Applied Mathematics and Computational Intelligence
Volume 10, No.1, Dec 2021 [1-17]

Damage Detection Formulation using Inverse Frequency Analysis incorporating Artificial Neural Network for Kirchhoff Plate Theory

M.H. Mokhtaram[1*], M.A. Mohd Noor[2], M.Z. Jamil Abd Nazir[2], D. Ahmad[1], M.K Marwah[1], H. Hamid[1], and A.Y. Mohd Yassin[3]
[1]Faculty of Engineering and Life Sciences, Universiti Selangor, Jalan Timur Tambahan, 45600 Bestari Jaya, Selangor, Malaysia.
[2]School of Civil Engineering, Faculty of Engineering, Universiti Teknologi Malaysia, 81310 Skudai, Johor, Malaysia.
[3] School of Energy, Geoscience, Infrastructure and Society, Heriot-Watt University Malaysia, 62200 Putrajaya, Malaysia.

[*] Corresponding author: mokhtazul@unisel.edu.my

Received: 25 May 2020; Accepted: 24 November 2020; Available Online: 29 January 2021

ABSTRACT

Nowadays, several different structural damage detection techniques are being developed with the goal of monitoring structure stability with high accuracy and low cost. One of the well-known techniques is inverse analysis based on model updating methods. However, the main challenges in this technique is the development of algorithms that assist in the processing of the enormous amounts of data for the inverse process. To overcome this, the Artificial Neural Network (ANN) has been used by many researchers to complement existing approaches. The integration of model updating methods and ANN requires not only a wealth of knowledge and experience in structural damage detection, but also appropriate numerical techniques, and proficiency in scripting programming languages. In this paper, the objective is to construct the formulation of structural damage detection using inverse analysis incorporating Artificial Neural Network (ANN) for Kirchhoff plate theory and to establish the source code. The output from the process is stiffness reduction ratio (SRF) while natural frequencies and mode shape as input data. Finite element method (FEM) was used in generating the formulation. The source code of the formulation has been written step-by-step and kept as simple as possible in Matrix Laboratory (Matlab) programming language. The performance of the formulation is verified against numerical work based on simulated damaged. The presented result shows that, this formulation exhibits excellent performance thus highly potential for damage detection of the plate structure.

Keywords: Structure Damage Detection, Artificial Neural Network, Inverse Frequency, Kirchhoff Plate Theory, Finite Element Method.

1 INTRODUCTION

Structural damage detection using a model updating method has gained considerable attention from many researchers. This method is based on the fact that damage causes a local change in the structural parameters such as stiffness, mass and damping matrices, which change its modal parameters, *i.e.*, natural frequency and mode shape. Model updating method is an inverse analysis of predicting damage from the differences between the updated structural models and before the presence of damage to localize and determine the extent of the damage. (Teughels *et al.*, 2003; Perera and Torres, 2006; Kouchmeshky *et al.*, 2007; Meruane and Heylen, 2010, 2011). However, this method involves iteration and optimization process, thus the algorithm is exceedingly slow, and the damage assessment process is achieved through costly and time-consuming inverse processes.

In recent decades, with the rapid development of computer technologies, the Artificial Neural Network (ANN) has been intensively studied by many researchers to complement existing approaches. ANN is a mathematical model of biological neural system and theoretical mind. It functions as a simulation of a brain created in the computer to solve real life problems. ANN is a powerful tool to model a complicated relationship between various input and output parameters. It has the ability to learn from their experience in order to improve their performance and adapt to changes in the environment.

The first application of ANN in structural damage detection was published by Wu *et al*. (1992). Frequency response function (FRF) acceleration data was used as input whilst binary number (0 and 1) was used as output to represent undamaged and damaged member of the analysis. The same binary system was adopted by Povich and Lim (1994) as output analysis of trusses system. Since then, it has been used for various structural elements such as beams (Marwala and Hunt, 1999), trusses (Yun and Bahng, 2000; He-sheng *et al*., 2005) and composite frames (Zapico *et al*., 2003). The use of ANN to the model updating method has attracted some researches as well. Sahoo and Maity (2007), published a hybrid-neuro genetic algorithm in order to automate the design by considering the frequency and strain as input parameter, and the location and amount of damage as output parameter. Gonzales and Perez (2011), extend the work with the aim of establishing the flexural damage in the girders of a bridge. Arangio and Beck (2012) enhanced the model by applying Bayesian probability logic and the improvement in the results are discussed. Meruane and Mahu (2014) used antiresonant frequencies as input data and developed a procedure for mass-spring system and a beam with multiple damage scenarios. A comprehensive review has also been published by Sarah *et al*., (2019) under the topic of dynamic analysis in frequency domain for structural health monitoring. The application of ANN in the detection of structural damage has been a favourite topic of researchers to date. Kim *et al*. (2020) used sensor fusion incorporating with the ANN model for detecting damage of a bottom-set gillnet. Jayasundara *et al*., (2020) proposed modified forms of the modal flexibility (MMF) and modal strain energy (MMSE) based damage indices coupled with the ANN technology to provide damage assessment on arch bridges. Padil *et al*., (2020) proposed a combination of a non-probabilistic method with principal component analysis (PCA) to consider the problem of the existing uncertainties and the inefficiency of using FRF data in ANN-based damage detection.

The application of ANN as an alternative to the model updating method requires complex mathematical calculation, hence, there is no specific recommendation on suitable design for it (Shankar, 2009). The integration of model updating methods and ANN requires not only a wealth of knowledge and experience in structural damage detection, but also appropriate numerical techniques, and proficiency in scripting programming languages. Therefore, the objective of this study is to construct a formulation of structural damage detection using inverse analysis incorporating ANN for Kirchhoff plate theory and to establish the source code. The source code will be written step-by-step and kept as simple as possible in Matrix Laboratory (Matlab) programming language. This work lies in the integrated formulation between Kirchhoff Plate Theory and ANN to formulate inverse frequency analysis which in the knowledge of this study, is the first-ever done thus improving the current knowledge of the model updating method in damage assessment. A two-span plate is used as an example in this study. The effectiveness of the developed formulation is then compared with simulated damage. The term 'simulated damage' refers to the damage structure data generated based on numerical analysis.

2 INVERSE FREQUENCY ANALYSIS

The idea of the developed formulation is based on inverse frequency analysis, which is the insertion of the measured natural frequencies and mode shapes into the characteristic dynamic equation. Then the equation can be inversed with these known values to yield the damaged stiffness matrix of the structure. Finally, this damaged stiffness matrix is compared with the undamaged stiffness matrix in determining the magnitude and the location of the damages. This correlation requires a sufficient amount of data, therefore ANN has been employed to provide the inverse quantities of the problem through a training process. Since this operation is best handled by matrix operation, FEM is the best choice of tool.

Figure 1 shows the overall flowchart of the developed formulation. The first step is build up a mathematical model based on Kirchhoff plate theory. Next process is to generate training data that will be used as input and output for ANN training process. The data are generated randomly with magnitude and the damage location were randomly selected as well. Next, construct an ANN model. The two main steps in constructing an ANN model are a selection of ANN architecture and ANN training process. Finally, by completing the network training, simulation is performed to predict the damage.

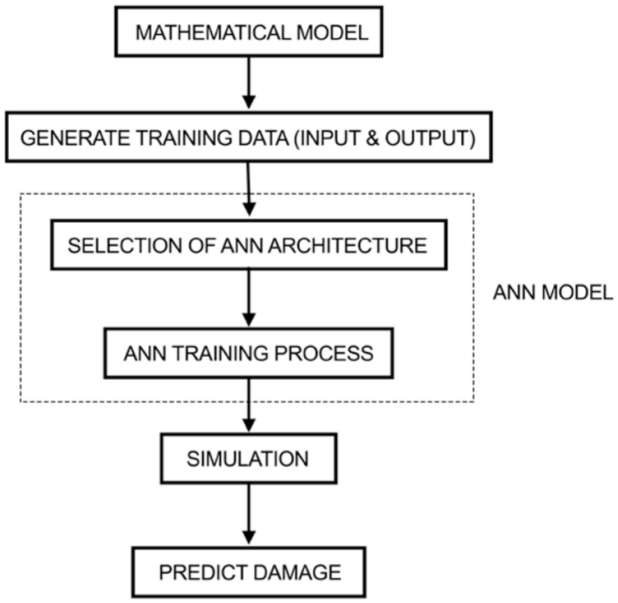


Figure 1. Flowchart of an overall developed formulas

3 MATHEMATICAL MODEL

The Kirchhoff plate theory is a two-dimensional mathematical model that is used to determine the stresses and deformations in thin plates subjected to forces and moments. This theory is an extension of Euler-Bernoulli beam theory with the assumption that a mid-surface plane can be used to represent a 3-dimensional plate in 2-dimensional form.

3.1 Equation of Motion for Kirchhoff Plate Theory

The damage detection problem can be explained by the equation of motion describing the undamped free vibration paradigm. The equation is derived by applying d'Alembert's dynamic equilibrium principle and can be expressed as below;

$$D\left(\frac{\partial^4 w}{\partial x^4}+2\frac{\partial^4 w}{\partial x^2 \partial x^2}+\frac{\partial^4 w}{\partial y^4}\right)+2\rho h\frac{\partial^2 w}{\partial t^2}=0 \tag{1}$$

and

$$D=\frac{Eh^3}{12(1-v^2)} \tag{2}$$

where w is the deflection of plate, ρ is the density and D is the flexural rigidity of plate. E and v are Young's modulus and Poisson's ratio, respectively.

3.2 Numerical Model by Finite Element Method (FEM)

To obtain the FEM formulation, it is necessary to discretize the Equation (1) by Galerkin Weighted-residual Method (Galerkin WRM). The discretization process begins with the provision of the interpolation function;

$$w(x,y)=[N_i]\{d_i\} \tag{3}$$

where N is the FEM shape function matrix and d represent the vector of nodal point displacement. The subscripts i refer to the nodal points, which is at each nodal point 3 displacement components are to be considered;

$$d_i=\begin{Bmatrix} w \\ \theta_x \\ \theta_y \end{Bmatrix}_i=\begin{Bmatrix} w \\ \partial w/\partial y \\ -\left(\partial w/\partial x\right) \end{Bmatrix}_i \tag{4}$$

Inserting Equation (3) into Equation (1) to establish the governing equation in terms of shape functions and DoFs, then multiply with weight functions, N_i consecutively and integrate the inner product to obtain the discretised equation. Next, conduct integration by parts (IBP) to optimize the continuity relaxation. By expressing the time second derivative by double dot, final equation can be given in matrix form as;

$$[k]\{d\}+[m]\{\ddot{d}\}=0 \tag{5}$$

Where $[k]$ is the stiffness matrix and $[m]$ is the mass matrix. $\{d\}$ and $\{\ddot{d}\}$ are vector of displacement and acceleration, respectively. The solution of the equation of motion can be expressed as;

$$\{d\}=\{\hat{d}\}\sin(\omega t)=0 \tag{6}$$

Based on Equation (6), $\{d\}$ vary in a harmonic manner with amplitudes $\{\hat{d}\}$ and natural frequencies ω. Inserting Equation (6) into Equation (5) and by conducting the differentiation in time twice would give;

$$([k] - \omega^2[m])\{\hat{d}\} = 0 \tag{7}$$

Equation (7) is a characteristic equation and the non-trivial solution of this equation is of interest in this study. The solution of natural frequencies ω requires that the determinant of the equations equal to zero. The mass matrix $[m]$ is unaltered even in a damaged condition. The matrices $[k]$ and $[m]$ can be given as;

$$k = \int_{\Omega} B^T E B d\Omega \tag{8}$$

$$m = \rho \int_{\Omega} N_i^T N_j d\Omega \tag{9}$$

where;

$$B = \begin{Bmatrix} \partial^2 w/_{\partial x^2} \\ \partial^2 w/_{\partial y^2} \\ -2\left(\partial^2 w/_{\partial x \partial y}\right) \end{Bmatrix} \tag{10}$$

N_i and N_j represent the corresponding matrices of the shape functions. Algorithm 1 shows the steps involved in solving the non-trivial solution of a characteristic equation.

Algorithm 1. Steps in solving the non-trivial solution of a characteristic equation

Step 1 Data preparation.
Step 2 Develops matrices $[k]$ and $[m]$. The Matlab code can be referenced in Appendix A (i).
Step 3 Assemble local matrix to global matrix. The Matlab code can be referenced in Appendix A (ii).
Step 4 Impose boundary condition.
Step 5 Solve the equation. This process can automatically be carried out using the built-in function '*eig ()*' in Matlab programming language.

3.3 Validation of the equation

The characteristic equation in Equation (7) has been validated by comparing the natural frequency obtained with the experimental work by Bakhary et al., (2007) and established commercial software, *i.e.*, COMSOL. Simply supported element with dimensions of 6400 $mm \times$ 800 $mm \times 100mm$ is modelled. The material properties used are: $E = 3.3 \times 10^{10}\, {}^{N}/_{mm^2}$, $\rho = 2.45 \times 10^{3}\, {}^{kg}/_{m^3}$, and $v = 0.2$. Figure 2 and Table 1 show the mode of frequencies and comparison results against experimental and COMSOL, respectively.

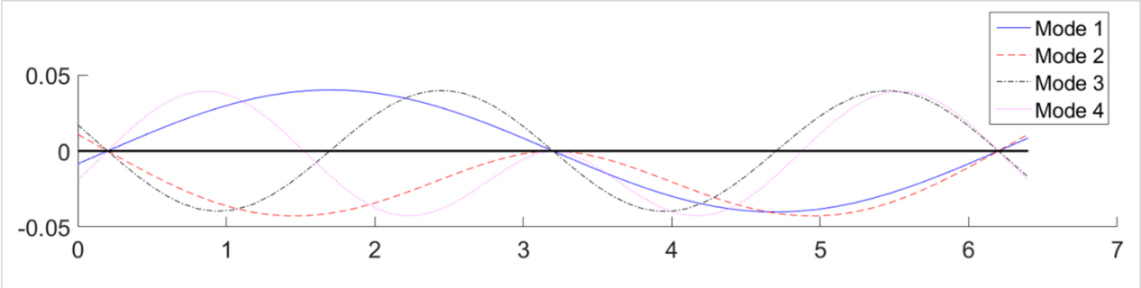


Figure 2. Mode of frequencies

Table 1 Comparison results for validating

Mode	Bakhary et al. (2007) (experimental)	COMSOL (FEM)	Present (FEM)
1	17.81	18.10	18.14
2	25.46	28.32	28.53
3	-	72.05	73.15
4	-	90.97	92.71

Table 1 shows that the natural frequency obtained from the developed formulation has magnitude comparable to experimental and commercial software. Figure 3 shows the convergence of natural frequency at mode 1 for the developed formulation with an increasing number of nodes. Based on that figure, the convergence achieved by the developed formulation is shown to be faster and in an excellent agreement against the converged result from commercial software with the number of nodes required only less than 150. Such verifications give a confirmed level of confidence and validate the use of the developed equation of motion for Kirchhoff plate theory in this study. Note that a small discrepancy between the numerical analysis and the experimental solution occurs due to uncertainty and experimental errors during the sample preparation.

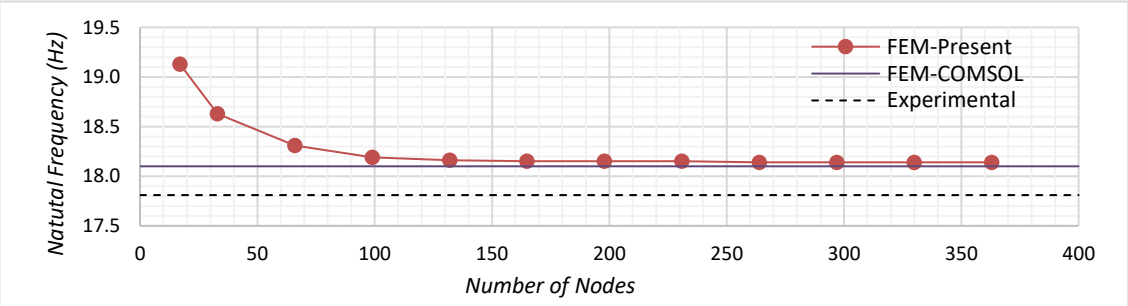


Figure 3. Convergence of natural frequency at mode 1

4 GENERATE TRAINING DATA

The process of ANN model requires a largest possible range of input and output data. The input data consist of the normalized data of natural frequencies and mode shapes, which are generated randomly from the non-trivial solution of the characteristic equation in Equation (7), and the

output layers consist of a stiffness reduction ratio (SRF). The normalized inputs are calculated as below;

$$P_n = \frac{P - min(P)}{\max(P) - min(P)} \tag{11}$$

where P is a column of frequencies and mode shapes, and P_n is the normalized values. The SRFs are calculated as below;

$$SRF = 1 - \frac{E'}{E} \tag{12}$$

where E is the Young's modulus in the undamaged state and E' is Young's modulus at the desired damage level. The SRF data is normalized at interval of [0, 1], which is 1 represented fully undamaged state and 0 fully damaged state. All the data are obtained from FEM algorithm, which is involved generating large numbers of damage cases. Algorithm 2 shows the steps involved in generating training data for the ANN model.

Algorithm 2. Steps in generating training data for the ANN model

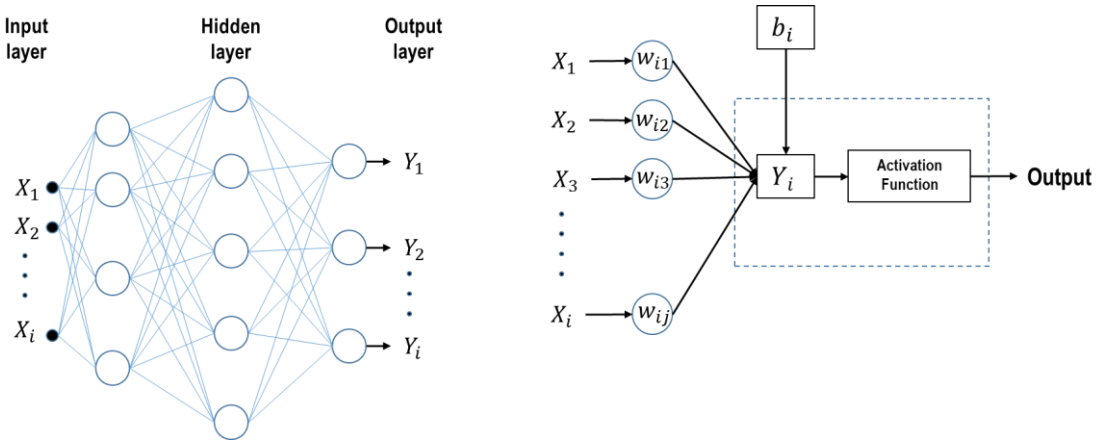
Step 1 Number of training data.
Step 2 Generates the frequency and mode shape (input) data for undamaged and damage state. The Matlab code can be referenced in Appendix B (i).
Step 3 Generates the normalized input and SRF parameters. The Matlab code can be referenced in Appendix B (ii).
Step 4 Test the data to the ANN model to obtain the locations and severities of damages.

5 ARTIFICIAL NEURAL NETWORK (ANN) MODEL

ANN is a self-organizing computational technique and it can solve many functions through pattern recognition. ANN can effectively be used to reconstruct nonlinear relationship learning from training. There are two main steps in constructing an ANN model which are selection of ANN architecture and training procedures for ANN model.

5.1 ANN Architecture

ANN architecture consist three different layers which are input layer, hidden layer and output layer as shown in Figure 4 (a). These layers are interconnected by the neurons. The main elements of an artificial neuron are indicated in Figure 4 (b) which includes weights w_{ij}, bias b_i and activation function. The neuron that were combined and arranged in layer is known as multilayer perceptron (MLP).



(a) (b)

Figure 4. (a) Architecture of ANN-MLP. (b) Elements of an artificial neuron

In this study, ANN-MLP is used. Outputs Y_i of ANN can be defined by following equation;

$$Y_i = f_i\left(\sum_{j=1}^{n} w_{ij}X_i + b_i\right) \tag{13}$$

Where X_i indicates the input values, w_{ij} illustrates the connection weight values between input, hidden and output layers, b_i is the bias, and f_i shows the transfer function.

5.2 Training an ANN Model

The ANN training process is a procedure to set the weights in order to minimize the prediction error. It can be performed by introducing a set of input and output data to ANN model. The data will go through the training process and the output will be compared to the desired output. This training process is known as supervised learning which performed by a learning algorithm. Through the algorithm, the process will be repeated until the error between the desired and predicted output met the stopping criteria that have been appointed. The differences between desired and predicted output are combined and denoted by an error function. In this study, Mean Squared Error (MSE) has been used as an error function. It can be described as below;

$$MSE = \frac{1}{n}\sum_{j=1}^{n}(O_T - O_P)^2 \tag{14}$$

where O_T and O_P are the target and predicted outputs, and n is the number of data, respectively. The relationship between input and output variables is considered established when MSE value is approaching 0.

5.2.1 Learning Algorithm

In this study, the Levenberg-Marquardt algorithm is used as the learning algorithm and Sigmoid function is employed as non-linear activation functions for all layers. '`Trainlm`' is a network training function that updates weight and bias values according to Levenberg-Marquardt optimization. It is often the fastest backpropagation algorithm in the Matlab toolbox, and is highly

recommended as a first-choice supervised algorithm. Appendix C shows the Matlab learning algorithm of Levenberg-Marquardt with the values of the '`Trainlm`' training parameters used.

5.2.2 Stopping Criteria

The ANN-MLP model is exposing to an over fitting problems. In this case, the training process efficiency continues to increase while the performance of the unseen data become worst. To overcome this problem, early stopping method was used as the stopping criteria in this study. To apply the early stopping method, separate the training data into a training set and validation set. Then, start train only on the training set and evaluate the pre-example error on the validation set once in a while. In this study the error is assessed in every fifty training cycle (epoch). Stop training process when the error function on the validation set is higher than the last time it was checked.

6 NUMERICAL EXAMPLE

Here, numerical example is presented to evaluate the performance of the developed formulation in assessing structural damage detection. A two-span plate with dimension of $6400\ mm \times 800\ mm \times 100\ mm$ was tested. The boundary conditions are idealized as pin supports at the middle span and at $200\ mm$ from left and right end of the plate as illustrated in Figure 5 (a).

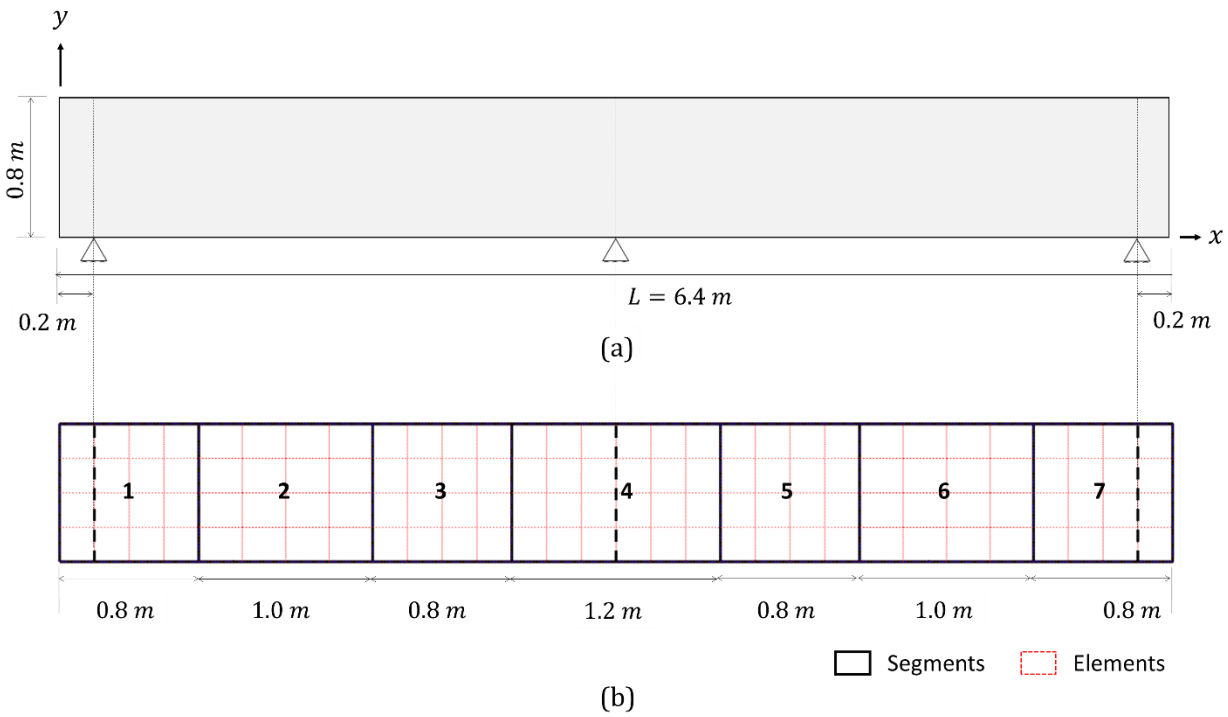


Figure 5. (a) A two-span plate model. (b) Segmentation and FEM elements

For damage detection purposes, the plate is divided to 7 segments and is modelled with 120 elements for FEM discretization as suggested in 3.3. Every segment is assumed to have the same material properties. The properties used are: Young Modulus, $E = 33 \times 10^9\ N/mm^2$, density, $\rho = 2550\ kg/m^3$, and Poisson ratio, $\nu = 0.2$.

The procedure of the damage detection by the developed formulation is then assessed based on Figure 1. In this example, 2000 cases of data sets are generated to train the ANN model. Frequencies and mode shapes for the first four modes and the E values for every segment are used as the input and output respectively. Three damage scenarios are generated to assess the developed formulation performance. Scenario 1 consists of single damage, scenario 2 consists of multiple damage, and scenario 3 consists of damage at the support segment. The damage is imposed by reducing the E values of each corresponding segment. For easy detection of reduced stiffness, the value of E is assumed to be equal to 1. Table 2 shows the E values for all scenarios. The frequencies of the first four modes are shown in Table 3.

Table 2 E values for scenario 1, 2, and 3

Segment		**1**	**2**	**3**	**4**	**5**	**6**	**7**
Scenario 1	***i***	$1.0 \times E$	$0.85 \times E$	$1.0 \times E$	$1.0 \times E$	$1.0 \times E$	$1.0 \times E$	$1.0 \times E$
	ii	$1.0 \times E$	$1.0 \times E$	$1.0 \times E$	$1.0 \times E$	$0.5 \times E$	$1.0 \times E$	$1.0 \times E$
Scenario 2	***i***	$1.0 \times E$	$0.7 \times E$	$1.0 \times E$	$1.0 \times E$	$1.0 \times E$	$0.55 \times E$	$1.0 \times E$
	ii	$1.0 \times E$	$0.3 \times E$	$0.7 \times E$	$1.0 \times E$	$1.0 \times E$	$0.5 \times E$	$1.0 \times E$
Scenario 3	***i***	$0.85 \times E$	$1.0 \times E$	$1.0 \times E$	$0.2 \times E$	$1.0 \times E$	$1.0 \times E$	$0.5 \times E$
	ii	$0.2 \times E$	$0.7 \times E$	$1.0 \times E$	$0.9 \times E$	$1.0 \times E$	$1.0 \times E$	$0.45 \times E$

Table 3 First four frequencies values

	Undamaged	**Damaged**					
		Scenario 1		**Scenario 2**		**Scenario 3**	
		i	***ii***	***i***	***ii***	***i***	***ii***
Mode 1	18.187	18.164	17.86	17.78	17.086	17.076	17.643
Mode 2	28.532	28.684	29.049	29.166	30.035	23.006	27.654
Mode 3	73.151	73.131	72.999	72.873	71.133	65.765	70.462
Mode 4	92.705	92.645	95.315	91.796	90.269	72.822	86.943

7 RESULTS AND DISCUSSION

The developed formulation has been verified by observing it capabilities to identify location of damage in the problem domain. Figure 6 shows the training, validation and test performance of the selected ANN model with increasing number of epochs. It is shown that the best validation performance at 109^{nd} epoch with MSE value of validation 0.00013786. Hence, this ANN model will be used to cater all the damage cases in Table 2 and the predicted results as in Figure 7.

Figure 7 shows the predicted E values in comparison with the actual values. Scenario 1 (i) and (ii) presented single damage with low and high percentage error, respectively. Scenario 2 shows the prediction for multiple damages for two and three segments, and scenario 3 presented damage to the support segments. All of these scenarios represent all possible structural damage. The results shows the prediction values are almost the same to the actual results. Hence, it can be concluded that the develop formulation of damage detection using inverse frequency analysis incorporating ANN is able to detect the damage location and SRF values of the structure correctly.

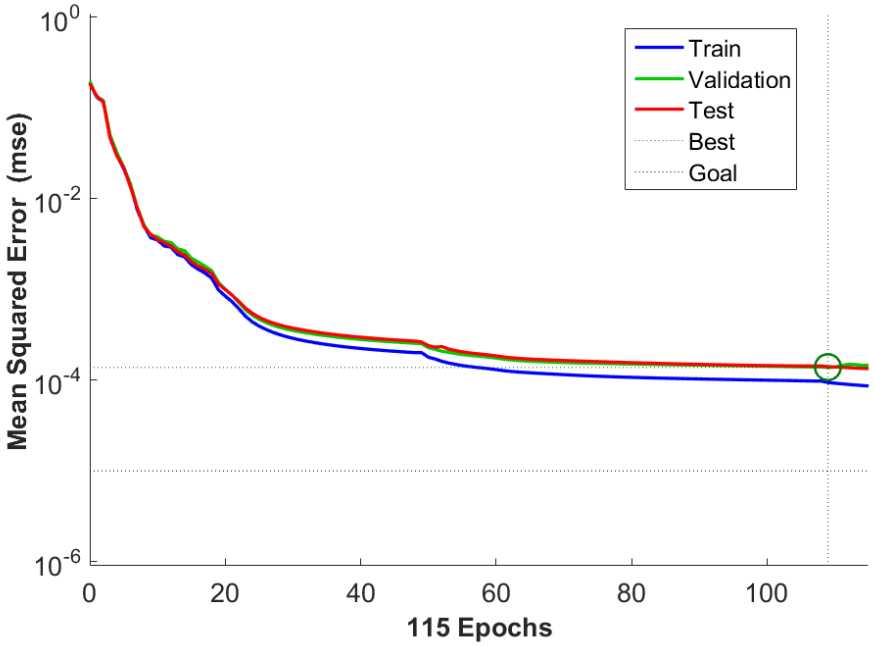
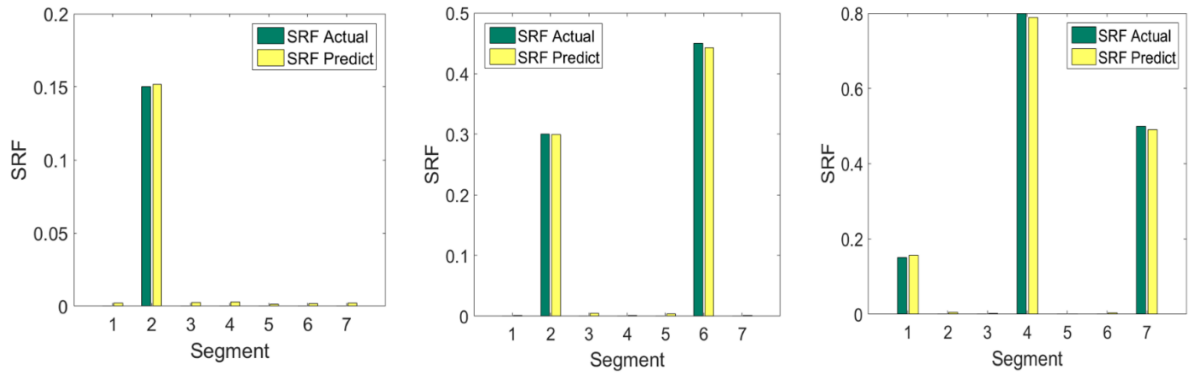
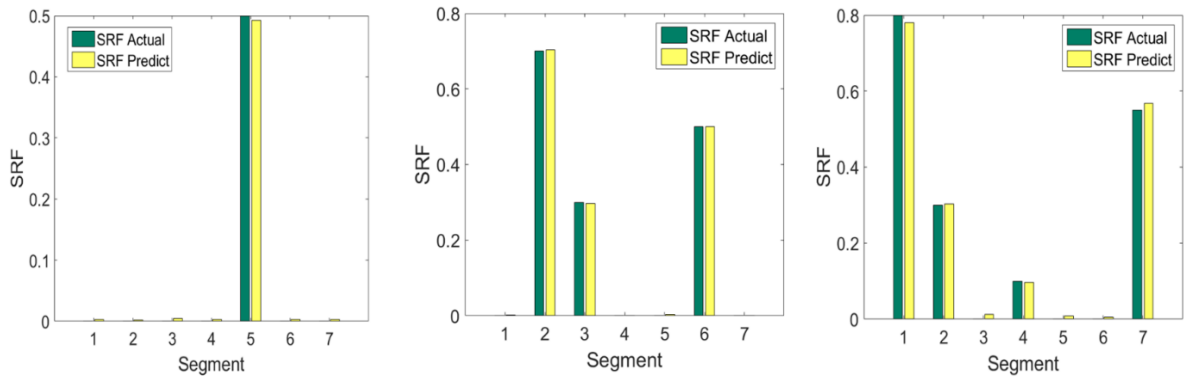


Figure 6: ANN performance with increasing number of epochs



i *i* *i*



ii *ii* *ii*

(a) (b) (c)

Figure 7: ANN prediction results for (a) scenario 1, (b) scenario 2, and (c) scenario 3

8 CONCLUSION

In this study, the development of structural damage detection formulation using inverse frequency analysis with ANN for Kirchhoff plate theory is presented. The integration of the formulation through corresponding algorithms have been discussed in a step-by-step manner. FEM was used in generating the formulation. The source code of the formulation has been written and kept as simple as possible in Matlab programming language. A MLP with Levenberg-Marquardt back-propagation algorithm is utilized to train the ANN model. Sigmoid functions are employed as nonlinear activation functions for all layers. To reduce the effect of overfitting, the early-stopping method was applied during the training.

The input data of the training consist of natural frequencies and mode shapes. The data are obtained from the solution of motion equation for Kirchhoff plate theory, which involved generating large number of damage cases. The motion equation is solved numerically by using FEM. The output layers during the training consists of Young's modulus, E to represent the stiffness parameter. In the process the stiffness parameter modified to reproduce the measured response as close as possible. Once the ANN model is well-trained, the testing data are then applied to obtain the locations and severities of damages. The severities of the damages is measured by SRF.

The developed formulation was applied to a two-span plate element. The plate domain was divided into 7 segments and modelled with 120 elements for FEM discretization. 2000 cases of data set were used to generate the input data to train the ANN models, with the best validation performance was 109^{nd} epoch with MSE value of validation 0.00013786. Then, the performance validation of the developed formulation was compared with the three types of damage scenarios, where the results clearly show excellent agreement for all case scenarios. Such verifications provide a confirmed level of confidence and validate the use of the developed formulation for assessment of damage detection in building structures.

ACKNOWLEDGEMENTS

The authors acknowledge the Universiti Selangor and the State Government of Selangor for the financial support through Geran Penyelidikan Kerajaan Negeri Selangor 2018 (GPNS18) vote number GPNS-01/UNISEL/18-048 for this research. The authors also acknowledge members of the Malaysian Society for Numerical Methods (MSNM), for assisting in the development of the computer algorithm.

REFERENCES

[1] S. Arangio and J. L. Beck, "Bayesian neural networks for bridge integrity assessment," *Struct. Control Health Monit.*, vol. 19, no. 1, pp. 3–21, 2010.

[2] N. Bakhary, H. Hao, and A. J. Deeks, "Damage detection using artificial neural network with consideration of uncertainties," *Eng. Struct.*, vol. 29, no. 11, pp. 2806–2815, 2007.

[3] H. S. Kim, C. Jin, M. H. Kim, and K. Kim, "Damage detection OF bottom-set gillnet using artificial neural network," *Ocean Eng.*, vol. 208, p. 107423, 2020.

[4] T. He-sheng, X. Song-tao, C. Rong and W. Yuan-gong, "Analyses on structural damage identification based on combined parameters," *Appl. Math. Mech.*, vol.26, pp. 44-51, 2020.

[5] N. Jayasundara, D.P. Thambiratnam, T.H.T. Chan and A.Nguyen, "Damage detection and quantification in deck type arch bridges using vibration based methods and artificial neural networks," *Eng. Fail. Anal.*, vol. 109, p. 104265, 2020.

[6] K. H. Padil, N. Bakhary, M. Abdulkareem, J. Li, and H. Hao, "Non-probabilistic method to consider uncertainties in frequency response function for vibration-based damage detection using Artificial Neural Network," *J. Sound Vib.*, vol. 467, p. 115069, 2020.

[7] B. Kouchmeshky, W. Aquino, J. C. Bongard and H. Lipson, "Co-evolutionary algorithm for structural damage identification using minimal physical testing," *Int. J. Numer. Methods Eng.*, vol. 69, no. 5, pp. 1085–1107, 2007.

[8] T. Marwala and H. Hunt, "Fault identification using finite element models and neural networks," *Mech. Syst. Sig. Process.*, vol. 13, no. 3, pp. 475–490, 1999.

[9] V. Meruane and M. Heylen, "Damage detection with parallel genetic algorithms and operational modes," *Struct. Health Monit.*, vol. 9, no. 6, pp. 481–496, 2010.

[10] V. Meruane and M. Heylen, "An hybrid real genetic algorithm to detect structural damage using modal properties," *Mech. Syst. Sig. Process.*, vol. 25, no. 5, pp.1559–1573, 2011.

[11] V. Meruane and J. Mahu, "Real-Time Structural Damage Assessment Using Artificial Neural Networks and Antiresonant Frequencies," *Shock Vib.*, vol. 2014, pp. 1–14, 2014. Art. no. 653279.

[12] R. Perera and R. Torres, "Structural damage detection via modal data with genetic algorithms," *J. Struct. Eng.*, vol. 132, no. 9, pp.1491–1501, 2006.

[13] C. Povich and T. W. Lim, "An artificial neural network approach to structural damage detection using frequency response function," in *Proceeding of AIAA Adaptive Structures Forum*, Washington, DC, (1994) pp. 151-159.

[14] B. Sahoo and D. Maity, "Damage assessment of structures using hybrid neuro-genetic algorithm," *App. Soft Comp.*, vol. 7, no. 1, pp. 89–104, 2007.

[15] J. Sarah, F. Hejazi, R. S. M. Rashid, and N. Ostovar, "A Review of Dynamic Analysis in Frequency Domain for Structural Health Monitoring," *IOP Conference Series: Earth and Environmental Science*, vol. 357, p. 012007, 2019.

[16] R. Shankar, "An integrated approach for structural health monitoring," Ph.D. dissertation, Dept. Civil Eng., Indian Institute of Technology Delhi, Delhi, India, 2009.

[17] A. Teughels, G. De Roeck, and J. A. K. Suykens, "Global optimization by coupled local minimizers and its application to FE model updating," *Comput. Struct.*, vol. 81, no. 24-25, pp. 2337–2351, 2003.

[18] C. González-Pérez and J. Valdés-González, "Identification of Structural Damage in a Vehicular Bridge using Artificial Neural Networks," *Struct. Health Monit.*, vol. 10, no. 1, pp. 33–48, 2010.

[19] X. Wu, J. Ghaboussi and J. H. Garrett, "Use of neural networks in detection of structural damage," *Comput. Struct.*, vol. 42, no. 4, pp. 649-659, 1992.

APPENDIX A

```
for iDom = 1:size(Data.Domain,2)
    ModE    = Data.Domain(iDom).ModE;     % Young's modulus (Pa or N/m2)
    Thick   = Data.Domain(iDom).Thick;    % Thickness (m)
    Poisson = Data.Domain(iDom).Poisson; % Poisson's ratio
    Density = Data.Domain(iDom).Density; % Density (kg/m2)
    Le_x    = Data.Domain(iDom).SegmentMeshLengthX;
    Le_y    = Data.Domain(iDom).SegmentMeshLengthY;

    for iSeg = 1:size(Data.Domain(iDom).Segment,2)
        Ele  = Data.Domain(iDom).Segment(iSeg).Ele;
        Kglobal_Seg = zeros(Data.TDOF,Data.TDOF);
        Mglobal_Seg = zeros(Data.TDOF,Data.TDOF);

        for iEle = 1:size(Ele,1)
            % Calculate local stiffness and mass
            [klocal,mlocal]= Stiff_Mass_KirchoffPlate...
                             (ModE,Poisson,Density,Thick,Le_x,Le_y);
            % Assemble
            Nd1 = Ele(iEle,1);
            Nd2 = Ele(iEle,2);
            Nd3 = Ele(iEle,3);
            Nd4 = Ele(iEle,4);
            uv  = [3*(Nd1-1)+1 3*(Nd1-1)+2 3*(Nd1-1)+3 ...
                   3*(Nd2-1)+1 3*(Nd2-1)+2 3*(Nd2-1)+3 ...
                   3*(Nd3-1)+1 3*(Nd3-1)+2 3*(Nd3-1)+3 ...
                   3*(Nd4-1)+1 3*(Nd4-1)+2 3*(Nd4-1)+3];
            Kglobal_Seg(uv,uv) = Kglobal_Seg(uv,uv) + klocal;
            Mglobal_Seg(uv,uv) = Mglobal_Seg(uv,uv) + mlocal;
        end
        Data.Domain(iDom).Segment(iSeg).Kglobal_Seg = Kglobal_Seg;
        Data.Domain(iDom).Segment(iSeg).Mglobal_Seg = Mglobal_Seg;
    end
end
```

(i)

```
Kglobal = zeros(Data.TDOF,Data.TDOF);
Mglobal = zeros(Data.TDOF,Data.TDOF);
count = 0;
for iDom = 1:size(Data.Domain,2)
    for iSeg = 1:size(Data.Domain(iDom).Segment,2)
        count = count + 1;
        Kglobal = Kglobal + ...
            Coeff(count,1).*Data.Domain(iDom).Segment(iSeg).Kglobal_Seg;
        Mglobal = Mglobal + ...
            Coeff(count,1).*Data.Domain(iDom).Segment(iSeg).Mglobal_Seg;
    end
end
```

(ii)

APPENDIX B

```
for iTrain = 1:NoTrain
    % Damage Coeff
    Coeff = Coeff(Data);
    Data.Damage(:,iTrain) = Coeff;

    % Damage Stiffness and Mass
    [Kglobal,Mglobal] = Assemble_StiffnessMass(Data,Coeff);

    % Impose Boundary Cond
    [Kglobal,Mglobal] = ImposeBCond(Data,Kglobal,Mglobal);

    % Solve for eigen value
    [eigVe,eigVa]=eig(Kglobal,Mglobal);

    % Extract Data
    nn = size(Kglobal,1);
    [ned,natfreq_Hz_D,eigVe_Mode_Train_D,eigVe_Mode_All_D] = ...
                    Extract_EigenValue(Data,eigVe,eigVa,nn,Coor_xy);

    % Normalization
    A1=eigVe_Mode_Train_D(:,1);   B1=eigVe_Mode_Train_UD(:,1);
    A2=eigVe_Mode_Train_D(:,2);   B2=eigVe_Mode_Train_UD(:,2);
    A3=eigVe_Mode_Train_D(:,3);   B3=eigVe_Mode_Train_UD(:,3);
    A4=eigVe_Mode_Train_D(:,4);   B4=eigVe_Mode_Train_UD(:,4);

    C1=A1'*B1/(B1'*B1);           D1=C1*A1;
    C2=A2'*B2/(B2'*B2);           D2=C2*A2;
    C3=A3'*B3/(B3'*B3);           D3=C3*A3;
    C4=A4'*B4/(B4'*B4);           D4=C4*A4;

    Data.ModeShape(1:4,iTrain)                  = Damage_Freq;
    Data.ModeShape(5:ned+4,iTrain)              = D1;
    Data.ModeShape(ned+4+1:2*ned+4,iTrain)      = D2;
    Data.ModeShape(2*ned+4+1:3*ned+4,iTrain)    = D3;
    Data.ModeShape(3*ned+4+1:4*ned+4,iTrain)    = D4;
end
input = Data.ModeShape;
output = Data.Damage;
```

(i)

```
for i=1:(4*ned+4)
    for j=1:iTrain
        P=input(i,j);
        minP=min(input(i,:));
        maxP=max(input(i,:));
        inputMax(i,1)=maxP;
        inputMin(i,1)=minP;
        inputN(i,j)=(P-minP)/(maxP-minP);
    end
end
for i=1:Total_Seg
    for j=1:iTrain
        Ed=output(i,j);
        EUd=1;
        outputN(i,j)=1-Ed/EUd;
    end
end
```

(ii)

APPENDIX C

```
load input_output.mat

trainLayer = 16;
input      = inputN;
output     = outputN;

% feedforward network.
% training using Levenberg-Marquardt backpropagation.
% Hyperbolic tangent sigmoid transfer function.

 net = newff(input,output,trainLayer,{'tansig' 'tansig'},'trainlm');

% training parameters.
      net.trainParam.show = 50;
      net.trainParam.Ir = 0.05;
      net.trainParam.Ir_inc = 1.05;
      net.trainParam.epochs = 200;
      net.trainParam.goal = 1e-5;
      net=init(net);

% train the network.
[NeuralNetwork,tr,Y,E]=train(net,input,output);

save Simulation.mat

% memory
```