Journal of Applied Analysis and Computation
Volume 12, Number 4, August 2022, 1277–1293
Website:http://www.jaac-online.com
DOI:10.11948/20200377

AN ALGORITHM FOR NUMERICAL SOLUTION OF DIFFERENTIAL EQUATIONS USING HARMONY SEARCH AND NEURAL NETWORKS

Neha Yadav[1,†], Thi Thuy Ngo[2] and Joong Hoon Kim[3]

Abstract In this article, an algorithm based on artificial neural networks (ANN) and harmony search algorithm (HSA) is presented for the numerical solution of ordinary and partial differential equations. The power of ANN is used to construct an approximate solution of differential equations (DEs) such that it satisfies the DEs initial conditions (ICs) or boundary conditions (BCs) automatically. An automated design parameter selection approach is utilised to pick the optimum ANN ensemble from various combinations of ANN design parameters, random beginning weights, and biases. An unsupervised error is constructed in order to approximate the DE solution and HSA is used to minimize this error by training the neural network design parameters. A few test problems of various types are considered for verifying the algorithm's accuracy, convergence, and efficacy. The proposed algorithm is assessed using the results of statistical analysis obtained from a large number of independent runs for each model equation. The correctness and validity of the algorithm is also verified by comparing the obtained numerical results to the exact solution.

Keywords Differential equations, harmony search algorithm, artificial neural networks, length factor.

MSC(2010) 65N99, 65Y99.

1. Introduction

Differential equations are frequently used to model problems in real-world applications (DEs). In some circumstances, approximating the solution of DEs is a difficult task, necessitating the use of numerical techniques. The finite difference, finite element, finite volume, and boundary element are commonly used methods to solve DEs numerically [1] . All of these approaches are effective and efficient in solving DEs, but they have limitations, such as the need for mesh discretization and the difficulty of solving nonlinear DEs.

ANNs have evolved as a sophisticated approach for numerically solving DEs because they avoid the drawbacks of traditional numerical methods. In recent

[†]The corresponding author. Email: nehayadav441@gmail.com (N. Yadav)
[1]Department of Mathematics and Scientific Computing, National Institute of Technology Hamirpur, Hamirpur, 177005, H.P., India
[2]Vietnam Institute of Meteorology, Hydrology and Climate Change, 10000 Hanoi, Vietnam
[3]School of Civil, Environmental and Architectural Engineering, Korea University, 136-713 Seoul, South Korea

research, the ANN method has been employed to tackle several DE problems, and some variants of the ANN method have also been developed to handle DE problems. A survey article [13] lists about 50 research that use ANNs to solve DEs. Various forms of DEs, including ordinary DEs (ODEs), partial DEs (PDEs), and fractional DEs (FDEs), have been solved using the ANN method in recent research [14–16, 19, 29]. For network parameter optimization, the majority of the studies described above used gradient-based local search optimization methods. Furthermore, there are two main problems in these methods: the first is the necessity of a differentiable transform function, and the second is the high risk of trapping in local optima.

In order to avoid the local optima convergence, many global search optimization algorithms have been embraced for training ANN for solving DEs, especially for some nonlinear DE problems that are problematic while using gradient-based algorithms [28]. Several ANN methods based on evolutionary computation and swarm intelligence have thus been utilised to address various DE problems in the literature [20, 21, 26].

The HSA is a music-inspired heuristic optimization algorithm developed by Geem et al. in 2001 [7]. Since then, the technique has been used to handle a wide range of optimization issues,including function optimization, groundwater modelling, water distribution networks,and vehicle routing etc. Several HSA variants have also been developed in the literature and can be seen at [10, 12]. The HSA has already been shown more efficient and more straightforward than other optimization techniques. All these factors motivate us to explore the applicability of the HSA for training ANN to solve some linear as well as nonlinear DEs.

In this paper, a standard HSA is used for training the feed-forward ANN model of DEs. The rest of the paper is organized as follows. Section 2 gives brief idea of the ANN method for solving DEs. Overview of HSA and implementation of HSA for training ANN parameters is described in Section 3 and Section 4 respectively. Section 5 is devoted to the test problems, their numerical formulation, and simulation. A statistical analysis of results has been presented in Section 6, and the conclusions drawn are presented in Section 7.

2. The artificial neural network method

The ANN method to solve DE can be described by considering a general k^{th} order DE of the form

$$F\left(y^{(k)}(x), \ldots, y'(x), y(x)\right) = f(x) \tag{2.1}$$

over the domain D. In (2.1) F represents some differential operator and $y(x)$ is analytical solution subject to the following BC 's, with $p(x)$ being the BC on a boundary

$$\begin{aligned} &y = p(x), \quad \forall x \in \partial D, \\ &\hat{n}(x).\nabla y(x) = p(x), \quad \forall x \in \partial D. \end{aligned} \tag{2.2}$$

In (2.2) $\hat{n}(x)$ represents the unit vector function normal to the boundary ∂D. The first step in ANN is to construct a trial approximate solution for the given DE problem. In the literature, there are many ways to write down the DE trial solution in terms of ANN in such a way that it satisfies the desired BCs [14–16]. Here we consider the following type of trial approximate solution for (2.1)

$$y_T(\bar{x}, \bar{w}) = A(\bar{x}) + B(\bar{x}, N(\bar{x}, \bar{w})). \tag{2.3}$$

In (2.3) y_T represents the trial approximate solution, $\bar{x}$ is the input vector (points in the domain), $\bar{w}$ is the ANN weight parameters (weights and biases). The first term $A(\bar{x})$ denotes a continuous function that is formed to satisfy the BCs, while the second term denotes an ANN with input and weight parameters. Regardless of the ANN output $N(\bar{x}, \bar{w})$, the trial solution specified in (2.3) satisfies the desired BCs defined in (2.2). In a similar way, the value of trial solution within the domain is optimized by ANN output $N(\bar{x}, \bar{w})$. Training in ANN refers to the optimization of ANN parameters by minimizing the DE error function.

$$E(\bar{x}, \bar{w}) = f\left(\bar{x}, F\left(y_T^k(\bar{x}), \ldots, y_T'(\bar{x}), y_T(\bar{x})\right)\right). \tag{2.4}$$

There are several techniques based on the gradient and non gradient methods used in the literature for minimizing the DE error function of type given in (2.4). Here, in this study we present a non gradient based method HSA for training ANN parameters and minimizing the error function. A feed forward neural network with a single hidden layer is commonly used network architecture of ANNs used in this study. A sigmoid activation function is used in hidden nodes for transferring approximate decisions, whereas a linear activation function is used at the output node.

3. Harmony search algorithm

The HSA is a population-based meta-heuristic algorithm developed by Geem et al. [7] and inspired by the improvisation method used by musicians. During the improvisation process, each musician plays a note to find the best state of harmony altogether. This procedure is analogous to finding the optimality in an optimization process. The "harmony at a time represents a solution vector in which; each musical instrument is identical to a decision variable; musical instruments' pitch range corresponds to search space, and the audience's esthetics represent an objective function" [18]. The improvisation process is mimicked by local and global searches in optimization. Generally, global optimization problems can be defined as follows

$$\text{Optimize (Max or Min)} f(x), \tag{3.1}$$

$$\text{subject to } \; x_j \in [L_j, U_j], j = 1, 2, 3, \ldots, N, \tag{3.2}$$

where $f(x)$ is an objective function; x is the set of each decision variable x_j; N is the number of decision variable, and $[L_j \leq x_j \leq U_j]$ are the possible upper and lower bound for each decision variable.

In a standard HSA, an initial population of harmonies equal to the harmony memory size (HMS) are randomly created and stored in harmony memory (HM). Each iteration uses three major operators to create new harmony: harmony memory consideration rate (HMCR $[0, 1]$), pitch adjustment rate (PAR $[0, 1]$), and random consideration. The HMCR parameter controls the probability of new harmony improvisation considering harmonies in HM, whereas the PAR parameter controls the pace at which new harmony is generated in the bandwidth (BW) of variables. The worst harmony in HM would be replaced with a superior new harmony. This method is repeated until the desired number of improvisation (NI) has been reached or when the stopping criterion satisfies. The general procedure of the standard HSA is described in Algorithm 1.

Algorithm 1 Pseudo-code of HSA

```
1: Initialization: First HSA parameters are initialized.
2: Calculating the fitness function value of each harmony vector.

        $X_j^i = L_j + a \times (U_j - L_j)\,;$ where $a \in (0,1)$

3: Update of new harmony ($X_{new}$) from HM as:
   For ($k = 1$ to $n$) do
   If ($a_1 < HMCR$) then $X_{new}(k) = X_o(k)$; where $o \in (1, 2, \dots, HMS)$ and
   $a_1 \in (0,1)$
   If ($a_2 < PAR$) then $X_{new}(k) = X_{new}(k) \pm a_3 \times BW$; where $a_2, a_3 \in (0,1)$
   else $X_{new}(k) = L_j(k) + a \times (U_j(k) - L_j(k))$; where $a \in (0,1)$
   End for

4: If $f(X_{new})$ is better than $f(X_{\text{worst}})$, update HM as $X_{\text{worst}} = X_{\text{new}}$.
5: Repeat steps 3 and 4 until NI is reached. Harmony vector $X_{\text{best}}$ in the HM is
   the solution of the defined problem.
```

4. Training the ANN

Training the ANN or optimizing the network parameters can be performed by using the HSA [12, 18]. For training ANN using HSA, weight parameters w of ANNs is represented by a harmony vector in HM. Separate data strings are used in HM to represent the weight parameters w of ANNs i.e., weights corresponding to the hidden layer processing element from the input layer, output layer processing element from hidden layer, hidden biases, and output biases, all are denoted by separate strings of harmony vector in HM. The fitness function for the HSA for training the ANN is the minimization of the DE error defined by (2.4). For all harmony vectors in HM, weights are randomly generated and updated using (4.1):

$$w_j^i = L_j + a \times (U_j - L_j)\,. \tag{4.1}$$

where, $i = 1, 2, \dots, \text{HMS}$ and $j = 1, 2, \dots, N$. The weights in the feed-forward ANN can be updated by the algorithm step given in Table 1 . The detailed computational procedure of the ANN-HSA method for obtaining approximate solutions of DEs is presented in Figure 1.

Table 1. Improvisation of new harmony for training feed forward ANNs

For ($k = 1$ to n) do
If ($a_1 < HMCR$) then $w_{\text{new}}(k) = w_0(k)$; where $o \in (1, 2, \dots, HMS)$ and $a_1 \in (0,1)$
If ($a_2 < PAR$) then $w_{\text{new}}(k) = w_{\text{new}}(k) +$ rand $\times BW_k$; where $a_2 \in (0,1)$
End if
else $w_{\text{new}}(k) = L_j(k) + \text{rand} \times (U_j(k) - L_j(k))$;
End for
If $f(w_{\text{new}}) < f(w_{\text{worst}})$, update HM as $w_{\text{worst}} = w_{\text{new}}$

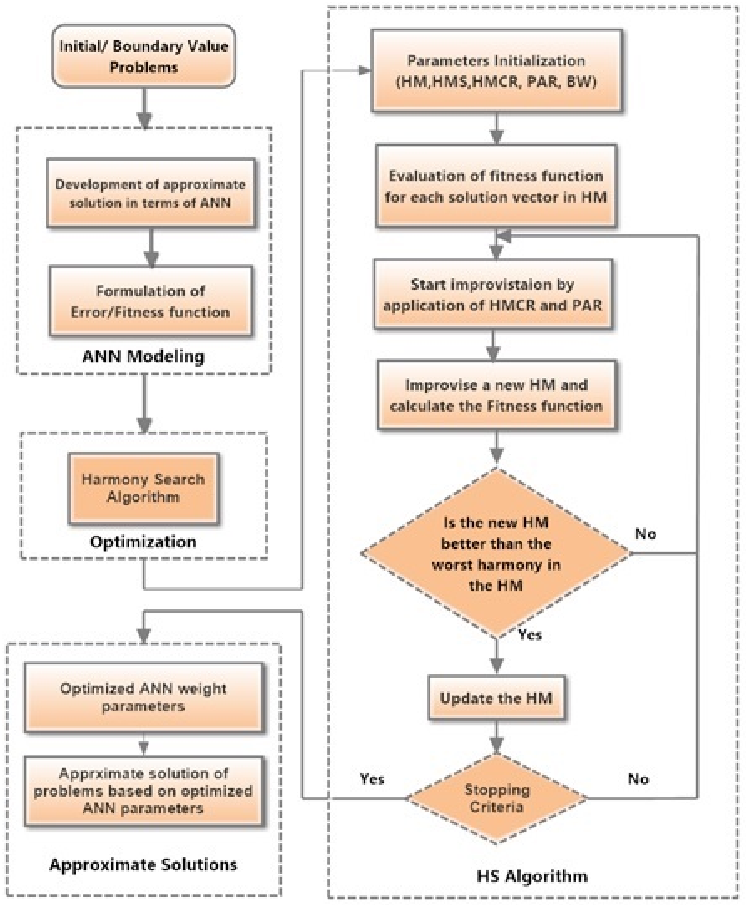
Initial/ Boundary Value Problems
Development of approximate solution in terms of ANN
Formulation of Error/Fitness function
ANN Modeling
Harmony Search Algorithm
Optimization
Optimized ANN weight parameters
Apprximate solution of problems based on optimized ANN parameters
Approximate Solutions
Parameters Initialization (HM,HMS,HMCR, PAR, BW)
Evaluation of fitness function for each solution vector in HM
Start improvistaion by application of HMCR and PAR
Improvise a new HM and calculate the Fitness function
Is the new HM better than the worst harmony in the HM
No
Yes
Update the HM
Stopping Criteria
Yes
No
HS Algorithm

Figure 1. Procedure to compute approximate solution of DEs using the ANN and HSA

5. Numerical Examples

In the previous sections, the proposed algorithm for the numerical solution of DEs using ANN and HSA was discussed. In this segment, we mean to explore the applicability of the algorithm to various test problems and analyze its precision. The proposed algorithm can be applied to various types of ODEs as well as PDEs. In order to demonstrate the illustrative applicability of the algorithm, we consider five test problems that cover linear as well as nonlinear equations with different types of initial or BCs. Numerical simulation for all the test problems was performed by training ANN using the HSA considering different values of the number of hidden nodes and starting weight parameters. The number of hidden nodes providing the lowest standard deviation (STD) in the DE error represented for 100 runs is considered as the best number of nodes, and the ANN solution is then computed by selecting the best ANN parameters. In this study we have considered a number of hidden nodes $h = 5, 10, 15, 20, 25, 30$ and 100 random starting weight parameters to find the best ANN node parameter. Numerical outcomes of the proposed algorithm are compared with the exact solution for all the test problems.

5.1. Test Problem 1

Troesch's problem defined by Weibel in [24] is an inherently unstable boundary value problem of the nonlinear ODE. The governing modelled equation and BCs for Troesch's problem stated in [23] are as follows:

$$\begin{aligned} &\frac{d^2u}{dt^2} = \mu \sinh(\mu u(t)); \quad t \in (0,1), \mu > 0 \\ &u(0) = 0, u(1) = 1 \end{aligned} \tag{5.1}$$

where μ is a constant parameter. The closed-form solution of (5.1) can be written in the form of an elliptic Jacobian function as

$$u(t) = \frac{2}{\mu} \sin h^{-1} \left\{ \frac{u'(0)}{2} sc \left(\mu x, 1 - \frac{1}{4} u'(0)^2 \right) \right\}, \tag{5.2}$$

where $u'(0) = 2\sqrt{1-k}, k$ is the solution of the equation $\frac{\sin h(\mu/2)}{\sqrt{1-k}} = \mathrm{sc}(\mu, k)$. The trial solution of (5.1) in terms of ANN parameters can be written as

$$u_T(t,w) = t + t(t-1)N(t,w). \tag{5.3}$$

Regardless of the ANN output, the trial solution provided by (5.3) satisfies the necessary BCs.The ANN solution of (5.1) is computed for $\mu = 0.5$ and 1. From Figure 2., it is observed that for the case $\mu = 0.5, h = 10$ number of hidden nodes provides the solution with minimum STD and for $\mu = 1, h = 15$ number of hidden nodes provides the solution with minimum STD in the DE error with 100 different starting weights.

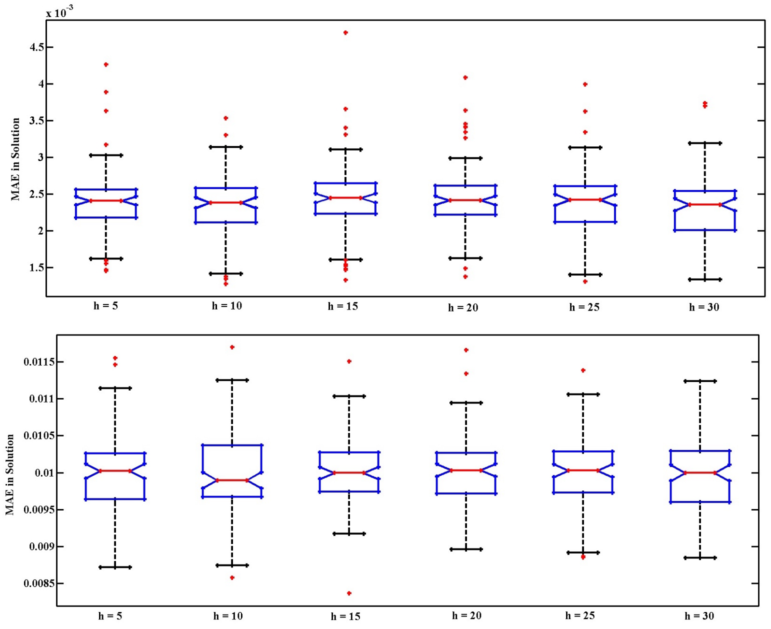

Figure 2. Boxplots for MAE in the solution with 100 runs for solving test problem 1 with (a)$\mu = 0.5$ and (b)$\mu = 1.0$

The error in the ANN solution throughout the domain can be observed by comparison with the exact solution and is presented in Figure 3 for $\mu = 0.5$ and 1.

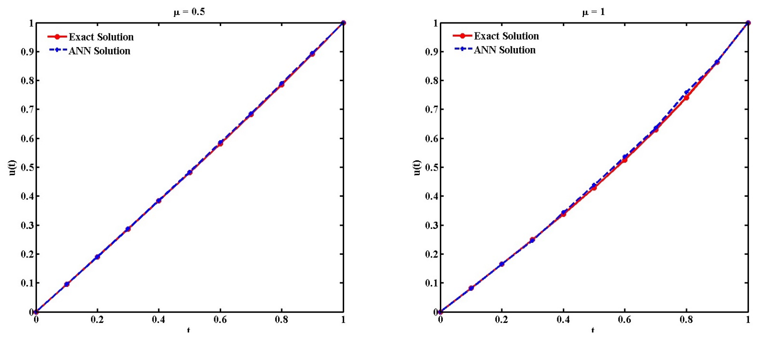

Figure 3. Exact and ANN solutions for test problem 1 for (a)$\mu = 0.5$ and (b)$\mu = 1.0$

5.2. Test Problem 2

The second problem considered is a nonlinear DE having BCs different from that in the test problem 1. The type of DE considered as test problem 2 is known as the Bernoulli equation, which is a useful second-order boundary value problem. Here, we consider a Bernoulli equation of the type shown below [1]:

$$\begin{aligned}&\frac{d^2u}{dx^2} + \left(\frac{du}{dx}\right)^2 - 2e^{-u(x)}; t \in (0,1)\\&u(0) = 0, \quad u(1) = 0.\end{aligned} \tag{5.4}$$

The exact analytical solution of (5.4) can be calculated by transforming the equation into a linear DE and is given as

$$u(x) = \ln\left((x - 1/2)^2 + 3/4\right). \tag{5.5}$$

In terms of ANN parameters, the trial solution of (5.4) can be written as

$$u_T(x, w) = x(x - 1)N(x, w). \tag{5.6}$$

Regardless of the ANN output (5.6) satisfies the necessary BCs. After that, the ANN is trained to find the best parameter combination. Figure 4. shows the boxplots for MAE in the solution for test problem 2 for different numbers hidden nodes.

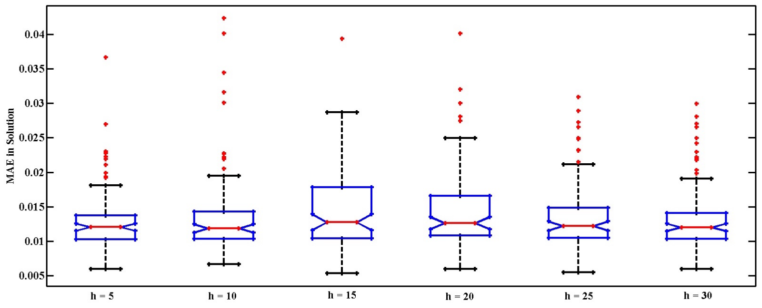

Figure 4. Boxplots for MAE in the solution for 100 runs for solving test problem 2

From Figure 4, we find that $h = 5$ number of hidden nodes gives the minimum STD in the DE error for 100 independent runs. The error in the ANN solution for (5.4) throughout the domain is then computed. Figure 5 shows a comparison of the ANN solution and the exact solution.

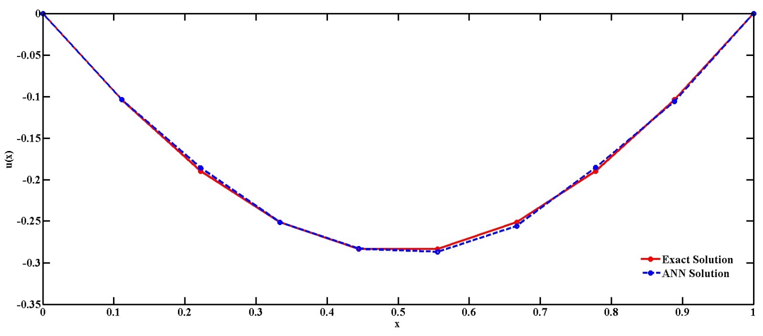

Figure 5. Exact and ANN solutions for test problem 2

5.3. Test Problem 3

Following that, we consider a nonlinear Riccati DE, which is used in a wide range of science and engineering applications, including random processes, stochastic realization theory, robust stabilization, controls, and in areas such as financial mathematics [22]. Let us consider the Riccati DE as given below [5,17] :

$$\begin{aligned} &\frac{d^v}{dt^v}u(t) = -u^2(t) + 1; \quad t \in (0,1) \\ &u(0) = 0. \end{aligned} \tag{5.7}$$

The exact solution of (5.7) for $v = 1$ can be written as

$$u(t) = \frac{e^{2t} - 1}{e^{2t} + 1}. \tag{5.8}$$

The trial solution of the(5.7) in terms of ANN can be written as

$$u_T(t, w) = t\,(N(t, w) + N_0 - tN_0') \tag{5.9}$$

which satisfies the desired initial condition given in(5.7). The trial solution presented in (5.9) includes the term $N_0 = N(0, w)$ and $N_0' = \frac{\partial N}{\partial t}\big|_{t=0}$, which represent the output of ANN evaluated at $t = 0$ and the derivative of ANN output at $t = 0$ respectively. The calculated MAE in the ANN solution for (5.7) is presented in Figure 6. From Figure 6, it can be observed that $h = 20$ number of hidden nodes provides minimum STD in the DE error for 100 runs and is therefore chosen as the winning number of hidden nodes for solving (5.7). To demonstrate the accuracy of the ANN method using the HSA, the ANN solution is compared with the available exact solution, as depicted in Figure 5.

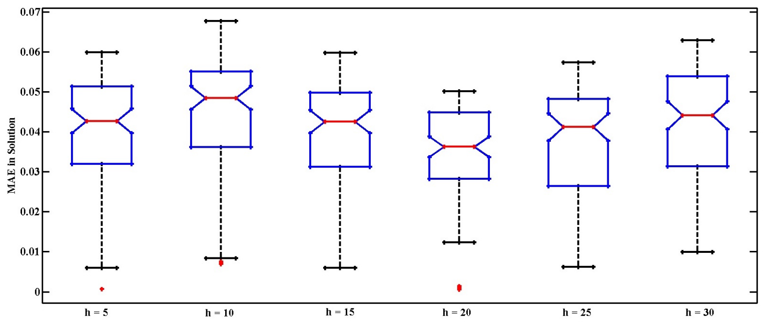

Figure 6. Boxplots for MAE in the solution for 100 runs for solving test problem 3

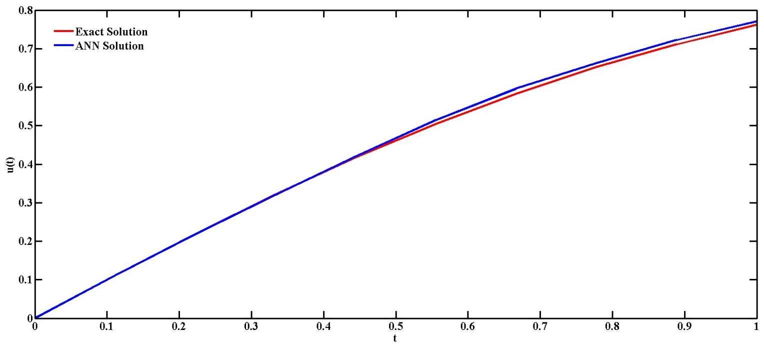

Figure 7. Exact and ANN solutions for test problem 3

5.4. Test Problem 4

In this example, we consider Bratu's initial value problem, also known as "Liouville–Gelfand or Liouville–Gelfand–Bratu problem in honor of Gelfand and the nineteenth-century work of the great French mathematician Liouville" [2]. Here we consider Bratu's initial value problem of the following type [25]:

$$\begin{aligned} &\frac{d^2u}{dx^2} - 2e^u = 0; t \in (0,1) \\ &u(0) = u'(0) = 0. \end{aligned} \tag{5.10}$$

The exact solution of Bratu's initial value problem can be given as

$$u(x) = -2\ln(\cos(x)). \tag{5.11}$$

The trial solution of (5.10) can be written as

$$u_T(x,w) = x^2\left(N + x\left(N_0 + N_0'\right)\right) \tag{5.12}$$

which satisfies the desired initial conditions defined in (5.10). The terms $N_0 = N(0,p)$ and $N_0' = \frac{\partial N}{\partial x}\big|_{x=0}$ in (5.12) represent the ANN output and its derivative evaluated at $x = 0$ respectively. From Figure 8. , it can be observed that $h = 25$ number of hidden nodes is providing the solution with the lowest STD in DE error. The ANN solution is also compared with the exact solution to check the accuracy of the method over the domain, as shown in Figure 9.

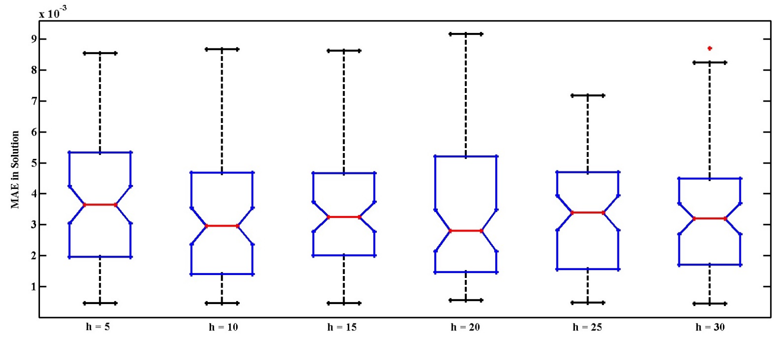

Figure 8. Boxplots for MAE in the solution for 100 runs for solving test problem 4

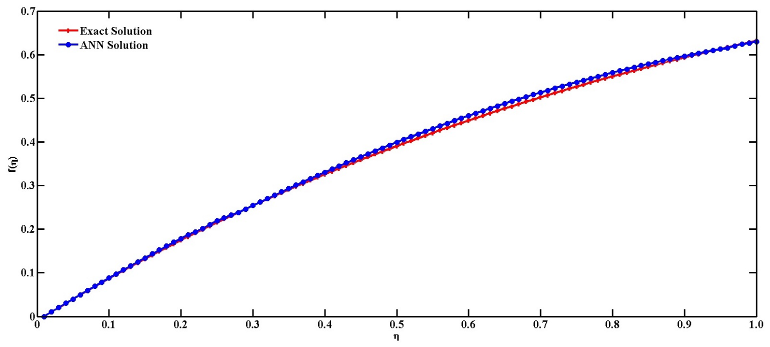

Figure 9. Exact and ANN solutions for test problem 4

5.5. Test Problem 5

An example of PDEs is considered as our next test problem to check the applicability of the method to PDEs as well. We considered an advection diffusion equation (ADE) used to describe many physical situations including heat transfer to a draining film , mass transfer, flow in porous media, movement of contaminants in subsurface, and water transport in soils [11]. The unsteady one-dimensional ADE can be written as

$$\frac{\partial u}{\partial t} + \frac{\partial u}{\partial x} = \frac{1}{\mathrm{Re}} \frac{\partial^2 u}{\partial x^2}, \quad 0 < x < 1,\ t > 0 \tag{5.13}$$

together with the initial and BCs defined as

$$\begin{cases} u(x,0) = 0, 0 < x < 1 \\ u(0,t) = 0, u(1,t) = 1, t > 0. \end{cases} \tag{5.14}$$

The analytical solution of (5.13) can be obtained by the method of separation of variables and given as

$$u(x,t) = \left[\frac{\exp(\mathrm{Re} \times x) - 1}{\exp(\mathrm{Re}) - 1}\right] + \sum_{m=1}^{\infty} \left\{ \frac{(-1)^m m\pi}{(m\pi)^2 + \frac{\mathrm{Re}^2}{4}} \exp\left(\frac{\mathrm{Re} \times (x-1)}{2}\right) \times \sin(m\pi x) \exp\left[-t\left(\frac{(m\pi)^2}{\mathrm{Re}} + \frac{\mathrm{Re}}{4}\right)\right] \right\}. \tag{5.15}$$

The problem defined in (5.13) has steep boundaries near $x = 1$ and many numerical methods provide nonphysical oscillation near steep boundaries. The trial solution for (5.13) can be written as

$$u_T(x,t,w) = \frac{xt}{\sqrt{\mathrm{Re}^2(x-1)^2 + t^2}} + \left(\frac{1}{\mathrm{Re}}\right) x(x-1)tN(x,t,w) \tag{5.16}$$

which satisfies the desired BCs in (5.14). The MAE in the solution was calculated for all considered hidden node number values and 100 random initial starting weight, and is shown in Figure 10. Among all the hidden node combinations of the ANN, $h = 10$ is chosen as the best performing hidden node having minimum STD in the DE error. The comparison of ANN and exact solutions is presented in Figure 11.

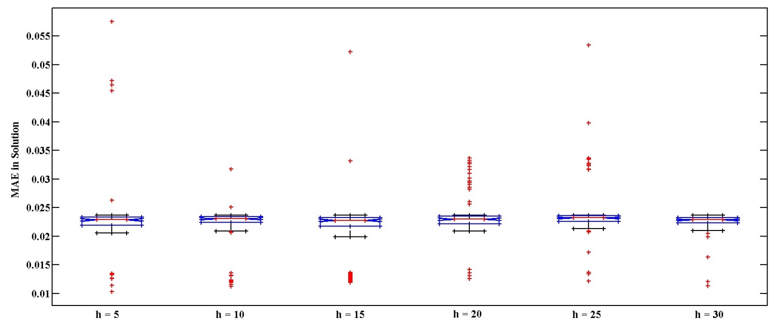

Figure 10. Boxplots for MAE in the solution for 100 runs for solving test problem 5

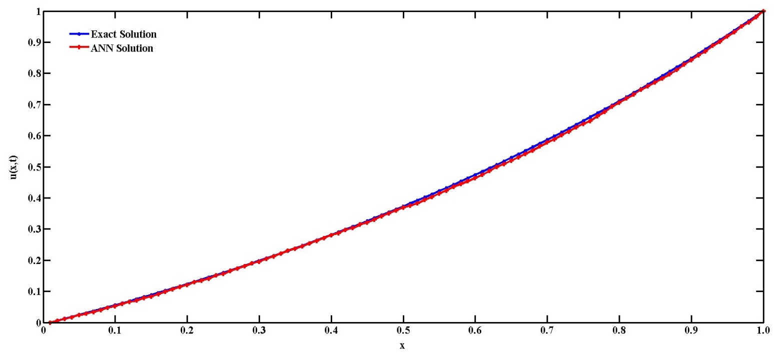

Figure 11. Exact and ANN solutions for test problem 5

6. Optimization results and discussions

In this section, the detailed analysis of statistical results obtained in optimizing ANN parameters using the HSA is presented. Neural network parameters were optimized using 100 independent runs for each of the test problems. A MATLAB code was developed for training ANN parameters using the HSA, and all the test problems were run on Intel(R) Core(TM) i7-3770 CPU @ 3.40 GHz with 8 GB RAM. The chosen values for user parameters of the HSA for solving all test problems are listed in Table 2.

Table 2. Parameter setting for the HSA

Parameters	Numbers
Maximum Iteration	1000
Harmony Memory Size (HMS)	20
Harmony Memory Consideration Rate (HMCR)	0.9
Pitch Adjustment Rate (PAR)	0.05
Bounds of decision variables	Initial or boundary conditions
Bandwidth (BW)	Range of initial or boundary points

Statistical optimization results for the error functions (EFs) of all the test problems using the HSA are given in Tables 3-4. In Table 3, we choose the two best representative hidden nodes of the ANN model for each test problem to demonstrate the statistical optimization results. Statistical results for the MAE in the solution of all the test problems are tabulated in Table 4, in which we present the worst(maximum), best(minimum), average, and standard deviation (SD) of the MAE in 100 independent runs with only the chosen winning number of hidden nodes for all the test problems. Test problems 1a and 1b in all the Tables 3-7 indicate test problem1 (Troesch's problem) for $\mu = 0.5$ and 1, respectively. A brief description of the minimization of EFs over the number of iterations is shown in Figure 12 for all the test problems, considering the chosen winning number of neurons. The running time analysis is also presented for all the test problems in Table 5; we consider the running time as the average computation time in 100 independent runs for all the test problems.

Table 3. Statistical optimization results for all test problems using the HSA

Test Problems	Number of Hidden Nodes	Worst EF	Average EF	Best EF	SD
$1a$	$h = 10$	$8.96E-04$	1.31E − 04	8.05E − 06	2.04E − 04
	$h = 20$	1.55E − 03	1.12E − 04	7.18E − 06	1.96E − 04
$1b$	$h = 15$	2.11E − 03	7.68E − 04	9.40E − 05	3.48E − 04
	$h = 20$	1.42E − 03	7.03E − 04	1.55E − 04	2.81E − 04
2	$h = 5$	2.14E − 03	1.36E − 03	1.20E − 03	1.47E − 04
	$h = 10$	2.51E − 03	1.38E − 03	1.22E − 03	2.00E − 04
3	$h = 20$	4.69E − 04	2.62E − 04	8.39E − 05	8.65E − 05
	$h = 25$	4.79E − 04	2.72E − 04	9.96E − 05	8.18E − 05
4	$h = 25$	1.38E − 02	1.06E − 02	7.88E − 03	2.09E − 03
	$h = 30$	1.42E − 02	1.02E − 02	7.83E − 03	2.14E − 03
5	$h = 10$	7.47E − 03	7.34E − 03	7.32E − 03	2.52E − 05
	$h = 20$	7.45E − 03	7.34E − 03	7.32E − 03	2.30E − 05

Table 4. Statistical optimization results for the MAE in the solution of all the test problems

Test Problems	Number of Hidden Nodes	Worst MAE	Best MAE	Average MAE	SD
1a	$h = 10$	3.53E − 03	1.28E − 03	2.33E − 03	4.15E − 03
1 b	$h = 15$	1.15E − 02	8.36E − 03	1.00E − 02	4.56E − 04
2	$h = 5$	3.66E − 02	5.96E − 03	1.29E − 02	4.60E − 03
3	$h = 20$	5.00E − 02	6.00E − 04	3.43E − 02	1.20E − 02
4	$h = 25$	7.17E − 03	4.73E − 04	3.26E − 03	1.76E − 03
5	$h = 10$	3.17E − 02	1.12E − 02	2.20E − 02	3.26E − 03

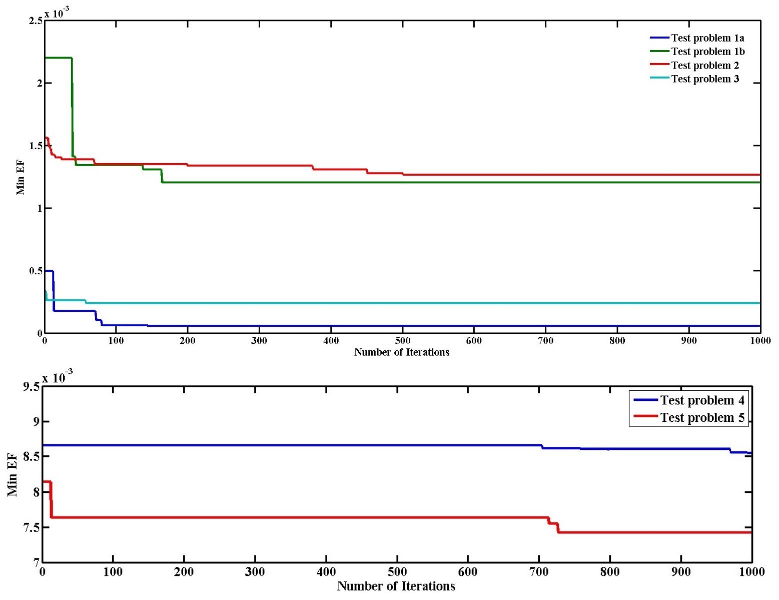

Figure 12. Summary of minimization of EFs over the number of iterations (a) for test problems 1–3 and (b) for test problems 4,5.

From Table 5, it can be seen that increasing the number of hidden nodes or training points inside the domain increases the computing time. Besides, the running time is problem-dependent and is based on the complexity of the problem and the constructed trial approximate solution. For the test problems considered in this work, we observe that on an average, the computing time for the six test problems using $n = 10$ (number of training points) and $h = 5, 10, 15, 20, 25, 30$ (number of hidden nodes) is 0.42, 0.62, 0.82, 1.03, 1.32, and 1.44 seconds respectively.

In order to prove the efficiency and correctness of the algorithm, the solution obtained using the ANN-HSA method is also compared with the solutions obtained by some traditional numerical methods in existing literature. The comparison of solutions obtained by the ANN-HSA method and other numerical methods is presented in Table 6 and 7 for all the test problems. ANN-HSA solution of Test Problem 1(a) and (b) is compared with the Adomain decomposition method (ADM) [6], Homotopy perturbation method (HPM) [4] and Homotopy analysis method (HAM) [9]. Numerical solution obtained using ANN-HSA for test problem 3 is compared with solution obtained using variational iteration method (VIM) , Modified homotopy perturbation method (MHPM) and ADM results tabulated in [5, 22] and refrences

Table 5. Average computational time for solving the six test problems using the ANN-HSA method

Test - problem	Initial /boundary condition	No of - training points	Average running time for 100 independent runs					
			$h=5$	$h=10$	$h=15$	$h=20$	$h=25$	$h=30$
1(a)	$u(0)=0, u(1)=1$	10	0.4062	0.5988	0.7836	1.019	1.707	1.360
1(b)	$u(0)=0, u(1)=1$	10	0.4086	0.5988	0.7920	0.9847	1.185	1.359
2	$u(0)=0, u(1)=1$	10	0.4382	0.6554	0.8857	1.071	1.282	1.496
3	$u(0)=0$	10	0.4438	0.6658	0.8757	1.086	1.307	1.510
4	$u(0)=u'(0)=0$	10	0.4401	0.6518	0.8698	1.082	1.326	1.574
5	$u(x,0)=0, 0<x<1$ $u(0,t)=0$ $u(1,t)=1, t>0$	10	0.3625	0.5564	0.7547	0.9554	1.153	1.360

therein. Following that for test problem 4 results are compared with Runge-Kutta method (RKM), optimal homotopy asymptotic method (OHAM) and Bezier curve method (BCR) [3, 8] and references therein. Results for test problem 5 is compared with the results of Cranck-Nicolson method, High order exponentiat scheme [HOE] and method based on difference scheme (DS) tabulated in [27] and references therein. From Table 7 (a) and (b), we infer that the solution obtained using the ANN-HSA method is approximate and comparable to the exact solution as well as the solutions obtained by other numerical methods.

Table 6. Absolute error in the solutions obtained by the ANN-HSA method and other numerical methods at some domain points for Test Problem 1(a), 1(b) and 2

Training - Points	Test problem 1a				Test problem1b				Test problem 2
	ANN-HSA	ADM [60]	HPM	HAM	ANN-HSA	ADM	HPM	HAM	ANN-HSA
0.1	$2.93E-03$	$5.99E-06$	$4.78E-06$	$2.69E-07$	$6.17E-03$	$4.12E-04$	$2.79E-04$	$1.201E-05$	$1.00E-03$
0.2	$1.30E-04$	$1.06E-05$	$9.42E-06$	$5.36E-07$	$2.69E-04$	$7.40E-04$	$5.50E-04$	$2.41E-05$	$3.9E-02$
0.3	$3.5E-04$	$1.40E-05$	$1.37E-05$	$8.13E-07$	$3.46E-03$	$9.79E-04$	$8.00E-04$	$3.633E-05$	$1E-03$
0.4	$1.08E-03$	$1.61E-05$	$1.73E-05$	$1.08E-06$	$7.26E-03$	$1.13E-03$	$1.01E-03$	$4.874E-05$	$5.0E-03$
0.5	$2.18E-03$	$1.68E-05$	$1.97E-05$	$1.34E-06$	$1.01E-02$	$1.197E-03$	$1.15E-03$	$6.118E-05$	$3.30E-03$
0.6	$5.3E-03$	$1.62E-05$	$2.04E-05$	$1.57E-06$	$0.10E-02$	$1.16E-03$	$1.20E-03$	$7.285E-05$	$4.50E-03$
0.7	$3.47E-03$	$1.43E-05$	$1.88E-05$	$1.71E-06$	$0.62E-03$	$1.044E-03$	$1.11E-03$	0.1105613	$4.40E-03$
0.8	$1.64E-03$	$1.09E-05$	$1.46E-05$	$1.65E-06$	$0.16E-01$	$8.20E-03$	$8.74E-04$	$8.186E-05$	$1.0E-03$
0.9	$2.73E-03$	$6.19E-06$	$7.99E-06$	$1.18E-06$	$1.53E-03$	$4.53E-04$	$4.78E-04$	$6.239E-05$	$2.20E-03$

Table 7. Absolute error in the solutions obtained by the ANN-HSA method and other numerical methods at some domain points for Test Problem 3, 4 and 5

Training points	Test problem 3				Test problem 4				Test problem 5			
	ANN -HSA	VIM	MHPM	ADM	ANN -HSA	RKM	BCM	OHAM	ANN -HSA	Cranck-Nicolson	DIM	HOE
0.1	$2.44E-04$	0	$1.00E-06$	$9.94E-07$	$6.32E-04$	$1.66E-05$	$2.98E-04$	$6.41E-07$	$2.08E-03$	$5.50E-01$	$3E-06$	$3E-06$
0.2	$7.35E-04$	0	0	$3.22E-07$	$1.56E-03$	$3.1E-07$	0.0	$9.74E-06$	$4.34E-03$	$5.7E-05$	$5E-06$	$5E-06$
0.3	$7.13E-04$	0	0	$6.20E-07$	$1.54E-03$	$1.13E-06$	$1.69E-04$	$4.52E-05$	$1.25E-03$	$8.1E-05$	$7E-06$	$7E-06$
0.4	$2.37E-03$	$2E-06$	$4E-06$	$9.80E-07$	$1.92E-04$	$2.12E-06$	$1.10E-04$	$1.27E-04$	$2.94E-03$	$9.9E-05$	$9E-06$	$9E-06$
0.5	$1.72E-02$	$1.45E-05$	$3.90E-05$	$1.88E-07$	$2.81E-03$	$2.9E-06$	0.0	$2.68E-04$	$6.84E-03$	$1.09E-04$	$1E-05$	$1E-05$
0.6	$1.30E-02$	$6.6E-05$	$1.92E-04$	$6.12E-07$	$6.59E-03$	$4.1E-06$	0.0	$4.83E-04$	$1.05E-02$	$1.12E-04$	$9E-06$	$9E-06$
0.7	$1.67E-02$	$2.43E-04$	$7.36E-04$	$8.41E-07$	$9.86E-03$	$4.5E-06$	$7.77E-04$	$8.36E-04$	$1.10E-03$	$1.05E-04$	$9E-06$	$9E-06$
0.8	$1.61E-02$	$7.36E-04$	$2.33E-03$	$1.00E-06$	$6.23E-03$	$3.35E-05$	0.0	$1.60E-03$	$4.55E-02$	$8.5E-05$	$7E-06$	0
0.9	$9.20E-02$	$7.16E-01$	$1.55E-02$	$2.51E-06$	$4.34E-03$	$4.37E-05$	$3.47E-04$	$3.64E-03$	$3.98E-03$	$5E-05$	$3E-06$	$3E-06$

Artificial intelligence algorithms have given numerical researchers new options for solving extremely difficult problems with less computation time and calculus effort. The recent development and advantages of using ANN methods for solving DEs over traditional numerical methods makes this study more significant. In most of the studies conducted in the recent past for solving DEs using ANNs, gradient-based optimization techniques including gradient descent, Levenberg–Marquardt, and steepest descent algorithms have been used. In some researches, nongradient-based metaheuristic algorithms including particle swarm optimization, genetic algorithm, and hybrid algorithms that combine both have also been used to train ANNs for solving DEs.

7. Conclusion

In this paper, we proposed an algorithm based on the length factor ANN method and the HSA for obtaining approximate solutions of both ODEs and PDEs. We considered six test problems having different initial or BCs to check the applicability of our method. Based on the simulation results for all the test problems, the following conclusions can be drawn: (i) The designed trial solution based on the ANN method satisfies initial/boundary conditions exactly using a log-sigmoid activation function. (ii) The computational intelligence algorithm based on ANN-HSA provides reliably approximate solution for all the test problems. (iii) In its current preliminary formulation, the algorithm produces a satisfactory accurate approximation of the solution. The proposed algorithm has been demonstrated to be comparable to other numerical methods and to provide a more precise solution at certain domain points. (iv) The lower values of statistical parameters such as average, SD in MAE, and EF confirm the proposed ANN models' reliability, stability, and effectiveness. (v) The results obtained using computational time show that the proposed algorithm is simple to implement and takes less time to complete. Computational time may vary depending upon the numbers of training points, hidden nodes, and initial weight parameters. As the parameter values increase, the computational time required for calculating the approximate solutions will also increase. In future work, investigation of an improved version of HSA for training network parameters can be performed for solving a wide variety of DEs that have not been solved until now as well as some other methods for construction of trial solution e.g. Legendre polynomial [29, 30] can be used to enhance the accuracy of the solution.

Acknowledgement

We express our sincere thanks to editor in chief, editor and reviewers for their valuable suggestions to revise this manuscript.

References

[1] W. E. Boyce, R. C. DiPrima and D. B. Meade, *Elementary differential equations and boundary value problems*, John Wiley & Sons, 2021.

[2] J. P. Boyd, *One-point pseudospectral collocation for the one-dimensional bratu equation*, Applied Mathematics and Computation, 2011, 217(12), 5553–5565.

[3] M. A. Darwish and B. S. Kashkari, *Numerical solutions of second order initial value problems of bratu-type via optimal homotopy asymptotic method*, American Journal of Computational Mathematics, 2014, 4(2), 47.

[4] E. Deeba, S. Khuri and S. Xie, *An algorithm for solving boundary value problems*, Journal of Computational Physics, 2000, 159(2), 125–138.

[5] M. A. El-Tawil, A. A. Bahnasawi and A. Abdel-Naby, *Solving riccati differential equation using adomian's decomposition method*, Applied Mathematics and Computation, 2004, 157(2), 503–514.

[6] X. Feng, L. Mei and G. He, *An efficient algorithm for solving troesch's problem*, Applied Mathematics and Computation, 2007, 189(1), 500–507.

[7] Z. W. Geem, *Music-inspired harmony search algorithm: theory and applications*, 191, Springer, 2009.

[8] F. Ghomanjani and S. Shateyi, *Numerical solution for fractional bratu's initial value problem*, Open Physics, 2017, 15(1), 1045–1048.

[9] H. N. Hassan and M. A. El-Tawil, *An efficient analytic approach for solving two-point nonlinear boundary value problems by homotopy analysis method*, Mathematical methods in the applied sciences, 2011, 34(8), 977–989.

[10] A. Kattan, R. Abdullah and R. A. Salam, *Harmony search based supervised training of artificial neural networks*, in *2010 International conference on intelligent systems, modelling and simulation*, IEEE, 2010, 105–110.

[11] W. Khan and I. Pop, *Boundary-layer flow of a nanofluid past a stretching sheet*, International journal of heat and mass transfer, 2010, 53(11–12), 2477–2483.

[12] S. Kulluk, L. Ozbakir and A. Baykasoglu, *Training neural networks with harmony search algorithms for classification problems*, Engineering Applications of Artificial Intelligence, 2012, 25(1), 11–19.

[13] M. Kumar and N. Yadav, *Multilayer perceptrons and radial basis function neural network methods for the solution of differential equations: a survey*, Computers & Mathematics with Applications, 2011, 62(10), 3796–3811.

[14] I. E. Lagaris, A. C. Likas and D. G. Papageorgiou, *Neural-network methods for boundary value problems with irregular boundaries*, IEEE Transactions on Neural Networks, 2000, 11(5), 1041–1049.

[15] K. S. McFall, *Automated design parameter selection for neural networks solving coupled partial differential equations with discontinuities*, Journal of the Franklin Institute, 2013, 350(2), 300–317.

[16] K. S. McFall and J. R. Mahan, *Artificial neural network method for solution of boundary value problems with exact satisfaction of arbitrary boundary conditions*, IEEE Transactions on Neural Networks, 2009, 20(8), 1221–1233.

[17] S. Momani and N. Shawagfeh, *Decomposition method for solving fractional riccati differential equations*, Applied Mathematics and Computation, 2006, 182(2), 1083–1092.

[18] L. Özbakır, A. Baykasoğlu and S. Kulluk, *A soft computing-based approach for integrated training and rule extraction from artificial neural networks: Difaconn-miner*, Applied Soft Computing, 2010, 10(1), 304–317.

[19] H. Qu, X. Liu and Z. She, *Neural network method for fractional-order partial differential equations*, Neurocomputing, 2020, 414, 225–237.

[20] M. A. Z. Raja, *Stochastic numerical treatment for solving troesch's problem*, Information Sciences, 2014, 279, 860–873.

[21] M. A. Z. Raja, J. Khan and I. Qureshi, *Swarm intelligent optimized neural networks for solving fractional differential equations*, International Journal of Innovative Computing, Information and Control, 2011, 7(11), 6301–6318.

[22] M. A. Z. Raja, J. A. Khan and I. M. Qureshi, *A new stochastic approach for solution of riccati differential equation of fractional order*, Annals of Mathematics and Artificial Intelligence, 2010, 60(3), 229–250.

[23] S. Roberts and J. Shipman, *On the closed form solution of troesch's problem*, Journal of Computational Physics, 1976, 21(3), 291–304.

[24] B. Troesch, *A simple approach to a sensitive two-point boundary value problem*, Journal of Computational Physics, 1976, 21(3), 279–290.

[25] A. M. Wazwaz, *Adomian decomposition method for a reliable treatment of the bratu-type equations*, Applied Mathematics and Computation, 2005, 166(3), 652–663.

[26] N. Yadav, T. T. Ngo, A. Yadav and J. H. Kim, *Numerical solution of boundary value problems using artificial neural networks and harmony search*, in International Conference on Harmony Search Algorithm, Springer, 2017, 112–118.

[27] N. Yadav, A. Yadav and J. H. Kim, *Numerical solution of unsteady advection dispersion equation arising in contaminant transport through porous media using neural networks*, Computers & Mathematics with Applications, 2016, 72(4), 1021–1030.

[28] N. Yadav, A. Yadav, M. Kumar and J. H. Kim, *An efficient algorithm based on artificial neural networks and particle swarm optimization for solution of nonlinear troesch's problem*, Neural Computing and Applications, 2017, 28(1), 171–178.

[29] Y. Yang, M. Hou and J. Luo, *A novel improved extreme learning machine algorithm in solving ordinary differential equations by legendre neural network methods*, Advances in Difference Equations, 2018, 2018(1), 1–24.

[30] Y. Yang, M. Hou, H. Sun et al., *Neural network algorithm based on legendre improved extreme learning machine for solving elliptic partial differential equations*, Soft Computing, 2020, 24(2), 1083–1096.