HiSST: International Conference on
High-Speed Vehicle Science Technology
26–29 November 2018, Moscow, Russia

Time-Domain Dynamic Response Prediction: from One Boundary Condition to Another

Jiaming Zhou[1,2], Longlei Dong[1,2], Guirong Yan[1,2]

Abstract

With the purpose of improving the validity and practicality of ground test data for flight vehicle design, a mapping model of the same structure under different boundary conditions is employed to predict the dynamic responses in the time domain. Unfortunately, the mapping model is difficult to solve by traditional mathematical methods. Therefore, a new approach is proposed by using Recurrent Neural Networks (RNNs) combined the Gated Recurrent Unit (GRU). This method is validated on different structures: a three-degree-of-freedom system and a thin plate. The training dataset is generated by numerical computation methods. The predicted results illustrate that this method has good learning and data prediction ability for stationary and non-stationary time sequences with 20% noise. The applications of this method in the aerospace field will be more mature and have bright prospects in the future.

Keywords: *Mapping model, Recurrent Neural Networks, Gated Recurrent Units, Time sequence prediction*

Nomenclature

$\boldsymbol{M}$ – Mass matrix
$\boldsymbol{C}$ – Damping matrix
$\boldsymbol{K}$ – Stiffness matrix
$\boldsymbol{M_i}$ – Mass matrix of boundary i
$\boldsymbol{C_i}$ – Damping matrix of boundary i
$\boldsymbol{K_i}$ – Stiffness matrix of boundary i
$\boldsymbol{F_i}$ – Load matrix of boundary i
$\boldsymbol{x}$ – Time sequence-1
$\boldsymbol{y}$ – Time sequence-2
$\boldsymbol{\varphi_I}$ $\boldsymbol{\varphi_{II}}$ – Transfer matrix
z_i – Coefficient vector
$\psi_i(t)$ – Decision function
$\boldsymbol{h}_t^l$ – Output at layer l in time t
$\boldsymbol{h}_t^{l-1}$ – Output at layer $l-1$ in time t
$\boldsymbol{h}_{t-1}^l$ – Output at layer l in time $t-1$
$\boldsymbol{H}_0$ – Initial state matrix
GRU_t^l – GRU function at layer l in time t
$\boldsymbol{GRU}$ – Multi-layer GRU function

1. Introduction

Inertial Navigation System (INS) must be evaluated on the rocket sled, a high-speed ground test platform, before its service as presented in U.S. standard, which indicates that ground tests get more and more significant for flight vehicle design. Actually, it's hard to simulate the real flight mechanical environment of flight vehicles in the laboratory, because of the boundary conditions. The problem how do we use the data from the ground test for predicting is becoming extremely serious.

Traditionally, the engineers' experiences play a decisive role in how to using data from one boundary condition to another, which will not output an exact predicted result on account of over-test or short-test. Yan [1] established a mapping model of a structure under different boundary conditions from the fundamental dynamical equations. Then a machine learning method, named least square support vector machine (LS-SVM), was introduced to make predicting, and many successful applications in the aerospace field had been finished [2-4]. One shortcoming of this method is that it can only apply to deal with frequency domain signal, and can't work well in the time domain although many improvement

[1] *School of Aerospace, Xi'an Jiaotong University, No.28, Xianning West Road, Xi'an 710049, P.R. of China, zhoujiaming@stu.xjtu.edu.cn, dongll@mail.xjtu.edu.cn, ngli@mail.xjtu.edu.cn*
[2] *State Key Laboratory for Strength and Vibration of Mechanical Structures, Xi'an Jiaotong University, No.28, Xianning West Road, Xi'an 710049, P.R. of China*

 Copyright © 2018 by the author(s)

measures have been adopted, like embedded dimension for increasing the learning features [3]. So it's indispensable to extend this idea to the time domain, and this will be the main issue we discuss in this research.

In the present work, we transform the problem mentioned above into time sequence prediction. As for considering the memory of time series, Recurrent Neural Network (RNN), one of the architectures of deep learning, is applied in this research. Gated Recurrent Unit (GRU) cell is adapted to replace the basic RNN cell to training longer time steps. Then several learning datasets about dynamics response of different structures, like three-degree-of-freedom system and thin plate, are calculated by numerical simulation methods. Finally, two RNN models are training repeatedly using training dataset and output the predicted values on test dataset in order to verify the validity and practicality of this method.

2. Predicted Method

2.1. Mapping Model

In general, the fundamental dynamic equations of a linear structure under different boundary conditions (called as system-1 and system-2) can be formed as

$$\begin{aligned}(\boldsymbol{M}+\boldsymbol{M}_1)\ddot{\boldsymbol{x}}+(\boldsymbol{C}+\boldsymbol{C}_1)\dot{\boldsymbol{x}}+(\boldsymbol{K}+\boldsymbol{K}_1)\boldsymbol{x}&=\boldsymbol{F}_1(t)\\(\boldsymbol{M}+\boldsymbol{M}_2)\ddot{\boldsymbol{y}}+(\boldsymbol{C}+\boldsymbol{C}_2)\dot{\boldsymbol{y}}+(\boldsymbol{K}+\boldsymbol{K}_2)\boldsymbol{y}&=\boldsymbol{F}_2(t)\end{aligned} \tag{1}$$

where the $\boldsymbol{M}$, $\boldsymbol{C}$ and $\boldsymbol{K}$ represent the mass matrix, damping matrix and stiffness matrix of the basic structure, respectively. The basic structure is a structure that does not impose any external constraints. Matric $\boldsymbol{M}_i$, $\boldsymbol{C}_i$ and $\boldsymbol{K}_i$ $(i=1,2)$ are generated by the appearance of structural boundaries. $\boldsymbol{F}_i(t)$ $(i=1,2)$ is the exciting force applied on the structures. $\boldsymbol{x}$, $\dot{\boldsymbol{x}}$, $\ddot{\boldsymbol{x}}$ and $\boldsymbol{y}$, $\dot{\boldsymbol{y}}$, $\ddot{\boldsymbol{y}}$ represent the displacement, velocity and acceleration of system-1 and system-2.

Mathematically, differential equations can be solved by the superposition method in the modal space

$$\begin{aligned}\boldsymbol{x}&=\boldsymbol{\varphi}_I\boldsymbol{z}_1\\\boldsymbol{y}&=\boldsymbol{\varphi}_{II}\boldsymbol{z}_2\end{aligned} \tag{2}$$

In engineering, the type and magnitude of the exciting force of the ground test should be as close as possible to the real state. Therefore, force $\boldsymbol{F_1}(t)$ will be equal to $\boldsymbol{F_2}(t)$. Then the mapping model will be written as follows

$$\psi_i(t)=\frac{\sum_{k=1}^{n}\boldsymbol{\varphi}_{I,ik}\cdot\boldsymbol{z}_{1,k}}{\sum_{k=1}^{n}\boldsymbol{\varphi}_{II,ik}\cdot\boldsymbol{z}_{1,k}} \tag{3}$$

where parameter n is the modal quantity should be considered. i represents a certain position on the structure. $\psi_i(t)$ is a mapping function of position i in the time domain. $\psi(t)$ is the collection of mapping functions, which can describe the relationship between the corresponding positions on system-1 and system-2.

The Eq. 3 is so complicated that it becomes very difficult to solve. In order to work this problem out, the machine learning technology is introduced here. Traditional machine learning methods, such as Artificial Neural Networks (ANNs), Support Vector Machine (SVM) and Decision Tree (DT), have inherent limitations in solving time sequence problems. Because the mapping function is highly dependent on time, which means the function $\psi_i(t)$ is change synchronously over time. Therefore, deep Recurrent Neural Networks (RNNs) is applied in this research.

2.2. Deep Recurrent Neural Networks

RNNs is a class of deep learning [5], which can extract, memory and forget the data information along the time direction. RNNs, once unfolded in time, can be seen as very deep feedforward networks in which the layer share the same weights.

In theory, RNNs are absolutely capable of handling this problem when the data sequences are long. Sadly, in practice, RNNs don't seem to be able to learn this because the backpropagated gradients either grow or shrink at each time step, so over many time steps they typically explode or vanish. The problem was explored in depth by Bengio et al [6], who found some pretty fundamental reasons why it might be difficult. Thankfully, a RNNs model with long short-term memory (LSTM) , which is first introduced by Hochreiter and Schmidhuber [7] in 1977, can be able to deal with long-term dependencies. Since then many LSTM variants have been introduced. Gated Recurrent Unit (GRU)

Copyright © 2018 by the author(s)

introduced by Cho [8] is a slightly more dramatic variation on the LSTM. It combines forget and input gates into a single "update gate". It also merges the cell states and hidden states and makes some other changes. The resulting cell is simpler than standard LSTM cell and will be employed in our research. the GRU architecture we used in this paper is shown in Fig. 1.

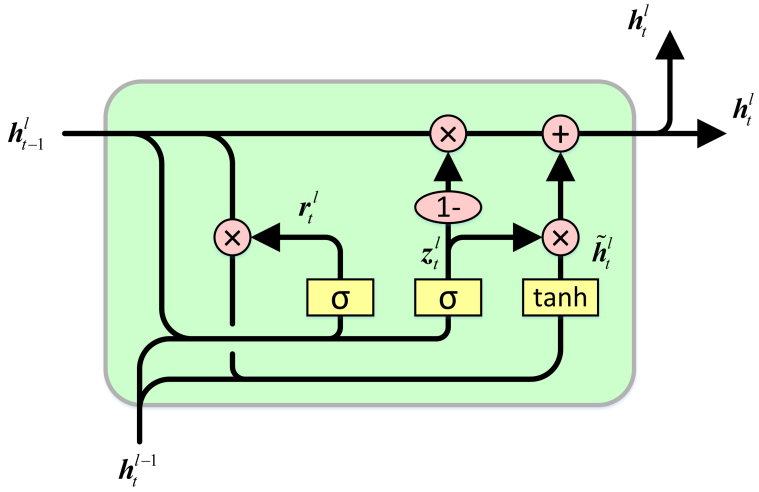

Fig 1. Gated Recurrent Unit (GRU) cell.(This figure is referenced to the website: http://colah.github.io/posts/2015-08-Understanding-LSTMs/)

The vector formulas for this GRU cell forward pass can be written as:

$$\begin{aligned}
\boldsymbol{r}_t^l &= \sigma(\boldsymbol{U}_r^l \boldsymbol{h}_t^{l-1} + \boldsymbol{W}_r^l \boldsymbol{h}_{t-1}^l + \boldsymbol{b}_r^l) \\
\boldsymbol{z}_t^l &= \sigma(\boldsymbol{U}_z^l \boldsymbol{h}_t^{l-1} + \boldsymbol{W}_z^l \boldsymbol{h}_{t-1}^l + \boldsymbol{b}_z^l) \\
\tilde{\boldsymbol{h}}_t^l &= tanh\left(\boldsymbol{U}_h^l \boldsymbol{h}_t^{l-1} + \boldsymbol{W}_h^l \boldsymbol{h}_{t-1}^l + \boldsymbol{b}_h^l\right) \\
\boldsymbol{h}_t^l &= (1 - \boldsymbol{z}_t^l) \odot \boldsymbol{h}_{t-1}^l + \boldsymbol{z}_t^l \odot \tilde{\boldsymbol{h}}_t^l
\end{aligned} \tag{4}$$

where t and l represent the time step and hidden layer number, respectively. N is the number of neurons in LSTM cell and M the dimension of inputs. The dimension of the input weight matrices is $\boldsymbol{U} \in \mathbb{R}^{N \times M}$, and recurrent weight mateix is $\boldsymbol{W} \in \mathbb{R}^{N \times N}$. $\boldsymbol{b}$ represent the bias.

The Eq. 4 can be replaced by a function GRU_t^l. There are two input variables, the output of GRU layer $l-1$ at time t and output of GRU layer l at time $t-1$.

$$\boldsymbol{h}_t^l = GRU_t^l(\boldsymbol{h}_t^{l-1}, \boldsymbol{h}_{t-1}^l) \tag{5}$$

In general, a deep learning model with more hidden layers will perform better. We use three RGU layers in this paper, and the deep RNN model is shown in Fig. 2. In this way, the dynamic response $\boldsymbol{y}$ can be predicted with the formula as follows

$$\boldsymbol{y} = \boldsymbol{GRU}(\boldsymbol{x}, \boldsymbol{H}_0) \tag{6}$$

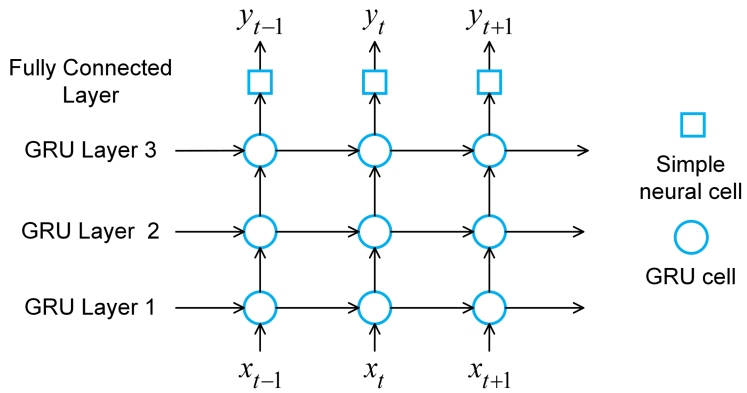

Fig 2. Three-GRU-layer recurrent neural network.

Copyright © 2018 by the author(s)

The Backpropagation Through Time (BPTT) algorithm is applied here to train our deep RNN model. The core of BPTT comes from the backpropagation algorithm by Hinton et al. [9] in 1986. Some detail information about the history of RNN and how to build and train it can refer to the literature [10].

3. Validations

3.1. 3DOF system

We assume two kinds of boundary conditions of 3DOF system as the Fig. 3 shows, and making a prediction of response from system-I to system-II. The mass matrix, damping matrix and stiffness matrix can be calculated as follows

$$\boldsymbol{M}=\begin{bmatrix} m_1 & & \\ & m_2 & \\ & & m_3 \end{bmatrix} \quad \boldsymbol{C}=\begin{bmatrix} c_2 & -c_2 & 0 \\ -c_2 & c_2+c_3 & -c_3 \\ 0 & -c_3 & c_3 \end{bmatrix} \quad \boldsymbol{K}=\begin{bmatrix} k_2 & -k_2 & 0 \\ -k_2 & k_2+k_3 & -k_3 \\ 0 & -k_3 & k_3 \end{bmatrix}$$

$$\begin{aligned} \boldsymbol{M}_1&=\boldsymbol{0}_{3\times3} & \boldsymbol{C}_1&=\begin{bmatrix} c_1 & & \\ & 0 & \\ & & c_4 \end{bmatrix} & \boldsymbol{K}_1&=\begin{bmatrix} k_1 & & \\ & 0 & \\ & & k_4 \end{bmatrix} \\ \boldsymbol{M}_2&=\boldsymbol{0}_{3\times3} & \boldsymbol{C}_2&=\begin{bmatrix} c_1 & & \\ & 0 & \\ & & 0 \end{bmatrix} & \boldsymbol{K}_2&=\begin{bmatrix} k_1 & & \\ & 0 & \\ & & 0 \end{bmatrix} \end{aligned} \tag{7}$$

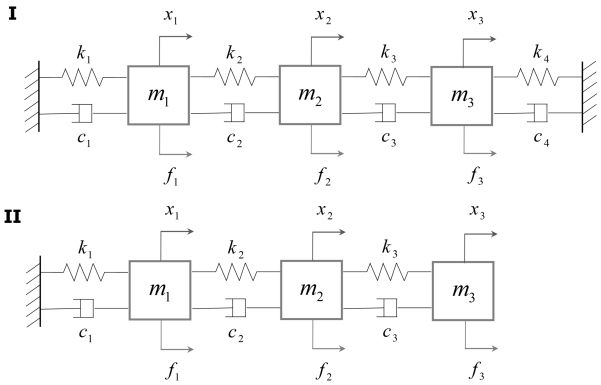

Fig 3. Three-degree-of-freedom system under two boundary conditions.

Where $m_1=5\mathrm{Kg}$, $m_2=3\mathrm{Kg}$, $m_3=1\mathrm{Kg}$, $k_1=k_2=k_3=5000\mathrm{N/m}$ and $c_i=2k_i(i=1,2,3)$. Three different random forces are applied to each mass block, and 10000 sets of displacement response data are generated with sample frequency 100 Hz. The dataset is split into three parts: 80% for training (training dataset), 10% for validation (validation dataset) and 10% for testifying (test dataset). We monitor the loss on the validation dataset (called validation loss) after each training epoch is finished. The initial learning rate is 0.005, and it will be decreased by a factor of 10 if the validation loss is not reduced after 15 epochs. The training program will be stopped if the validation loss is not reduced after 30 epochs. The hidden neurons are 256 per GRU layers. Mean square error (MSE) is selected as the loss function, and deep RNN models are trained by Adam optimizer [11]. The size of each mini batch is 512, which can be set larger but restricted by GPU memory. Finally, the training processing is stopped when the epochs is up to 1522. The loss curve on training dataset and validation dataset is shown in Figure 4.

Copyright © 2018 by the author(s)

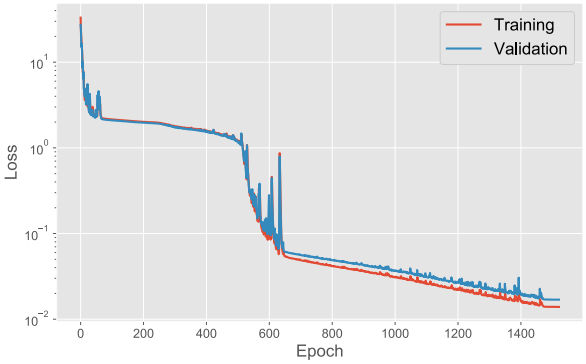

Fig 4. Loss history on training and validation dataset

In order to satisfy real engineering requirements, different levels of noise are added to the simulation data. The operation of adding noise is implemented by Eq. 8. The predicted results of four examples with 20% noise is demonstrated in Fig. 5. The figure shows a good predictive performance of the deep GRU model. In order to quantify the performance of the model, a criterion is introduced here called Root Mean Square Error (RMSE), as shown in Eq. 9. The mean RMSE, the Maximum RMSE, the minimum RMSE and the standard deviation of RMSE is listed in Table 1. It suggests that predictive accuracy decreases as the noise level increases. But the error can still be accepted in engineering applications.

$$\boldsymbol{y}_{noise} = \boldsymbol{y} + noise\_level \times \sqrt{\frac{1}{T}\sum_{t=1}^{T} y_t^2} \times \boldsymbol{rand}(-1, 1) \tag{8}$$

$$\mathrm{RMSE} = \frac{\sqrt{\frac{1}{T}\sum_{t=1}^{T}(y_t - \hat{y}_t)^2}}{\sqrt{\frac{1}{T}\sum_{t=1}^{T} y_t^2}} \times 100\% \tag{9}$$

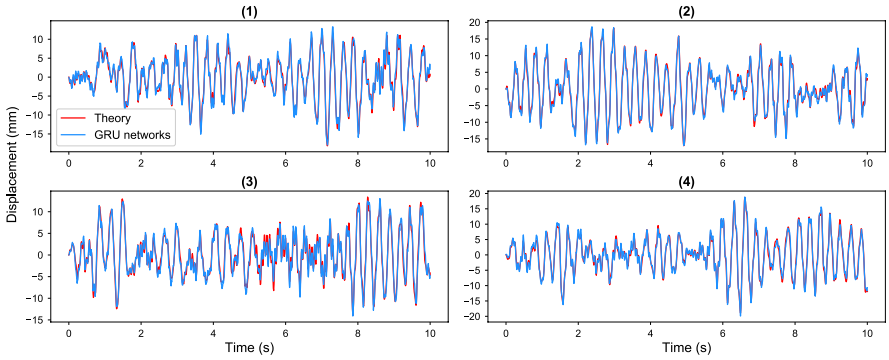

Fig 5. The predicted results of four examples on the test dataset (20% noise).

Table 1. Comparison of errors at different noise levels.

Noise level	Mean RMSE (%)	Max RMSE (%)	Min RMSE (%)	RMSE Std. (%)
0%	1.74	8.61	0.78	0.66
5%	4.03	9.20	2.16	0.80
10%	7.42	13.10	3.98	1.43
15%	10.93	19.24	5.86	2.12
20%	14.48	25.61	7.77	2.82

3.2. Thin plate

In this case, a linear thin plate with two different boundary conditions is studied. One is fixed on one side and the other is fixed on four sides, shown in Fig. 6. The dataset, training strategy and parameters of deep GRU model are the same as Section 3.1. It should be noted that this dataset is non-stationary,

Copyright © 2018 by the author(s)

which can't be predicted by frequency-domain methods. A well-training model was determined when the epochs are up to 1967. The predicted results of four examples with 20% noise is demonstrated in Fig. 7. The mean RMSE, the Maximum RMSE, the minimum RMSE and the standard deviation of RMSE is listed in Table 2. The same results are obtained as the 3DOF structure, but the all errors are larger than the previous ones. The cause of this phenomenon may be that the model does not converge to a sufficiently low error level, or this deep RNN architecture isn't suitable for this structure, or the amount of training data is not large enough. Now, it is hard to make a convincing explanation for this. Subsequently, we compare the predictions with the real acceleration sequence in the frequency domain, as shown in Fig. 8. It can be suggested that the predicted PSD curve has a low degree of coincidence in the low-frequency band while has a high degree of coincidence in the rest. Therefore, we have enough confidence that the results obtained by this prediction method are of engineering practical significance.

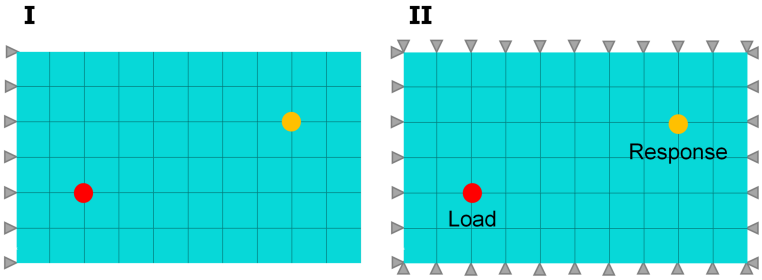

Fig 6. Linear thin plate under two boundary conditions.

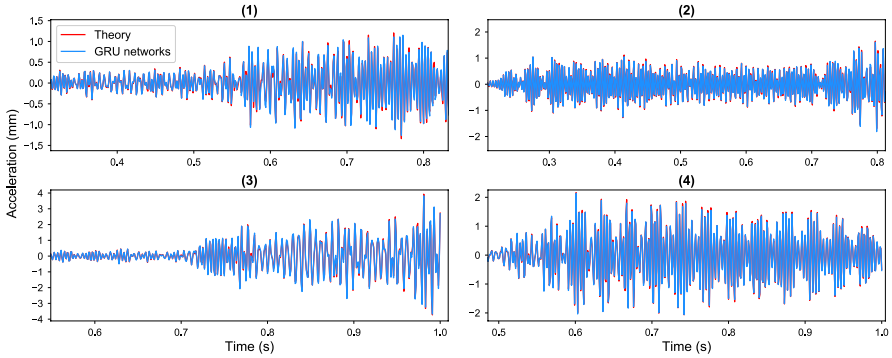

Fig 7. The predicted results of four examples on the test dataset (20% noise).

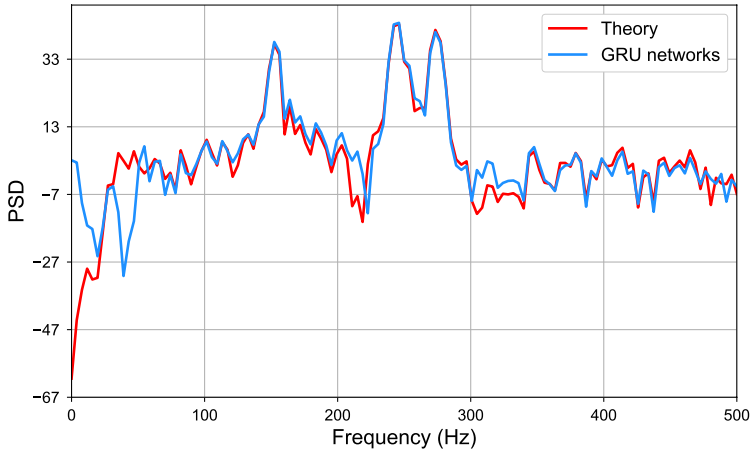

Fig 8. PSD comparison of one example on the test dataset (20% noise).

Table 2. Comparison of errors at different noise levels

Noise level	Mean RMSE (%)	Max RMSE (%)	Min RMSE (%)	RMSE Std. (%)

Copyright © 2018 by the author(s)

0%	7.65	20.31	3.37	2.00
5%	9.87	24.68	4.16	2.50
10%	14.45	48.19	5.29	4.31
15%	19.78	71.56	6.88	6.47
20%	25.40	94.59	8.70	8.67

4. Conclusions

With the fundamental dynamic equations, mapping model in the time domain has been introduced. A new method using Recurrent Neural Network with GRU cell to get an approximate expression of the mapping function under the statistical sense has been proposed. The proposed deep RNN technique is verified on two structures: a three-degree-of-freedom system and a thin plate. In general, the predictive accuracy and difficulty of training mainly depend on the architecture of deep learning model, the complexity of structures and scale of the training dataset. It is worth noting that the machine learning models used in this paper are not surely best and still have some room for optimization, like using auto machine learning [12]. In short, the predicted error of each case satisfies the engineering application requirements and shows that the proposed method have good learning and data prediction ability for stationary and non-stationary time sequence.

References

1. Yan, G., L. Dong, and L. Yu, *A New Dynamical Environment Prediction Method Based on Machine Learning.* CHINESE JOURNAL OF APPLIED MECHANICS, 2013. **30**(1): p. 13-18.
2. Yan, G.R., L.L. Dong, and L.Q. Song, A Method and Its Application for Improving the Validity of Ground Testing of Mechanical Environment's Effects on Aircraft Structure. Equipment Environmental Engineering, 2016.
3. Mao, W., G. Yan, and L. Dong, A novel machine learning based method of combined dynamic environment prediction. Mathematical Problems in Engineering, 2013. 2013.
4. Dong, L., Z. Liu, and G. Yan, Comprehensive Environments Prediction Method for Hypersonic Vehicle. Nearspace Science & Engineering, 2014. 6(4): p. 45-51.
5. LeCun, Y., Y. Bengio, and G. Hinton, Deep learning. nature, 2015. 521(7553): p. 436.
6. Bengio, Y., P. Simard, and P. Frasconi, Learning long-term dependencies with gradient descent is difficult. IEEE transactions on neural networks, 1994. 5(2): p. 157-166.
7. Hochreiter, S. and J. Schmidhuber, Long short-term memory. Neural computation, 1997. 9(8): p. 1735-1780.
8. Cho, K., et al., Learning phrase representations using RNN encoder-decoder for statistical machine translation. arXiv preprint arXiv:1406.1078, 2014.
9. Rumelhart, D.E., G.E. Hinton, and R.J. Williams, Learning representations by back-propagating errors. nature, 1986. 323(6088): p. 533.
10. Salehinejad, H., et al., Recent Advances in Recurrent Neural Networks. arXiv preprint arXiv:1801.01078, 2017.
11. Kingma, D.P. and J. Ba, Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980, 2014.
12. Elsken, T., J.H. Metzen, and F. Hutter, Neural Architecture Search: A Survey. arXiv preprint arXiv:1808.05377, 2018.

 Copyright © 2018 by the author(s)