNIBOA model, the AMSE is significantly lower at 3.94e-15. This result clearly demonstrates the higher solution accuracy attained by the proposed WNNs method. It is important to note that both RBFNs and WNNs share a similar network design, differing primarily in their activation functions. Despite this similarity, the superiority of the WNNIBOA model can be attributed to the utilization of the robust IBOA as the training method.

Table 16
The values of dilation and translation of WNNs for ODE examples.

Example	1	2	3	4	5
Translation	[− 4,4]	[− 2,3]	[− 4,6]	[− 3,3]	[− 2,3]
Dilation	0.8	0.6	0.7	0.8	0.6

Subsequently, a comprehensive examination of the WNNs models was conducted using five testing points. The testing solutions obtained by each WNN model were compared and presented in Table 7. The results demonstrate that the WNNIBOA model exhibits exceptional generalization performance at the testing points, accurately matching the exact solutions with a precision of at least 10^{-08}.

Example 4. We consider a second-order non-homogeneous ODE of Lane-Emden type, given by:

$$\frac{d^2y}{dx^2} + \frac{2}{x}\frac{dy}{dx} + y = 6 + 12x + x^2 + x^3, x \in \left[0,1\right] \tag{29}$$

with the initial conditions $y(0) = 0$ and $y'(0) = 0$. The exact solution for this ODE is provided as $\widehat{y} = x^2 + x^3$ [41]. The trial solution for this equation is expressed as $y(x) = x^2 z(x)$. The Lane-Emden equation is frequently encountered in astrophysics and mathematical physics, as it describes various phenomena [9]. Solving this equation using traditional numerical techniques is challenging due to a singularity at the origin. Consequently, several numerical methods have been developed to address this type of ODE.

In this example, we discretized the interval [0,1] into twenty equidistant points with a step size of 0.05. The corresponding AE values for

Table 15
The best results of runtime outcomes from ten independent runs comparing WNNIBOA with others.

Example	WNNIBOA	WNNBOA	WNNPSO	WNNPSOA	WNNMBP	WNNDEV
1	3.0552	2.0753	7.4577	8.1047	1.8352	0.7300
2	3.7651	2.6616	7.8666	8.0811	3.0735	0.8007
3	4.4344	2.5103	8.2809	8.0182	2.7748	0.7886
4	3.7188	2.7231	9.1056	8.6559	1.9383	0.9702
5	3.2194	2.3515	8.1033	8.1682	2.3178	0.8721

all the WNNs models trained with different methods to solve the IVP were calculated and presented in Table 8. Additionally, Table 8 provides a comparison of the approximation accuracy between the proposed method and an existing ANNs model that utilizes the Morlet wavelet activation function, as reported in the literature [42]. The graphs of the AE, illustrating the discrepancies between the exact solution and the solutions obtained by the WNNs models, are shown in Fig. 7.

The results presented in Table 8 demonstrate the excellent agreement between the numerical solutions obtained by the proposed WNNIBOA method and the exact solution, with an extremely low order of AE ranging from 10^{-11} to 10^{-12}. Comparing the different training methods employed in the WNNs models, it is evident that the utilization of the IBOA training method leads to a significant improvement in terms of AE, as shown in Table 8 and Fig. 7. Moreover, when compared to the AE values achieved by the ANNs method described in [42], which typically exhibit an accuracy within 6–8 decimal places, the precision of the solutions obtained by the proposed WNNIBOA method surpasses those of the ANNs approach.

Table 9 displays the AE values for the testing solutions obtained from the WNNs models trained using different methods. The testing solutions generated by the WNNIBOA exhibit good agreement with the exact solution, achieving significantly lower AE values ranging from 10^{-11} to 10^{-13}. Furthermore, a notable observation is the substantial decrease in AE specifically at the first testing point compared to the other testing points across all WNNs models. This consistent trend, as seen in Example 1, suggests that the proximity of the points to the initial point provides an advantageous effect on the approximation accuracy, potentially attributed to the utilization of the trial solution.

Example 5. In this example, the WNNs are employed to solve a nonlinear singularly perturbed BVP of the second-order ODE described by:

$$-0.1\frac{d^2y}{dx^2}+y+y^2=e^{-\frac{2x}{\sqrt{0.1}}},x\in\left[0,1\right] \tag{30}$$

along with the boundary conditions $y(0)=1$ and $y(1)=e^{-\frac{1}{\sqrt{0.1}}}$.To satisfy these boundary conditions, the trial solution is formulated as $y(x)=1-x+xe^{-\frac{1}{\sqrt{0.1}}}+(x-x^2)z(x)$. The exact solution for this problem is given as$\hat{y}=e^{-\frac{x}{\sqrt{0.1}}}$. In order to facilitate a fair comparison with the findings of [43], the computational domain interval [0,1] was divided into fifteen equal segments to ensure consistency. The AE values for the WNNs models trained using six different training methods are presented in Table 10 and visually represented in Fig. 8.

The integration of the IBOA training method in the WNNs significantly improves their performance, as demonstrated in Fig. 8. The graph illustrates that the AE for the WNNIBOA aligns perfectly with the x-axis. Similarly, the results presented in Table 10 confirm that incorporating the IBOA training method in the WNNs results in a substantial reduction in the AE, ranging between 10^{-08} and 10^{-09}. These findings unequivocally establish the superior performance of the proposed model compared to other WNN models.

The performance of the proposed WNNs model was compared to the trigonometric quintic B-spline method, a numerical method, in terms of the MAX AE [43]. The numerical method achieved a MAX AE value of 5.82e-07, while the proposed WNNIBOA model achieved a significantly lower MAX AE of 1.39e-08. This result suggests that the proposed WNNs model has the potential to achieve similar or even higher performance compared to the numerical method.

Table 11 displays the AE values obtained by different WNNs models at various testing points. It can be observed that the WNNIBOA model demonstrates a good agreement with the exact solution at each testing point.

6. Performance analysis

The consistently low AE values achieved by WNNIBOA across all analyzed cases using both training and testing points (refer to Table 2 to Table 11) indicate the absence of under-fitting or over-fitting issues. The accuracy and convergence of the WNNs models were then analyzed by calculating the convergence rate percentages based on acceptability criteria at different levels of MAE. The results of ten independent runs are summarized in Table 12. The percentage of convergence rate was computed using the following formula:

$$\text{PercentageofConvergenceRate}=\frac{NP}{NT}\times 100\%, \tag{31}$$

where NP is the number at which the MAE of run achieves the acceptability criteria of MAE and NT is the total number of independent runs.

It can be observed from Table 12 that the WNNBOA and WNNDEV achieved a convergence rate of 0% for all levels of the acceptability criteria of MAE, except for the acceptability criterion of MAE$\leq 10^{-03}$ in most of the examples, which involve IVPs and BVPs with two conditions. The relatively improved performance of the WNNBOA and WNNDEV in these examples may be attributed to the presence of specific conditions in the IVPs and BVPs, which contribute to generating solutions that are closer to the exact solutions. Similarly, the WNNs trained with other methods also demonstrate satisfactory performance when considering the BVPs in Examples 2 and 5, as well as the IVP in Example 4. On the other hand, the WNNIBOA achieved a 100% success rate in meeting all levels of the acceptability criteria of MAE in all examples. The WNNIBOA was the only model that consistently satisfied the criterion of MAE$\leq 10^{-07}$ across all examples, distinguishing its superiority over other WNNs models in effectively solving IVPs and BVPs.

Statistical analysis of the MAE values obtained from ten independent runs for the six WNNs models was conducted, and the results are presented in Table 13. The MIN, mean, MAX, and STD values of the MAE were calculated and compared among the WNNs models. The bold font indicates the best values achieved among the WNNs models. The MIN represents the best MAE value, while the MAX represents the worst MAE value. The STD is used to evaluate the stability of convergence for each WNNs model, and the mean provides an assessment of the average performance of each method.

To statistically validate the performance of the proposed method, a one-way analysis of variance (ANOVA) was conducted. This test is used to determine whether there is a significant difference between the results obtained by the IBOA training method and the other training methods, namely BOA, PSO, PSOA, DEV and MBP. The null hypothesis states that all mean values are equal. The rejection of the null hypothesis occurs when the p-value obtained from the ANOVA test is less than 0.05, indicating a statistically significant difference. Table 13 displays the p-values obtained from the one-way ANOVA test for all the examples. It is evident that the p-values are significantly lower than 0.05 for all the examples, confirming that there is a substantial difference in the means of the MAE between the IBOA training method and the other methods, with a 95% confidence level. This provides strong evidence that the IBOA training method leads to a statistically significant improvement in training the WNNs. As observed from Table 13, the WNNIBOA model consistently outperforms the other WNNs models in terms of the MIN, mean, MAX, and STD values. These results are consistently superior across all examples, highlighting the superior approximation capability of the WNNIBOA model compared to its counterparts.

Table 14 summarizes the percentage of accuracy improvement achieved by the WNNIBOA model compared to other training methods, specifically BOA, PSO, PSOA, DEV and MBP, based on the MIN MAE values. The results clearly demonstrate a remarkable accuracy improvement of nearly 100% across all examples when utilizing the WNNIBOA model. This significant enhancement validates the efficacy of the IBOA training method in effectively determining optimal weights for

WNNs.

The convergence of each WNNs model across ten independent runs is evaluated by analyzing the MIN and MAX MAE values presented in Table 13. The WNNBOA, WNNPSO, WNNPSOA, WNNDEV and WNNMBP exhibit MIN and MAX MAE within the range of 10^{+00} to 10^{-05}. In contrast, the WNNIBOA, using the same WNN architecture, achieves significantly lower MAE values ranging from 10^{-07} to 10^{-11}, indicating more accurate and convergent solutions compared to the other WNNs models across all runs. Notably, for all ODEs considered, the MAX MAE of the WNNIBOA outperforms the MIN MAE of the WNNs models trained with alternative algorithms, further confirming the exceptional accuracy of the WNNIBOA. Additionally, the WNNIBOA demonstrates the lowest STD MAE, indicating a high level of convergence stability for the IBOA.

Table 15 presents the runtime outcomes from ten independent runs comparing WNNIBOA against other WNN models. Results indicated that the DEV algorithm demonstrates the fastest performance among metaheuristic approaches for this specific problem, primarily due to employing fewer iterations. However, the achieved MIN MAE by WNNDEV reaches only up to 10^{-03}. In contrast, IBOA achieves superior accuracies in training WNNs within a reasonable timeframe. Therefore, it can be inferred that IBOA stands out as the optimal choice for training WNNs in ODE problem-solving.

7. Initialization of WNNs parameters

In this section, the impact of utilizing different dilation and translation values in the numerical simulations are discussed, as they significantly impact the shapes of wavelet activation functions and their derivatives. Table 16 presents the specific values of dilation and translation for each ODE example.

The range of translation is determined heuristically based on the training points of an ODE, ensuring that the computational domain of the ODE falls within the selected range. The goal is to stretch the wavelet activation functions rather than shrink them. If the translation range is smaller than the computational domain, it can result in overly localized responses, which may impede the learning process of WNNs. Therefore, it is important to choose a maximum value for translation that is larger than the right endpoint of the computational domain, and a minimum value that is smaller than the left endpoint. Generally, a value difference ranging from one to four is considered acceptable.

Meanwhile, the values of the dilation parameter are determined heuristically based on knowledge of the ODEs and wavelet activation functions. As discussed in Section 3, the first and second order derivatives of the wavelet activation functions with respect to the input variable x in Eq. (13) and Eq. (14), respectively, are computed using the chain rule.

$$\begin{aligned}\frac{d\varphi}{dx} &= \frac{d\varphi}{dh}\frac{dh}{dx}\\ &= \varphi'\left(\frac{1}{d}\right)\end{aligned} \tag{32}$$

$$\begin{aligned}\frac{d^2\varphi}{dx^2} &= \frac{d^2\varphi}{dh^2}\left(\frac{dh}{dx}\right)^2 + \frac{d\varphi}{dh}\left(\frac{d^2h}{dx^2}\right)\\ &= \varphi''\left(\frac{1}{d}\right)^2 + 0\\ &= \varphi''\left(\frac{1}{d^2}\right)\end{aligned} \tag{33}$$

where φ' and $\varphi''$are given in Eq. (3) and Eq. (4), respectively, h is the input function of the hidden layer and d is the dilation parameter. As can be seen from Eq. (32) and Eq. (33), the power of the dilation parameter d is incremented by 1 for each subsequent order of the derivatives of the wavelet activation functions. Consequently, if the value of the dilation parameter is less than 1, the bounds of the derivatives of the wavelet activation functions expand beyond their original bounds. Conversely, if the value of the dilation parameter is greater than 1, the bounds of the derivatives contract within their original bounds.

The bounds of the derivatives of the wavelet activation functions are carefully analyzed to ensure that the chosen dilation values maintain a balance between the left-hand side and the right-hand side of the approximate values of an ODE. For instance, in the case of Example 4, where the right-hand side of the ODE falls within the range of [6,20] for the training domain [0,1], it is necessary for the sum of the terms $\frac{d^2y}{dx^2}$, $\frac{dy}{dx}$and the y terms on the left-hand side of the ODE to exceed 6 in order to equate to the right-hand side. Considering the relatively small and close range of the GW activation function and its derivatives, i.e., [− 0.61, 0.61], [− 1, 0.45], and [− 1.38, 1.38], respectively, it is expected that higher bounds are required for the derivatives of the GW activation function. The selection of dilation values for the remaining ODE examples is based on a careful examination of the ODEs, considering the substitution of initial conditions and evaluating both sides of the ODEs. By analyzing the initial values of the solutions and their first-order derivatives in Examples 1 and 3 (i.e., $y(0) = 0.5, \frac{dy(0)}{dx} = 1.5$ and $y(0) = 3$, $\frac{dy(0)}{dx} = -5$), it is expected that the bounds of the first-order derivatives of the GW activation function would be slightly higher in Example 3 compared to Example 1. Conversely, Examples 2 and 5 exhibit a larger difference between the initial values of the solutions and their second-order derivatives, i.e., $y(0) = 1$ and $\frac{d^2y(0)}{dx^2} = 10$. Consequently, much higher bounds of the second-order derivatives of the GW activation function are expected in order to increase the gap between the bounds of the wavelet activation functions and the bounds of their second-order derivatives. This can be achieved by using a smaller dilation value compared to Examples 1 and 3. The numerical simulation results demonstrate the effectiveness of the initialized dilation and translation values for the WNNs, indicating their promising performance.

8. Conclusions

In this study, a neural intelligent computational approach that combines WNNs and the IBOA is proposed to enhance the accuracy of solving first and second order ODEs compared to existing ANNs and numerical methods. The WNNs outputs with adjustable weights parameters are used in formulating the approximate solutions while the IBOA training method is employed to search an optimal set of weights for the WNNs such that the trial solutions satisfy the ODEs as close as possible. The training and testing results demonstrated that the WNNIBOA model outperformed other WNNs models, achieving highly accurate solutions for all considered ODEs. The AE values for the WNNIBOA model range from 10^{-07} to 10^{-13}, which is significantly smaller compared to the AE ranges of 10^{+00} to 10^{-08} obtained by the WNNBOA, WNNMBP, WNNDEV, WNNPSOA and WNNPSO models. Notably, the IBOA training method exhibited the most notable improvement in accuracy, particularly at the endpoint for the IVPs. This superiority is attributed to the IBOA's ability to avoid local optima and its effectiveness in finding optimal weight sets for the WNNs. Furthermore, when comparing the proposed WNNIBOA method with other ANNs and numerical methods, i.e., RBFNs and Heun's methods, the WNNIBOA model demonstrated superior approximation ability. The significant differences observed in the AE values between the WNNIBOA and its counterparts further validate the superiority and effectiveness of the proposed WNNIBOA model for solving ODEs.

To gain a comprehensive understanding of the performance of WNNs models trained with different algorithms, a statistical analysis was conducted using one-way ANOVA. This analysis examined the MIN, mean, MAX, and STD values of the MAE across ten independent runs for each of the six WNNs models. The results consistently demonstrated that the WNNIBOA method outperformed other training algorithms in terms of achieving the lowest MIN, mean, MAX, and STD values across all considered linear and nonlinear ODEs. The statistical significance of the performance differences was confirmed by the small p-values, indicating that the improvement achieved by the WNNIBOA method over WNNBOA, WNNMBP, WNNDEV, WNNPSOA and WNNPSO was highly significant with a confidence level of 95%. This substantial accuracy enhancement of nearly 100% further highlights the effectiveness and superiority of the WNNIBOA method in solving ODEs.

Despite the promising results achieved by the proposed WNNs method for solving ODEs, there are certain limitations that should be acknowledged. Firstly, careful parameter initialization is crucial for achieving optimal solution accuracy in the WNNs model. Further research is warranted to explore more sophisticated techniques for the initialization of translation and dilation parameters, which may lead to enhanced performance. Another limitation of the proposed method lies in its applicability limited to first and second order ODEs. Future studies could focus on extending the capabilities of WNNs to handle higher order ODEs, which present additional challenges and complexities. Exploring the adaptation of WNNs for higher order ODEs would expand the range of problems that can be effectively addressed using this approach.

CRediT authorship contribution statement

Tan Lee Sen: Writing – review & editing, Writing – original draft, Methodology, Formal analysis, Data curation, Conceptualization. **Abdullah Farah Aini:** Supervision, Conceptualization. **Ong Pauline:** Writing – review & editing, Writing – original draft, Supervision. **Zainuddin Zarita:** Writing – review & editing, Writing – original draft, Supervision, Resources, Project administration, Funding acquisition.

Declaration of Competing Interest

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

Data Availability

No data was used for the research described in the article.

Acknowledgments

The authors gratefully acknowledge the financial assistance provided by Universiti Sains Malaysia under the Research University Grant 1001/PMATHS/8011041 and Bridging Research Grant 304/PMATHS/6316079.

References

[1] I.E. Lagaris, A. Likas, D.I. Fotiadis, Artificial neural networks for solving ordinary and partial differential equations, IEEE Trans. Neural Netw. vol. 9 (5) (1998) 987–1000.
[2] S.P. Fard, J. Pouramini, An investigation of approximate solutions for second order ordinary differential equations using sigmoid-weighted neural networks, Int. J. Appl. Comput. Math. vol. 8 (3) (2022) 103.
[3] Y. Wen, T. Chaolu, X. Wang, Solving the initial value problem of ordinary differential equations by Lie group based neural network method, Plos One vol. 17 (4) (2022) e0265992.
[4] S. Li, X. Wang, Solving ordinary differential equations using an optimization technique based on training improved artificial neural networks, Soft Comput. vol. 25 (2021) 3713–3723.
[5] Z. Zainuddin, P. Ong, An effective and novel wavelet neural network approach in classifying type 2 diabetics, Neural Netw. World vol. 22 (5) (2012) 407.
[6] S. Mall, S. Chakraverty, Application of Legendre neural network for solving ordinary differential equations, Appl. Soft Comput. vol. 43 (2016) 347–356.
[7] B. Yang, H. Wang, Prediction of Shanghai Index based on Additive Legendre Neural Network, in: *MATEC Web of Conferences*, vol. 95, EDP Sciences, 2017, p. 19001.
[8] P. Ong, Z. Zainuddin, Optimizing wavelet neural networks using modified cuckoo search for multi-step ahead chaotic time series prediction, Appl. Soft Comput. vol. 80 (2019) 374–386.
[9] S. Mall, S. Chakraverty, Chebyshev neural network based model for solving Lane–Emden type equations, Appl. Math. Comput. vol. 247 (2014) 100–114.
[10] S. Mall, S. Chakraverty, Single layer Chebyshev neural network model for solving elliptic partial differential equations, Neural Process. Lett. vol. 45 (2017) 825–840.
[11] F.B. Rizaner, A. Rizaner, Approximate solutions of initial value problems for ordinary differential equations using radial basis function networks, Neural Process. Lett. vol. 48 (2018) 1063–1071.
[12] A. Verma, M. Kumar, Numerical solution of Lane–Emden type equations using multilayer perceptron neural network method, Int. J. Appl. Comput. Math. vol. 5 (2019) 1–14.
[13] A. Jafarian, S.M. Nia, A.K. Golmankhaneh, D. Baleanu, On artificial neural networks approach with new cost functions, Appl. Math. Comput. vol. 339 (2018) 546–555.
[14] S. Effati, M. Pakdaman, Artificial neural network approach for solving fuzzy differential equations, Inf. Sci. vol. 180 (8) (2010) 1434–1457.
[15] M. Pakdaman, A. Ahmadian, S. Effati, S. Salahshour, D. Baleanu, Solving differential equations of fractional order using an optimization technique based on training artificial neural network, Appl. Math. Comput. vol. 293 (2017) 81–95.
[16] M.A.Z. Raja, R. Samar, Numerical treatment for nonlinear MHD Jeffery–Hamel problem using neural networks optimized with interior point algorithm, Neurocomputing vol. 124 (2014) 178–193.
[17] M.A.Z. Raja, J.A. Khan, A.M. Siddiqui, D. Behloul, T. Haroon, R. Samar, Exactly satisfying initial conditions neural network models for numerical treatment of first Painlevé equation, Appl. Soft Comput. vol. 26 (2015) 244–256.
[18] M.A.Z. Raja, M.A. Manzar, R. Samar, An efficient computational intelligence approach for solving fractional order Riccati equations using ANN and SQP, Appl. Math. Model. vol. 39 (10-11) (2015) 3075–3093.
[19] M.A.Z. Raja, R. Samar, M.A. Manzar, S.M. Shah, Design of unsupervised fractional neural network model optimized with interior point algorithm for solving Bagley–Torvik equation, Math. Comput. Simul. vol. 132 (2017) 139–158.
[20] M.A.Z. Raja, F.H. Shah, M. Tariq, I. Ahmad, S. u I. Ahmad, Design of artificial neural network models optimized with sequential quadratic programming to study the dynamics of nonlinear Troesch's problem arising in plasma physics, Neural Comput. Appl. vol. 29 (2018) 83–109.
[21] S. Lodhi, M.A. Manzar, M.A.Z. Raja, Fractional neural network models for nonlinear Riccati systems, Neural Comput. Appl. vol. 31 (2019) 359–378.
[22] H. Sun, M. Hou, Y. Yang, T. Zhang, F. Weng, F. Han, Solving partial differential equation based on Bernstein neural network and extreme learning machine algorithm, Neural Process. Lett. vol. 50 (2019) 1153–1172.
[23] H. Liu, B. Xing, Z. Wang, L. Li, Legendre neural network method for several classes of singularly perturbed differential equations based on mapping and piecewise optimization technology, Neural Process. Lett. vol. 51 (2020) 2891–2913.
[24] Y. Yang, M. Hou, H. Sun, T. Zhang, F. Weng, J. Luo, Neural network algorithm based on Legendre improved extreme learning machine for solving elliptic partial differential equations, Soft Comput. vol. 24 (2020) 1083–1096.
[25] S. Panghal, M. Kumar, Optimization free neural network approach for solving ordinary and partial differential equations, Eng. Comput. vol. 37 (2021) 2989–3002.
[26] M. Liu, W. Peng, M. Hou, Z. Tian, "Radial basis function neural network with extreme learning machine algorithm for solving ordinary differential equations, Soft Comput. vol. 27 (7) (2023) 3955–3964.
[27] N. Mai-Duy, T. Tran-Cong, Numerical solution of differential equations using multiquadric radial basis function networks, Neural Netw. vol. 14 (2) (2001) 185–199.
[28] A. Malek, R.S. Beidokhti, Numerical solution for high order differential equations using a hybrid neural network—optimization method, Appl. Math. Comput. vol. 183 (1) (2006) 260–271.
[29] R.S. Beidokhti, A. Malek, Solving initial-boundary value problems for systems of partial differential equations using neural networks and optimization techniques, J. Frankl. Inst. vol. 346 (9) (2009) 898–913.
[30] M.A.Z. Raja, S.-u-I. Ahmad, R. Samar, Solution of the 2-dimensional Bratu problem using neural network, swarm intelligence and sequential quadratic programming, Neural Comput. Appl. vol. 25 (2014) 1723–1739.
[31] M.A.Z. Raja, S. Abbas, M.I. Syam, A.M. Wazwaz, Design of neuro-evolutionary model for solving nonlinear singularly perturbed boundary value problems, Appl. Soft Comput. vol. 62 (2018) 373–394.
[32] Z. Sabir, D. Baleanu, M. Shoaib, M.A.Z. Raja, Design of stochastic numerical solver for the solution of singular three-point second-order boundary value problems, Neural Comput. Appl. vol. 33 (2021) 2427–2443.
[33] Z. Sabir, S. Saoud, M.A.Z. Raja, H.A. Wahab, A. Arbi, Heuristic computing technique for numerical solutions of nonlinear fourth order Emden–Fowler equation, Math. Comput. Simul. vol. 178 (2020) 534–548.
[34] L.S. Tan, Z. Zainuddin, P. Ong, Wavelet neural networks based solutions for elliptic partial differential equations with improved butterfly optimization algorithm training, Appl. Soft Comput. (2020) 106518.
[35] Q. Zhang, A. Benveniste, Wavelet networks, IEEE Trans. Neural Netw. vol. 3 (6) (1992) 889–898.
[36] Z. Zainuddin, O. Pauline, Modified wavelet neural network in function approximation and its application in prediction of time-series pollution data, *Appl. Soft Comput. J.* vol. 11 (8) (2011) 4866–4874, https://doi.org/10.1016/j.asoc.2011.06.013.
[37] S. Arora, S. Singh, Butterfly optimization algorithm: a novel approach for global optimization, Soft Comput. vol. 23 (2019) 715–734.

[38] S. Sai Rayala, N. Ashok Kumar, Particle Swarm Optimization for robot target tracking application, /01/01/ 2020, Mater. Today.: Proc. vol. 33 (2020) 3600–3603, https://doi.org/10.1016/j.matpr.2020.05.660.

[39] T.I. Haweel, T.N. Abdelhameed, Power series neural network solution for ordinary differential equations with initial conditions. 2015 International Conference on Communications, Signal Processing, and their Applications (ICCSPA'15), IEEE,, 2015, pp. 1–5.

[40] X. Li-ying, W. Hui, Z. Zhe-zhao, The algorithm of neural networks on the initial value problems in ordinary differential equations. 2007 2nd IEEE Conference on Industrial Electronics and Applications, IEEE, 2007, pp. 813–816.

[41] S.A. Yousefi, Legendre wavelets method for solving differential equations of Lane–Emden type, Appl. Math. Comput. vol. 181 (2) (2006) 1417–1422.

[42] Z. Sabir, H.A. Wahab, M. Umar, M.G. Sakar, M.A.Z. Raja, Novel design of Morlet wavelet neural network for solving second order Lane–Emden equation, Math. Comput. Simul. vol. 172 (2020) 1–14.

[43] K. K. Ali, A.R. Hadhoud, M.A. Shaalan, Numerical study of self-adjoint singularly perturbed two-point boundary value problems using collocation method with error estimation, /09/01/ 2018, J. Ocean Eng. Sci. vol. 3 (3) (2018) 237–243, https://doi.org/10.1016/j.joes.2018.07.001.