Stochastic Modelling and Computational Sciences

TIME AND ERROR ANALYSIS OF NEURAL NETWORK MODEL OF RECTANGULAR VIBRATING MEMBRANE

Vikas Kumar Pandey[1*] and Himanshu Agarwal[2]
[1,2]Department of Mathematics, Jaypee Institute of Information Technology, A-10, Sector 62, Noida, 201309, Uttar Pradesh, India
vikasmathematics90@gmail.com[1*] and [2]himanshu.agarwal@jiit.ac.in

ABSTRACT
The transverse motion $(u = u(x, y, t))$ of a rectangular membrane with fixed edges under the influence of a transverse driving force $A \sin \omega t$ attached at a position (x_0, y_0) is governed by the differential equation (DE) $\frac{\partial^2 u}{\partial t^2} = \frac{1}{c^2}\Delta u + \frac{A}{m}\delta(x - x_0)\delta(y - y_0)\sin \omega t$, where c, m determine membrane material qualities, A and ω are force constants, $\delta(.)$ is the impulse function, the space coordinates are (x, y), and time variable is t. This paper identifies the architecture and weights of artificial neural network (ANN) for the series solution representation of DE model of the rectangular vibrating membrane. Seventy-seven different ANN models in terms of architecture, and data-set samples have been investigated. Root mean squared error (RMSE) has been used for the selection of the best ANN model. The RMSE value of the best ANN model is 0.0009, which is ten times better than the error of state-of-the-art. For solution prediction, ANN models perform 26 times more quickly than series solution method.

Keywords: Vibrating Membrane, Artificial Neural Network, Training Time, Weight and Bias.

1 INTRODUCTION
A membrane [16] is a thin film structure similar to drum heads [32], flags, trampolines [12], soap films [3], biological and nitrocellulose membranes [1]. It is a very important component in architectural and civil structures; switch and transducer diaphragms; biomedical prostheses (like artificial kidneys, arteries, blood oxygenators, and organs); and other uses in space (such as optical reflectors and radio antennas) [19]. Membrane technology [1] is the basis of industrial processes like electrodialysis, reverse osmosis, ultrafiltration, microfiltration, gas separation, and pervaporation. The pharmaceutical industry uses it to increase the effectiveness and safety of drug delivery [1].

Transport and vibration are two significant membrane phenomena. Under suitable assumptions, vibration is governed by the wave equation [14], while transport is described by Fick's law and irreversible thermodynamics [1].

The equation of membrane [16] is governed by

$$\frac{\partial^2 u(x,y,t,\omega)}{\partial t^2} = \frac{1}{c^2}\left(\frac{\partial^2 u(x,y,t;\omega)}{\partial x^2} + \frac{\partial^2 u(x,y,t;\omega)}{\partial y^2}\right) + \frac{A}{m}\delta(x - x_0)\delta(y - y_0)\sin \omega t \quad (1)$$

where A, c are constant, x, y are space variable, t is time variable, ω is external frequency, m is the mass per unit area of the membrane, and

$$\delta(x - x_0) = \begin{cases} \frac{1}{\Delta x}, & \text{if } x_0 - \frac{\Delta x}{2} < x < x_0 + \frac{\Delta x}{2} \\ 0, & \text{otherwise} \end{cases},$$

$$\delta(y - y_0) = \begin{cases} \frac{1}{\Delta y}, & \text{if } y_0 - \frac{\Delta y}{2} < y < y_0 + \frac{\Delta y}{2} \\ 0, & \text{otherwise} \end{cases}$$

under the initial condition

$$u(x, y, 0) = 0, \quad \frac{\partial u}{\partial t}(x, y, 0) = 0 \quad (2)$$

and the Dirichlet boundary condition

$$u(0,y,t) = u(l_x,y,t) = u(x,0,t) = u(x,l_y,t) = 0. \quad (3)$$

Series solution of (1) using separation of variables method and superposition principle (when $\omega \neq \omega_{l,m}$) [8, 32] is

$$u(x,y,t;\omega) = \frac{4A}{m}\sum_{l,m=1}^{\infty}\frac{1}{\omega_{l,m}^2-\omega^2}\left[\sin\omega t - \frac{\omega}{\omega_{l,m}}\sin\omega_{l,m}\,t\right]\sin l\pi x_0 \sin m\pi y_0 \ \sin l\pi x \sin m\pi y\,. \quad (4)$$

where natural frequency $\omega_{l,m}$ is defined as

$$\omega_{l,m} = c\pi\sqrt{\frac{l^2}{l_x^2}+\frac{m^2}{l_y^2}}. \quad (5)$$

Due to a computational constraint, the solution u is roughly represented as $u'(x,y,t;\omega)$ by following equation

$$u(x,y,t;\omega) \approx u(x,y,t;\omega) = \frac{4A}{m}\sum_{l,m=1}^{L,M}\frac{1}{\omega_{l,m}^2-\omega^2}\left[\sin\omega t - \frac{\omega}{\omega_{l,m}}\sin\omega_{l,m}\,t\right]\sin l\pi x_0 \sin m\pi y_0 \ \sin l\pi x \sin m\pi y. \quad (6)$$

We use a variety of applications of the wave phenomenon [20, 29, 32] in our daily lives, including information communication, electricity generated by solar panels, household appliances that employ microwaves, X-rays used to view the organs of the body, and γ rays used to find flaws in metalworking.

Ultrasonic waves are used by sonar to find and detect items underwater, the inside of the human body can be visualized by medical professionals using sonograms, and infrared sound is used to control the flow. Both tsunami waves and infrasonic waves, which resemble pond ripples or the sound of a guitar string, are employed to track earthquakes. Seismic waves are measured by synthetic aperture radar [29].

Artificial Neural Network (ANN) is a modern statistical model for solving regression and classification-related problems [9]. The success of ANN has attracted researchers for solving the differential equation (DE) using ANN [4]. Lee et al. (1990) [22] pioneered the use of ANN to investigate the solution of first-order ordinary differential equations (ODEs). Henseler and Braspenning (1992) [17] presented a model of a cellular neural network (CNN) based on a vibrating membrane. Szology et al. (1993) [33] studied the CNN paradigm in mechanical vibrating systems. Afterward, many researchers worked on ANN for the study of the DE, and some important works are given as follows: Mall and Chakraverty (2013) [24] have worked on the solution of an ordinary differential equation (ODE) using ANN. Jafarian and Baleanu (2017) [18] have solved wave and heat equations using a combination of ANN and power series methods. Weinan et al. (2017) [11] have used deep learning-based numerical algorithms to solve high dimensions parabolic PDEs and backward stochastic differential equations (SDEs). Raissi et al. (2019) [23] solved non-linear partial differential equations (PDEs), including Burger's equation, the Schrödinger equation, and the Allen-Cahn equation, using PINN. Dwivedi and Srinivasan (2020) [10] discussed Physics informed extreme learning machines (PIELM). Cai et al. (2020) [7] discussed a phase shift in the deep ANN to solve the wave equation. Samaniego et al. (2020) [31] discussed the solution of mechanics-related PDEs using the ANN paradigm. The approximation space has been constructed using ANN to solve a number of computational mechanics applications. Moseley and Markham (2020) [2] created physics-informed neural networks (PINNs), an expanded form of ANN, and used PINNs to solve the wave equation. Ren et al. (2020) [26] proposed the seismic waveform inversion method using ANN. Xio and Frank (2021) [34] solved the Boltzmann equation using ANN. Zaho et al. (2021) [36] used biological ANN to discuss particle dispersion on a vibrating screen surface. Blechschmidt and Oliver (2021) [4] described three methods for solving PDEs with ANN, including PINNs, the Feynmann-Kac formula, and the solution of backward SDEs. Brink et al. (2021) [6] solved PDE using the ANN collocation method. Yeung et al. (2022) [35] developed a deep-learning system to analyze highly oscillatory wave patterns and extract ray directions at

specific spots. Pandey et al. [27] have provided a ANN model of a wave equation. The model's effectiveness and accuracy have been evaluated.

In this paper, ANN has been used to model the vibration of a rectangular membrane of the equation (6) for different values of the force constant ω. For various sample sizes and hidden layers, numerous experiments were undertaken to determine the minimal root mean square error (RMSE) and training time (TT). The RMSE value of the best ANN model is 0.0009, which is ten times better than the error of state-of-the-art. The developed ANN model's average TT is faster than the state-of-the-art for solving differential equations using ANN. For solution prediction, ANN models perform 26 times more quickly than series solution of a rectangular vibrating membrane (SSRVM). In Section 2, the data set format and the algorithm of ANN are presented. Section 3 provides an explanation of performance evaluation strategies for accuracy and time cost. Section 4 provides specifics on the experiments, results, analyses, and comparisons. Section 5 then provides conclusions.

2 Artificial Neural Network: The number and arrangement of perceptrons [30] determine the structure of ANN. A perceptron [30] consists of weights, bias, and a transfer function. It takes R- dimensional input of real numbers and returns a real number, which is the transfer function value of a weighted sum of the input numbers plus bias. The shorthand notation for representing the structure of a multilayer feed-forward neural network (MFFNN) [15] with H minus one hidden layer(s) is $R' - s^1 - s^2 - \cdots s^H - o$, where R' is the number of inputs, $s^1 - s^2 - \cdots s^H$ are the number of perceptron's in the first (input) layer, second (first hidden) layer, … H (H minus one hidden) layer respectively and o is the number of output. The number of input and output variables in equation (6) is four and one respectively. Therefore, $R' = 4$ and $o = 1$. The order of weight matrix $\boldsymbol{W^1}$ corresponding to the first layer is $s^1 \times 4$, and the dimension of the bias vector $\boldsymbol{b^1}$ is $s^1 \times 1$ for first layer. Similarly, weight and bias matrices $\boldsymbol{W^2}, \boldsymbol{W^3}, \ldots$ and $\boldsymbol{b^1}, \boldsymbol{b^2}, \boldsymbol{b^3}, \ldots$ are associated with other layers.

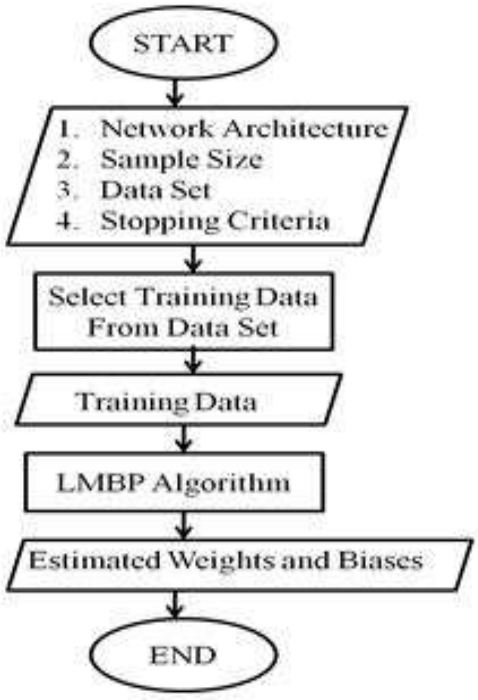


Fig. 1 Flow diagram of ANN

ANN design is the estimation of parameters (weight and bias matrices) associated with each layer. It depends on training data and the choice of hyperparameters (number and arrangement of perceptrons, transfer function, loss function, and numerical technique for computing the minimum of the loss function).

ANN works on discrete data. The discrete data of the approximation $u'(x,y,t;\omega)$ (6) is computed at discrete points $u'(x^i,y^i,t^i;\omega^i), i = 1,2,\dots N'_x \times N'_y \times N'_t \times N'_\omega (= N')$, where N' is the total size of data and $\frac{1}{N'_x}, \frac{1}{N'_y}, \frac{1}{N'_t}, \frac{1}{N'_\omega}$ are the uniform sampling rate (number of points in the unit interval) of x, y, t, ω variables respectively. The computed value of the output variable $u'(x,y,t;\omega)$ at the point of $(x^i,y^i,t^i;\omega^i)$ is represented by T^i (say target value). The format of the data is provided in table 1. Training data set $\{D = (x^i,y^i,t^i;\omega^i,T^i), i = 1,2,\dots Q\}$ is a subset of Table 1.

Table 1 Data set format. $G = \{(x^i,y^i,t^i;\omega^i): i = 1,2,\dots N'\}$.

S.N.	Input Variables	Target Variables
1	x^1,y^1,t^1,ω^1	$u'(x^1,y^1,t^1,\omega^1) = T^1$
2	x^2,y^2,t^2,ω^2	$u'(x^2,y^2,t^2,\omega^2) = T^2$
3	x^3,y^3,t^3,ω^3	$u'(x^3,y^3,t^3,\omega^3) = T^3$
.		
.		
.		
N'	$x^{N'},y^{N'},t^{N'},\omega^{N'}$	$u'(x^{N'},y^{N'},t^{N'},\omega^{N'}) = T^{N'}$

Algorithm 1 Artificial Neural Network Design for u' given in equation (6)

Input: 1. Hyperparameters of ANN: structure, initial weights, transfer function, loss function, optimization technique.

2. Dataset G.

3. Size of the training dataset (Q).

Stage 1: Select randomly without replacement $D \subset G$ of size Q.

Stage 2: Train ANN using hyperparameters Q and D.

Stage 3: Store weights and bias of trained ANN.

Stage 4: Obtain output values (O^i) for each input data entry of G, $i = 1, 2, \dots N'$.

Stage 5: Obtain $e^i = O^i - T^i$ for error analysis.

Output: 1. Weights and biases of trained ANN.

2. Errors e^i, $i = 1, 2, \dots N'$.

It can be difficult to choose an appropriate transfer function. The best option for regression is the hyperbolic tangent sigmoid (HTSF), which is centered at zero [27].

For the training data set $\{D = (x^i,y^i,t^i;\omega^i,T^i), i = 1,2,\dots Q\}$, loss function is *(LF)*. The following equation (7) yields the *LF*.

$$LF(W) = \sum_{i=1}^{Q} [O^i - T^i]^2, \quad (7)$$

where O^i is ANN output for input data corresponding to i.

For the purpose of minimizing *LF*, a variety of optimization techniques are available, including steepest descent algorithm [28], Gauss-Newton algorithm [5], and the Levenberg Marquardt Backpropagation

Algorithm (LMBP) [15]. The LMBP method, which leads to backpropagation ANN, is the most effective way to minimize the loss function. In order to train the ANN, LMBP has a control parameter called μ_k [15]. When μ_k is raised, the algorithm approaches the steepest descent algorithm with a low learning rate, and when μ_k is lowered to zero, the algorithm becomes Gauss-Newton [15]. A summary of the procedures needed for weight and bias estimation is shown in Figure (1). The squared error loss function is minimized in the estimation process. Algorithm 1 provides step-wise details for designing ANN for the solution of equation (1) given as u' in equation (6).

3 **Artificial Neural Network Performance Evaluation:** For the purpose of choosing a decent network architecture, performance is measured in terms of calculation time and accuracy [9]. The computation time is discussed in terms of TT and prediction time (PT). For a discussion of accuracy, the RMSE, peak signal to noise ratio (PSNR), and error histogram have been employed. In relation to random factors (starting weights and biases, as well as the choice of data), statistical measures [25] including average, standard deviation, boxplots of TT, PT, converged weights and biases and *RMSE* has been explored.

The term 'TT' refers to the amount of time needed to estimate a ANN's weights for a specific training data set. The execution time of the LMBP is given in this work as TT.

The amount of time required for a trained ANN to predict the value of the unknown variable u for a given point (x, y, t, ω) is referred to as PT. The value of an unknown at point (x, y, t, ω) is calculated for other NMs.

RMSE at $\{(x^i, y^i, t^i; \omega^i): i = 1,2, \dots N'\}$, is defined as follows:

$$RMSE = \sqrt{\frac{\sum_{i=1}^{N''}(u'(x^i,y^i,t^i,\omega^i)-O^i)^2}{N''}} \quad (8)$$

where the approximation made at point $(x^i, y^i, t^i; \omega^i)$, using ANN or numerical approaches is O^i, approximation of u determined by equation (6) is $u'(x^i, y^i, t^i; \omega^i)$, and data set size is N''.

One well-liked method for displaying scientific data is the error histogram [21]. It is calculated by subtracting u' from O^i for the supplied data set $\{(x^i, y^i, t^i; \omega^i): i = 1,2, \dots N'\}$.

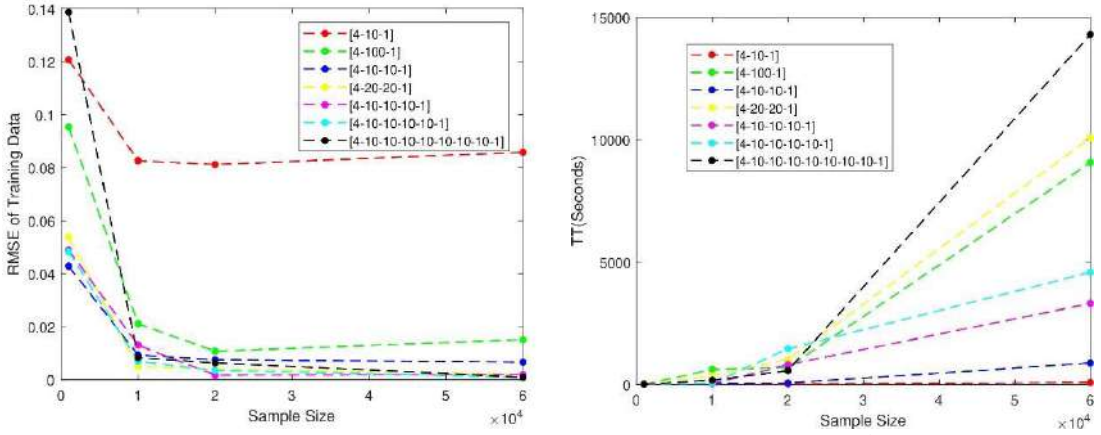
A well-liked quantitative method for comparing images based on human eyes' properties is the PSNR [13]. PSNR at point $\{(x^i, y^i, t, \omega): i = 1,2, \dots N'''\}$ defined as follows:

$$PSNR = 10 \log_{10} \frac{N''' \times (MAX)^2}{\sum_{i=1}^{N'''}(u'(x^i,y^i,t,\omega)-O^i)^2} \quad (9)$$

$$MAX = \max\{\max_i |u'|, \max_i |O^i|\}$$

Where $|u'|, |O^i|$ indicate the absolute value of u' and O^i at (x^i, y^i, t, ω) respectively.

3.1 Statistical Measures: Statistical methods [25], including box plots, averages, and standard deviations (SD), have been used to examine performance measures like RMSE, TT, and PT as well as converged weights/biases in relation to random factors (selection of data, initial weights/biases). The box plot [25] describes the median, 25 percentile, 75 percentile, and outlier of the data set. For the investigation of RMSE and TT, a boxplot is employed. To gather statistical data, images of the average and SD of converged weights and biases were employed.



(a) *(b)*

Fig. 2 (a) Graph showing the relationship between sample size and RMSE for various structures, (b) Graph showing the relationship between sample size and training time for various structures.

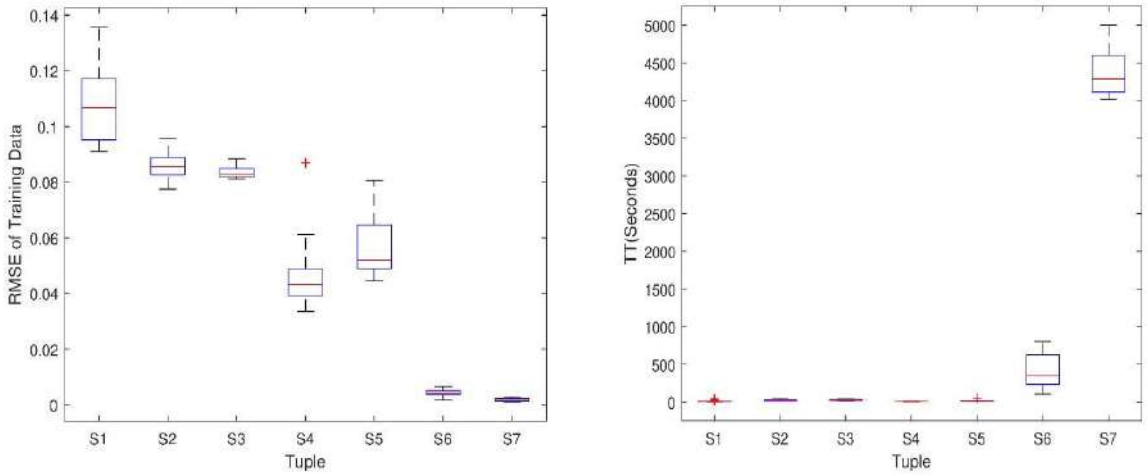
4 Experiments, Result and Analysis: The series solution (6) of the vibrating membrane equation (1) is represented by ANNs. It is assumed that the parameter values are $m = 1,\ x_0 = y_0 = 0.5,\ l_x = l_y = 1$ (equation (5)), $A = 1$ (equation (6)), $L = M = 15,\ c = 0.1$ (equation (1)). All of set

$G = \{(x, y, t, \omega): x \in \{0.02, 0.04, \ldots 1\}, y \in \{0.02, 0.04, \ldots 1\}, t \in \{0.1, 0.2, \ldots 1\}, \omega \in \{0.1, 0.2, \ldots 1\}$ points define the whole data set. Consequently, the whole data set size is 250000. Data sets of the following sizes have been taken into consideration for training: 1000, 10000, 20000, 60000. Taken into account here are ANNs with up to 7 hidden layers and a variety of structures, like $[4-10-1]$, $[4-100-1]$, $[4-10-10-1]$, $[4-20-20-1]$, $[4-10-10-10-1]$, $[4-10-10-10-10-1]$, $[4-10-10-10-10-10-10-1]$ have been examined. As a transfer function, the hyperbolic tangent sigmoid function $\frac{e^z - e^{-z}}{e^z + e^{-z}}$ (where z is real variable) has been applied to all of the structures. Control parameter μ_k is $(0.001,\ 10^{10})$, Maximum iteration variable (k) is 1000, minimum performance gradient is $\frac{1}{10^7}$.

4.1 Analysis of Prediction Error and TT Considering Sample Sizes and ANN Structures: Figures (2a) and (2b) show an analysis of RMSE and training time relative to an ANN structure's training data sample sizes.

Here are a few important observations:

1. In relation to the number of perceptrons in the hidden layers, RMSE declines and TT rises.
2. Relative to the size of the training data, RMSE reduces and TT rises.
3. RMSE nearly converges at sample sizes of 20000 and 60000 for the network structures $[4-10-10-10-1]$ and $[4-10-10-10-10-1]$ respectively, and TT is nearly satisfactory.



(a) (b)

Fig. 3 Box plots with RMSE and TT for different tuples.

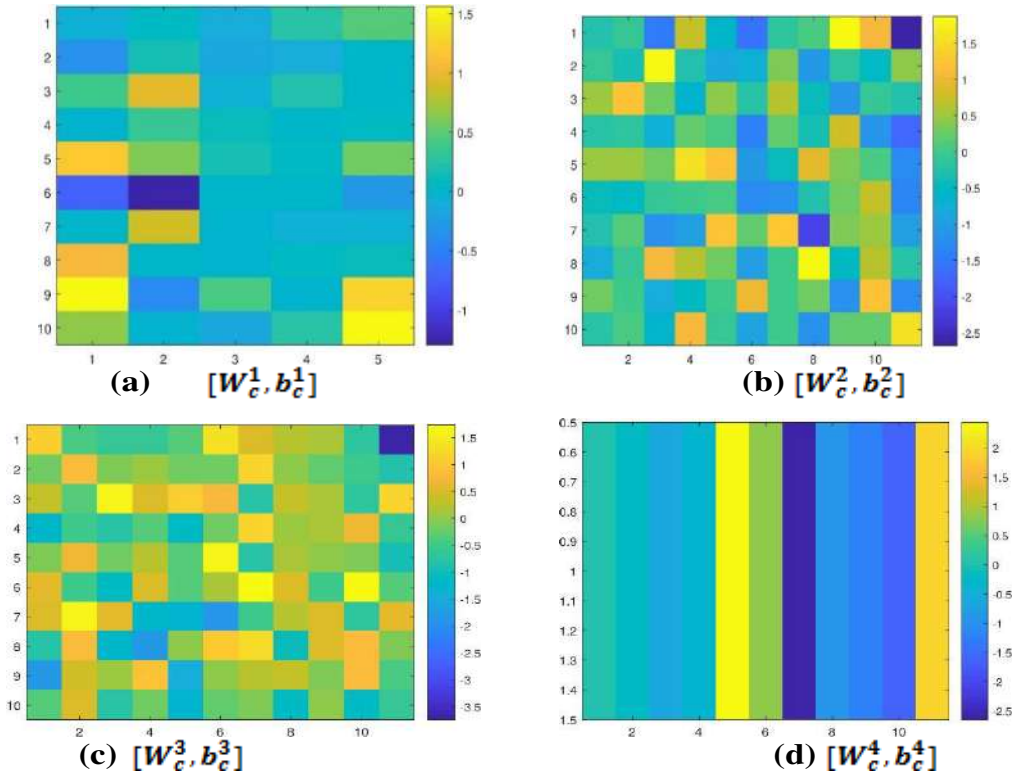
Random factors, like data selection, starting weights, and biases may have an impact on RMSE and TT. Therefore, studies on RMSE and TT in relation to random elements are important.

4.2 RMSE and TT Distribution:

Tuple	SD of RMSE	SD of TT (Sec)	Avg of RMSE	Avg of TT (Sec)
$S1$	0.013	11.391	0.107	11.170
$S2$	0.005	12.268	0.086	17.972
$S3$	0.002	7.687	0.083	28.688
$S4$	0.015	2.683	0.0478	8.087
$S5$	0.0111	14.113	0.056	15.117
$S6$	0.001	232.211	0.004	408.511
$S7$	0.0006	328.873	0.001	4378.897

RMSE and TT box graphs for the tuples $S1 = ([4 - 10 - 1], 1000)$, $S2 = ([4 - 10 - 1], 10000)$, $S3 = ([4 - 10 - 1], 20000)$, $S4 = ([4 - 10 - 10 - 1], 1000)$, $S5 = ([4 - 10 - 10 - 10 - 1], 1000)$, $S6 = ([4 - 10 - 10 - 10 - 1], 20000)$, $S7 = ([4 - 10 - 10 - 10 - 10 - 1], 60000)$, are displayed in figure (3a), (3b) respectively. Ten random values had been considered for each box plot because of random elements. It is difficult to find the most reliable combination. The idea of decreasing common RMSE and TT in addition to decreasing SD of RMSE and TT has been carried out in this examination.

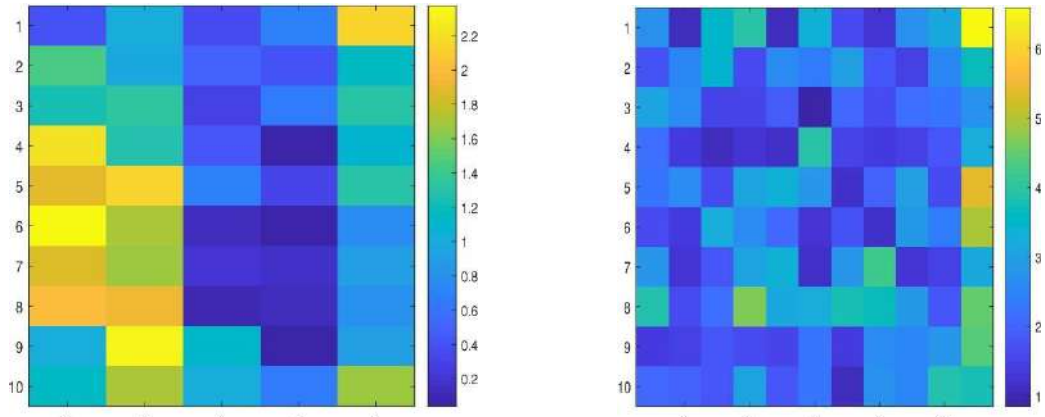
The average (Avg) and standard deviation (SD) of the RMSE and TT are displayed in table 2 Average RMSE values for $S6$ and $S7$ are lower than those for other tuples at the expense of acceptable TT. The SD of the RMSE for $S6$ and $S7$ is likewise less. However, the TT's Avg and SD values of $S6$ and $S7$ are higher. $S6$ and $S7$ are chosen for further study since they have low RMSE and acceptable TT.



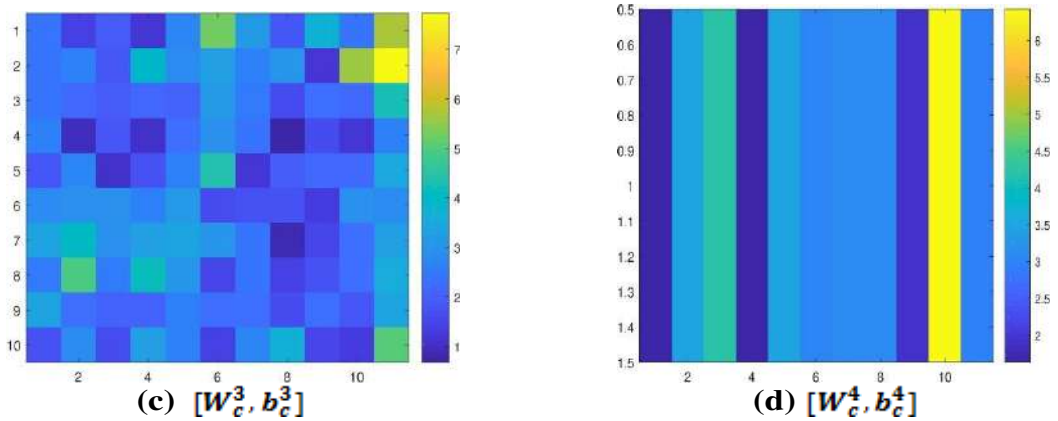
(a) $[W_c^1, b_c^1]$ **(b)** $[W_c^2, b_c^2]$

(c) $[W_c^3, b_c^3]$ **(d)** $[W_c^4, b_c^4]$

4.3 Converged Weights and Biases Distribution: Figures (4), (5), illustrate the average and SD of the converged weight and bias matrices of the tuple $S6$, respectively. Figure (6) displays the converged weight and bias matrices for the minimal RMSE of the tuple $S6$ ($S6^*$). For the value $S6^*$, the RMSE and TT (seconds) are 0.0017 and 806.134, respectively. Figures (7), (8), illustrate the average and SD of the converged weight and bias matrices of the tuple $S7$, respectively. Figure (9) illustrates the converged weight and bias matrices for the minimal RMSE of the tuple $S7$ ($S7^*$). For the value $S7^*$, the RMSE and TT (seconds) are 0.00093 and 4579.077, respectively.

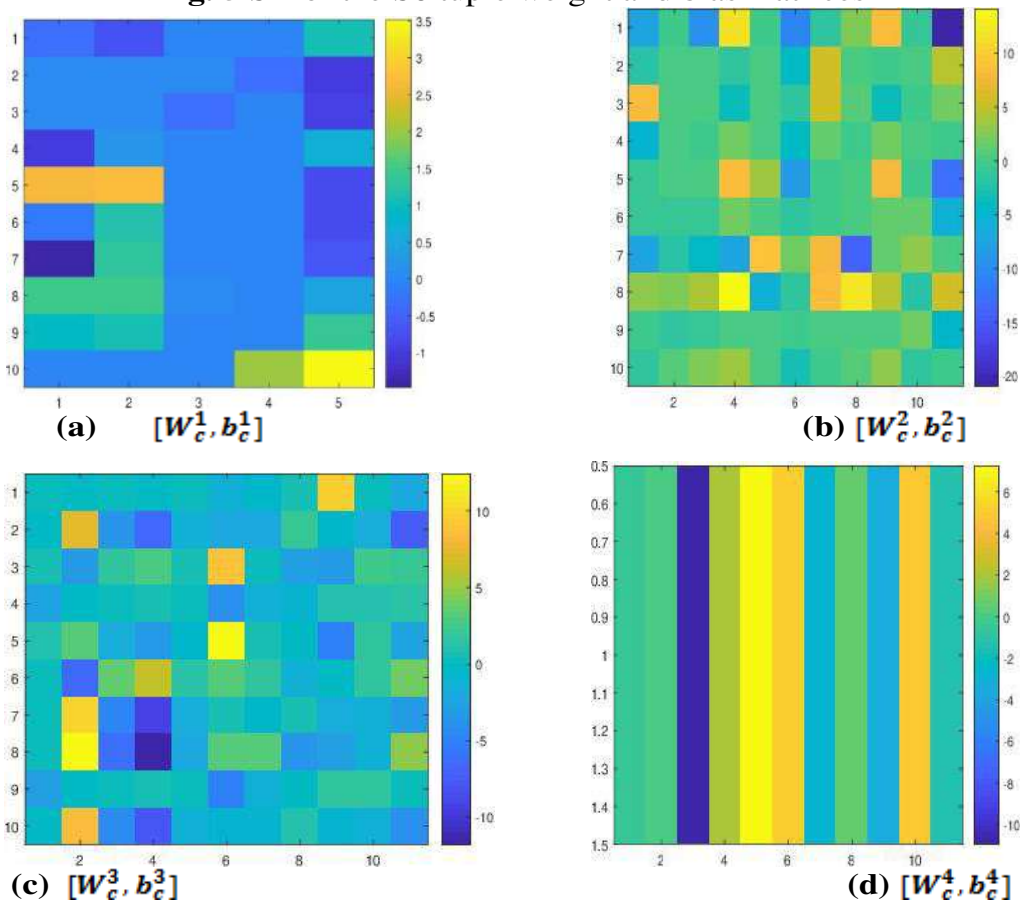


(a) $[W_c^1, b_c^1]$ **(b)** $[W_c^2, b_c^2]$



(c) $[W_c^3, b_c^3]$ (d) $[W_c^4, b_c^4]$

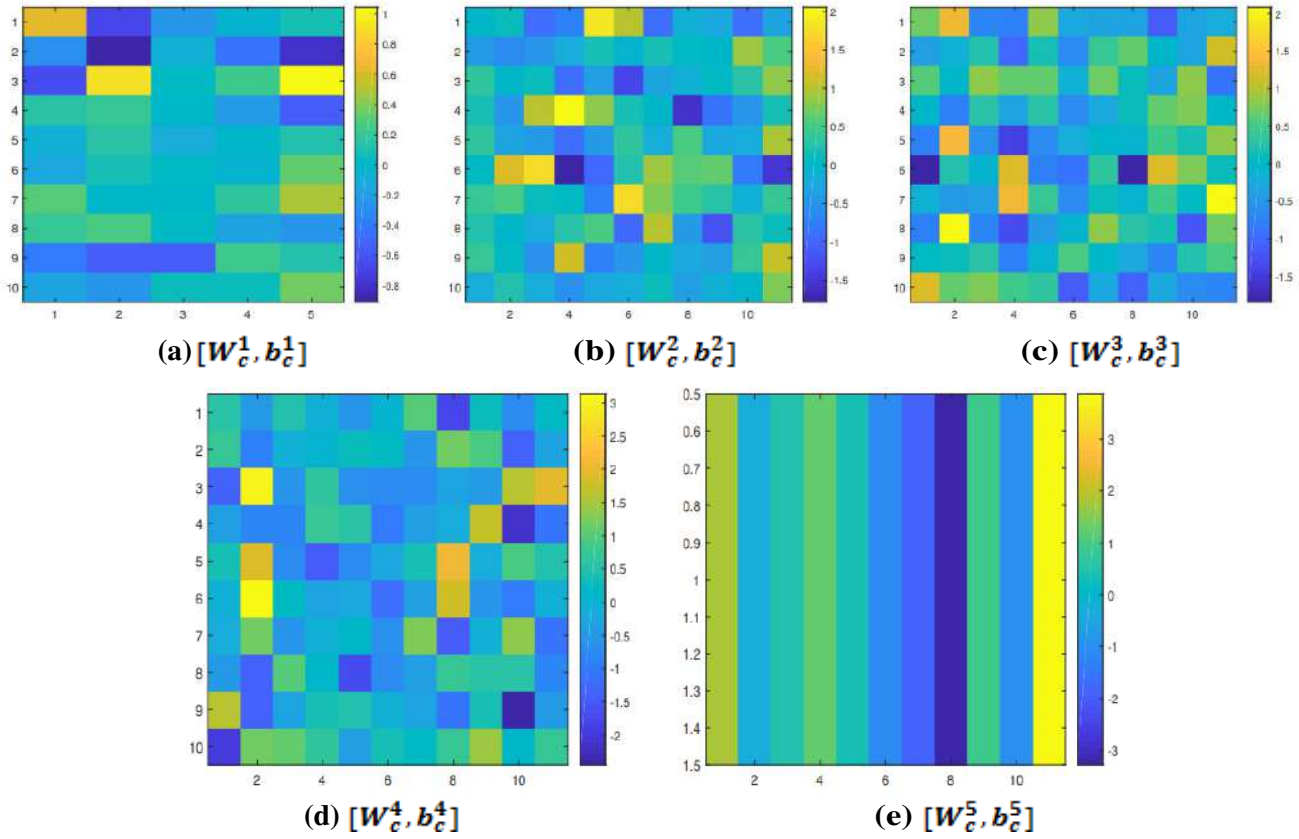
Fig. 5 SD of the S6 tuple weight and bias matrices



(a) $[W_c^1, b_c^1]$ (b) $[W_c^2, b_c^2]$

(c) $[W_c^3, b_c^3]$ (d) $[W_c^4, b_c^4]$

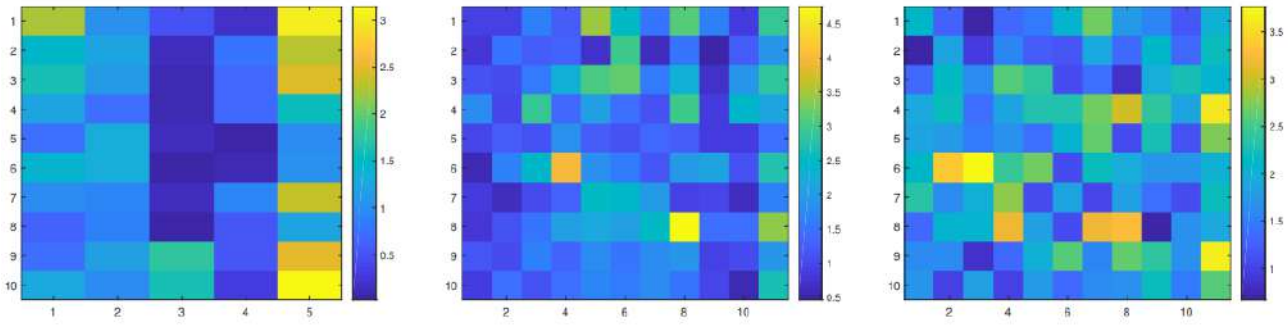
Fig.6 S6* tuple weight and bias matrices



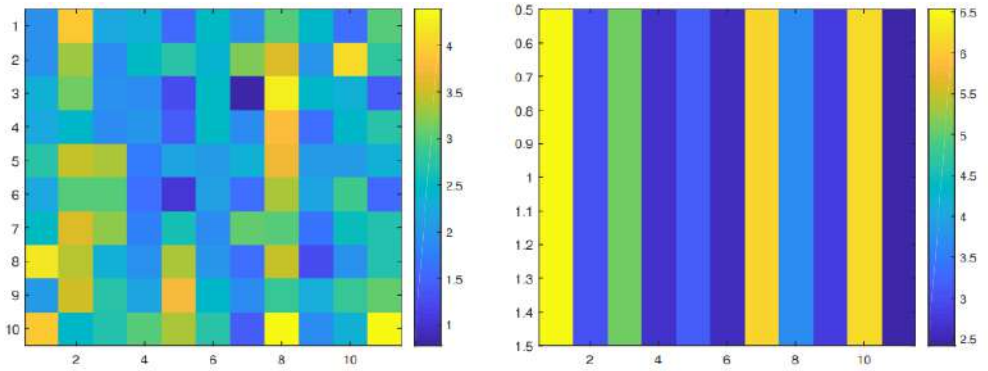
(a) $[W_c^1, b_c^1]$ (b) $[W_c^2, b_c^2]$ (c) $[W_c^3, b_c^3]$

(d) $[W_c^4, b_c^4]$ (e) $[W_c^5, b_c^5]$

Fig. 7 Average of the S7 tuple weight and bias matrices



(a) $[W_c^1, b_c^1]$ (b) $[W_c^2, b_c^2]$ (c) $[W_c^3, b_c^3]$



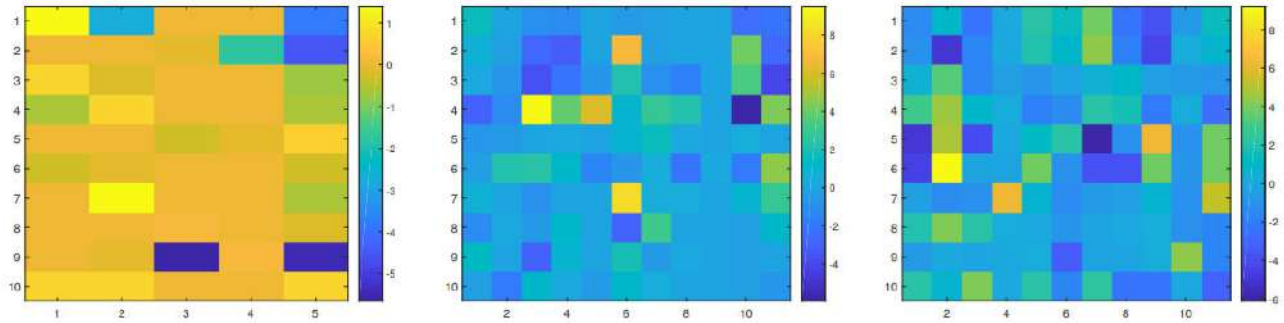
(d) $[W_c^4, b_c^4]$ **(e)** $[W_c^5, b_c^5]$

Fig. 8 SD of the S7 tuple weight and bias matrices

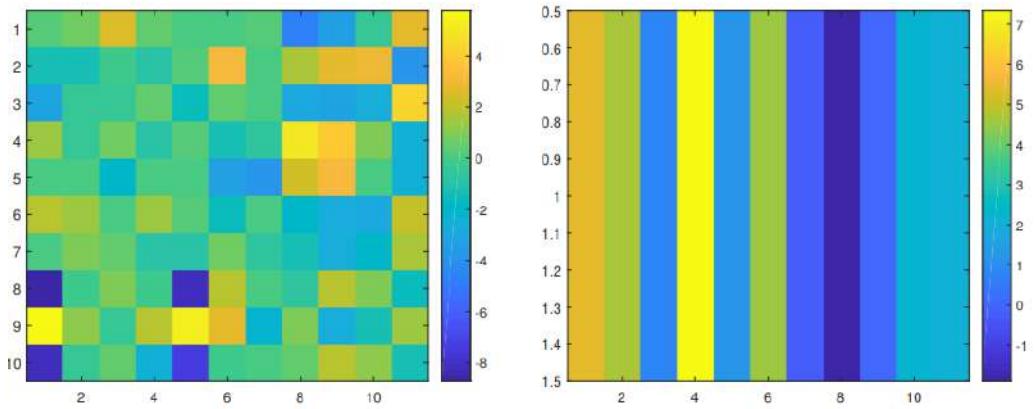
4.4 SSRVM and Proposed Work Comparison: Regarding computation time and solution similarity, series solutions of rectangular vibrating membrane (SSRVM) and ANN approaches have been examined. Images, difference histograms, and PSNR are used to analyze the similarity of the solutions. At every point in the set G, the solutions are estimated.

Figures 10, 11 and 12 successively illustrate the pictures for the solution at $(t, \omega) = (0.2, 0.2), (0.5, 0.5), (1, 1)$. Table 3 provides the PSNR between the SSRVM (6) and $S6^*$ and $S7^*$. PSNR by $S7^*$ is superior to $S6^*$. The difference between SSRVM (6) and $S7^*$ is not substantial in the human visual system because the PSNR value for $S7^*$ is greater than 50 dB. Figure 13 depicts the difference's histogram.

The table 4 shows the average time required by SSRVM, $S6^*$, and $S7^*$ for estimating the solution (u) of equation (6) at all points of G. Figure (14) displays the average PT of several ANN models. ANN models outperform SSRVM 26 times faster in terms of solution prediction.

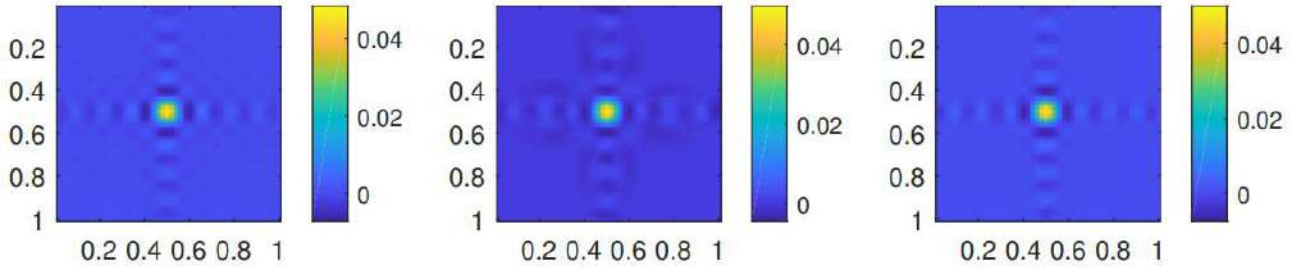


(a) $[W_c^1, b_c^1]$ **(b)** $[W_c^2, b_c^2]$ **(c)** $[W_c^3, b_c^3]$



(d) $[W_c^4, b_c^4]$ (e) $[W_c^5, b_c^5]$

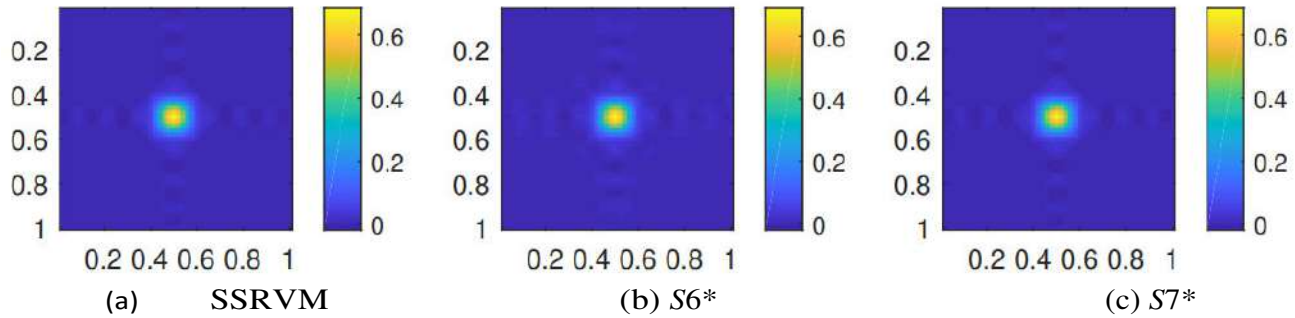
Fig. 9 S7* tuple weight and bias matrices



(a) SSRVM (b) *S*6* (c) *S*7*

Fig. 10 Solution images of equation of rectangular vibrating membrane (1) at (t, ω) = (0.2, 0.2). (a)SSRVM (b) solution by S6* (c) solution by S7*

4.5 Proposed Work and Existing Solution of ANN Comparison: Table 5 provides a planned work that is evaluated in relation to current best practices. A (2+1) dimensional partial differential equation (PDE) has been taken into consideration and analyzed with (1+1) and (2+1) dimensional PDE in previous work in the suggested work. Compared to the state of the art, the error order is reasonable. Compared to Samaniego et al. [31], the average inaccuracy of *S*6 and *S*7 is lower.

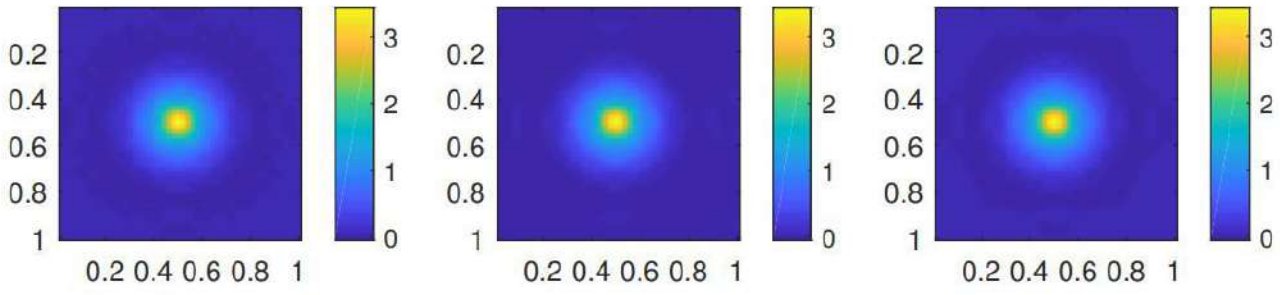


(a) SSRVM (b) *S*6* (c) *S*7*

Fig. 11 Solution images of equation of rectangular vibrating membrane (1) at (t, ω) = (0.5, 0.5). (a) SSRVM (b) solution by S6* (c) solution by S7*

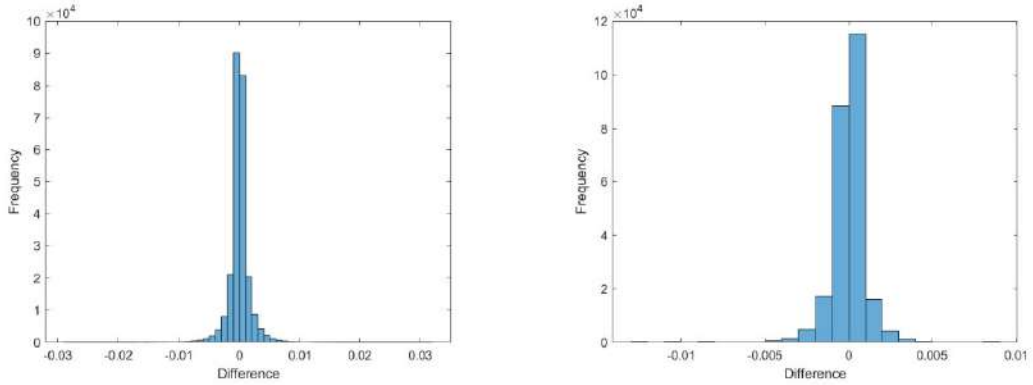
Table 3 PSNR (dB) between images of series solution (6) of equation of rectangular vibrating membrane (1) and solution by tuples S6*, S7*.

S.N.	Time, Frequency (t, ω)	PSNR (S6*)	PSNR (S7*)
1	(t, ω) =0.1, 0.1	62.79	79.01
2	(t, ω) =0.2, 0.2	63.79	72.63
3	(t, ω) =0.2, 0.5	60.22	68.02
4	(t, ω) =0.4, 0.4	59.59	66.58
5	(t, ω) =0.4, 0.6	58.62	63.22
6	(t, ω) =0.5, 0.5	59.04	62.33
7	(t, ω) =0.5, 0.9	54.73	57.41
8	(t, ω) =0.8, 0.8	52.51	56.21
9	(t, ω) =0.9, 0.9	50.25	54.47
10	(t, ω) =1.0, 0.2	59.00	61.85
11	(t, ω) =1.0, 0.4	59.00	61.85
12	(t, ω) =1.0, 0.6	52.58	57.94
13	(t, ω) =1.0, 0.7	51.57	57.31
14	(t, ω) =1.0, 0.9	49.35	55.81
15	(t, ω) =1.0, 1.0	48.01	54.33



(a) SSRVM (b) *S*6* (c) *S*7*

Fig. 12 Solution images of equation of rectangular vibrating membrane (1) at (t, ω) = (1, 1). (a) SSRVM (b) solution by S6* (c) solution by S7*



(a) *S*6* (b) *S*7*

Fig. 12 Solution images of equation of rectangular vibrating membrane (1) at (t, ω) = (1, 1). (a) SSRVM (b) solution by S6* (c) solution by S7*

Table 4 Average (Avg) and Standard deviation (SD) of PT (seconds) of various models.

	SSRVM	**S6***	**S7***
Avg	40.0816	1.2556	1.5007
SD	1.4669	0.2032	0.1728

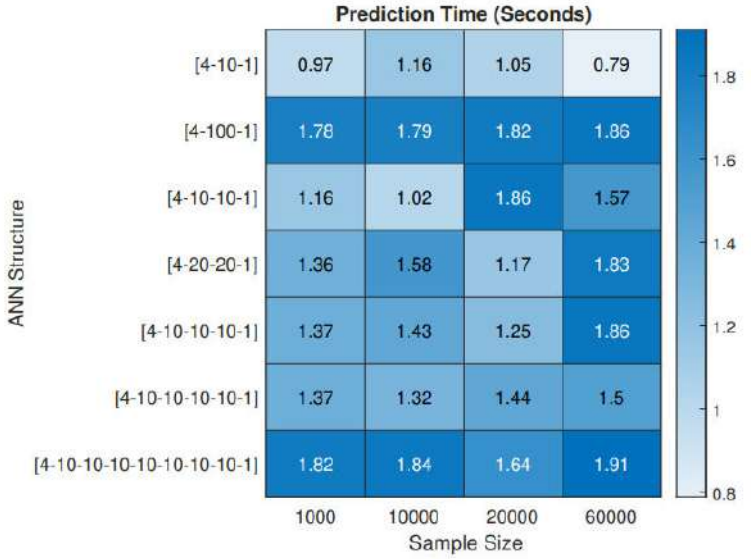


Fig. 14 Average PT of various ANN structures and sample size tuples.

5 CONCLUSIONS

In this paper, an experimental study has been performed for the identification of an artificial neural network (ANN) for the series solution representation of the differential equation model of the rectangular vibrating membrane. Minimization of the modeling error by selection of data-set size and ANN architecture and estimation of ANN parameters (weights and biases) is the main focus of the study. The root mean square error (RMSE) has been usedfor the error computations. Seven different architectures have been used in the experiments, and it has been found that the architectures $[4-10-10-10-1]$ and $[4-10-10-10-10-1]$ with data-set size 20000 and 60000 respectively, offer the lowest error. The corresponding minimum RMSE values are 0.0017 and 0.0009. The obtained RMSE values of 0.0009 is ten times better than the RMSE of the solution reported by the state-of-the-art ANN method.

Table 5 State-of-the-art and proposed work comparison.

References	**TT (Hrs)**	**Error**	**PDE**	**PT (Sec)**
Jafarian and Baleanu [18]	-	$10^{-}11$	(1+1)	-
Moseley and Markham [2]	-	-	(2+1)	-
Samaniego et al. [31]	-	10^{-3} order	(1+1)	-
Brink et al. [6]	-	-	(2+1)	-
Proposed Work (*S*6, *S*7)	1.1	10^{-4} order	(2+1)	1.5

For the ANN with RMSE 0.0009, the average of weights and biases spans a range [-0.8, 1], [-1.5, 2], [-2, 3], [-3, 3], and the standard deviation of weights and biases spans a range of [-0.8, 1], [-1.5, 2], [-2, 3], [-3, 3], for the first, second, third, fourth, and fifth layers, respectively. The training time of the five layers ANN model is 4579 seconds. The time required by the ANN model to predict the 250000 values is approximately 1.5 seconds, which is 26 times faster than the series solution method.

The peak signal-to-noise ratio (PSNR) has been used to evaluate the acceptance of the solution by the human visual system. Its value is 50 dB between the ANN solution and the series solution. The high value of PSNR ensures that the difference between both solutions is beyond the observation limit of the human visual system.

REFERENCES

[1] R. W. Baker. *Membrane technology and applications*. John Wiley and Sons, 2012.

[2] A. M. Ben Moseley and T. Nissen-Meyer. Solving the wave equation with physics-informed deep learning. *arXiv, Computational Physics*, 2020.

[3] L. Bergmann. Experiments with vibrating soap membranes. *The Journal of the Acoustical Society of America*, 28(6):1043–1047, 1956.

[4] J. Blechschmidt and O. G. Ernst. Three ways to solve partial differential equations with neural networks, a review. *GAMM-Mitteilungen*, 44(2): e202100006, 2021.

[5] A. Botev, H. Ritter, and D. Barber. Practical gauss-newton optimisation for deep learning, 2017.

[6] N. F. D. Brink A.R. and M. C. The neural network collocation method for solving partial differential equations. *Neural Comput and Applic*, 33(5591- 5608), 2021.

[7] W. Cai, X. Li, and L. Liu. A phase shift deep neural network for high frequency approximation and wave problems. *SIAM Journal on Scientific Computing*, 42(5): A3285–A3312, 2020.

[8] R. Courant and D. Hilbert. *Methods of Mathematical Physics: Partial Differential Equations*. John Wiley and Sons, 2008.

[9] H. B. Demuth, M. H. Beale, O. De Jess, and M. T. Hagan. *Neural Network Design*. Martin Hagan, 2nd edition, 2014.

[10] V. Dwivedi and B. Srinivasan. Physics informed extreme learning machine (pielm), a rapid method for the numerical solution of partial differential equations. *Neurocomputing*, 391:96–118, 2020.

[11] W. E, J. Han, and A. Jentzen. Deep learning based numerical methods for high-dimensional parabolic partial differential equations and backward stochastic differential equations. *Communications in Mathematics and Statistics*, 5(4):349–380, Nov 2017.

[12] T. Gilet and J. W. M. Bush. The fluid trampoline: droplets bouncing on a soap film. *Journal of Fluid Mechanics*, 625:167–203, 2009.

[13] R. C. Gonzalez and R. E. Woods. *Digital Image Processing*. Prentice Hall, 2002.

[14] W. Greiner. The vibrating membrane, classical mechanics. *Springer*, pages 133–157, 2009.

[15] M. Hagan and M. Menhaj. Training feedforward networks with the marquardt algorithm. *IEEE Transactions on Neural Networks*, 5(6):989–993, 1994.

[16] V. Henner, T. Belozerova, and M. Khenner. *Ordinary and partial differential equations*. CRC Press, 2013.

[17] J. Henseler and P. J. Braspenning. Membrain: a cellular neural network model based on a vibrating membrane. *International journal of circuit theory and applications*, 20(5):483–496, 1992.

[18] A. Jafarian and D. Baleanu. Application of anns approach for wave-like and heat-like equations. *Open Physics*, 15(1):1086–1094, 2017.

[19] C. H. Jenkins and U. A. Korde. Membrane vibration experiments: An historical review and recent

results. *Journal of Sound and Vibration*, 295(3):602–613, 2006.

[20] Y.-T. C. Jichun Li. *Computational Partial Differential Equations Using MATLAB*. CRC Press, 2nd edition, 2019.

[21] M. N. Kobrak. Systematic and statistical error in histogram-based free energy calculations. *Journal of Computational Chemistry*, 24(12):1437– 1446, 2003.

[22] H. Lee and I. S. Kang. Neural algorithm for solving differential equations. *Journal of Computational Physics*, 91(1):110–131, 1990.

[23] G. M. Raissi, P. Perdikaris. Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *Journal of Computational Physics*, 378:686–707, Feb. 2019.

[24] S. Mall and S. Chakraverty. Comparison of artificial neural network architecture in solving ordinary differential equations. *Advances in Artificial Neural Systems*, 2013, 2013.

[25] M. Mirzargar, R. T. Whitaker, and R. M. Kirby. Curve boxplot: Generalization of boxplot for ensembles of curves. *IEEE Transactions on Visualization and Computer Graphics*, 20(12):2654–2663, 2014.

[26] R. Y. X. X. S. Y. L. Nie and Y. Chen. A physics based neural network way to perform seismic full waveform inversion. *IEE Access*, 2020.

[27] V. K. Pandey, H. Agarwal, and A. K. Aggarwal. Time and solution error analysis of neural network model of (2+ 1) dimensional wave equation. *Sādhanā*, 48(1):2, 2022.

[28] S. S. Rao. *Engineering optimization: theory and practice*. John Wiley and Sons, 2019.

[29] F. Rocca, C. Cafforio, and C. M. Prati. Synthetic aperture radar: A new application for wave equation techniques. *Geophysical Prospecting*, 37:809–830, 1989.

[30] F. Rosenblatt. The perceptron - a perceiving and recognizing automaton. Technical Report 85-460-1, Cornell Aeronautical Laboratory, Ithaca, New York, January 1957.

[31] E. Samaniego, C. Anitescu, S. Goswami, V. Nguyen-Thanh, H. Guo, K. Hamdia, X. Zhuang, and R. T. An energy approach to the solution of partial differential equations in computational mechanics via machine learning: Concepts, implementation and applications. *Computer Methods in Applied Mechanics and Engineering*, 362:112790, Apr 2020.

[32] S. Sandro. *Partial Differential Equations in Action: From Modelling to Theory - Unitex*. Springer Publishing Company, Incorporated, 2nd edition, 2015.

[33] P. Szolgay, G. Voros, and G. Eross. On the applications of the cellular neural network paradigm in mechanical vibrating systems. *IEEE Trans- actions on Circuits and Systems I: Fundamental Theory and Applications*,40(3):222–227, 1993.

[34] T. Xiao and M. Frank. Using neural networks to accelerate the solution of the boltzmann equation. *Journal of Computational Physics*, 443:110521, 2021.

[35] T. S. A. Yeung, K. C. Cheung, E. T. Chung, S. Fu, and J. Qian. Learning rays via deep neural network in a ray-based ipdg method for high-frequency helmholtz equations in inhomogeneous media. *Journal of Computational Physics*, 465:111380, 2022.

[36] Z. Zhao, M. Jin, F. Qin, and S. X. Yang. A novel neural network approach to modeling particles distribution on vibrating screen. *Powder Technology*,382:254–261, 2021.