Available online at www.sciencedirect.com
SCIENCE DIRECT®
Journal of Sound and Vibration 272 (2004) 461–467
JOURNAL OF SOUND AND VIBRATION
www.elsevier.com/locate/jsvi

Letter to the Editor

An application of neural networks to a non-linear dynamics problem

S. Tomasiello*

DiSGG, Faculty of Engineering, University of Basilicata, C. da Macchia Romana, 85100 Potenza, Italy

Received 10 March 2003; accepted 30 June 2003

1. Introduction

A goal of dynamical systems theory is to predict certain aspects of the future behaviour of some evolving systems from its present state. Particularly interesting is determining the possible different final states of a given dynamical system for certain initial conditions.

An ambitious objective is to find, as a parameter is varied, the extension of the intervals where a periodic solution of a given period appears.

A widely used method for finding periodic orbits is Newton's method and variants thereof. There are various programs whose routines are based on this method, for example Ref. [1]. These procedures can be used to locate periodic points of specified period on the time -2Π map of the examined dynamical system and to follow the evolution of the periodic orbit by varying a parameter in a range fixed by the user, but they do not give direct information about the interval where periodic orbits exist. As a first attempt to solve this problem, the possibility of a neural networks approach is discussed here.

Artificial neural networks, by offering a completely different approach to problem-solving, have been successfully applied in solving a wide variety of problems, where other methods failed. There are various application areas, for example classification, pattern recognition, function approximation, control, filtering. For an exhaustive exposition of all the related arguments one can see Ref. [2]. The literature offers several applications of neural networks in engineering and applied mechanics fields, but there is no example referring to the problem examined here [3–6]. In fact, chaos control and system identification problems have a different nature [7–9].

In this paper, a feedforward network trained with the back-propagation algorithm has been used. This structure is by far the most popular.

In designing the network, a particular activation function with a redefined shape has been adopted.

*Tel.: +39-971-205-102; fax: +39-971-205-070.
E-mail address: tomasiello@unibas.it (S. Tomasiello).

0022-460X/$ - see front matter
doi:10.1016/j.jsv.2003.06.016

The parameters of a one-degree-of-freedom Duffing oscillator have been fixed, with exception of the forcing term frequency, and the results of numerical simulations for different values (choosen in a limited range) of the forcing term amplitude are used to train the network.

Even if the response of the Duffing oscillator is quite rich, including subharmonic motions and non-periodic behaviours [10], in order to test the potentialities of a neural network approach, attention has been dedicated to period 3 orbits only, which are usually the most important windows.

By means of the designed network, the prediction of the frequency variation intervals, where the period 3 orbit exists, is satisfactory, provided the input values are within the range used in the training set.

2. Artificial neural networks: an overview

Artificial neural networks are electronic models based on the neural structure of the brain which basically learns from experience. Obviously, the biological brain neurons are complicated and the artificial models try to replicate only their basic elements. A multilayer neural network has an input layer, an output layer and at least one hidden layer. Each layer is composed of some units called neurons or perceptrons, which has led to the name multilayer perceptron network being used. Synapses, the junction between biological neurons, are called connections in the artificial model. If each layer is connected only to the previous layer the system is called feedforward. The units in the input layer serve to distribute the values they receive to the next layer. To each connection a multiplicative (positive or negative) factor is associated that modifies the incoming signal. This factor is called weight or connection strength: it assigns a different importance to the signals transferred through connections from one layer to the next. Each processing element performs a weighted sum of its input; this sum is called net input. For the ith unit, the net input for a pattern p (i.e. the pth example presented to the network) is

$$v_{pi} = \sum_{j=1}^{n} y_{pj} w_{ij},$$

where n is the number of units having connections to the ith unit, y_{pj} is the output of the jth unit and w_{ij} is the weight of the connection from the jth unit to the ith unit. In this sum can be included the bias term, which can be regarded as a weight on a connection that has its input value always equal to one.

The net input constitutes the argument of a transfer function, also called activation function, which generates a result. In order to obtain the desired output, this result can be scaled and added to an offset. The scaling operation can be seen as a change in shape and location of the transfer function. A particularly useful transfer function is the sigmoidal function:

$$f(x) = \frac{1}{1 + \mathrm{e}^{-x}}$$

taking values in the range between zero and one.

In order to obtain the desired output, one needs to find a proper set of weights. The learning rule provides the method for adjusting the weights in the network.

The best known learning rule for multilayer perceptrons is called the generalized delta rule or the back-propagation rule. The back-propagation algorithm uses a gradient descent search method for minimizing an error defined as the mean-square difference between the desired output d_{pk} and the actual output y_{pk}

$$E = \sum_{p=1}^{P} E_p = \frac{1}{2} \sum_{p=1}^{P} \sum_{k=1}^{M} (d_{pk} - y_{pk})^2,$$

where M is the number of output layer nodes. E_p represents a local approximation to the global error surface E. To initiate training one assigns random values to weights. Operating with a pattern mode, as a training pattern is presented to the network, one calculates differences between the actual outputs and the desired outputs and brings those errors back to the hidden layer(s) to calculate the surface gradient there. Then connection weights are adjusted in proportion to the error (i.e., in the direction of the negative of the gradient) by a scaling factor which is called the learning rate whose value is less than unity. So the next input pattern is presented to the network and the weight-update process is repeated. The process continues until all the output layer errors have been reduced to an acceptable value.

The weight changes for a layer, being proportional to the gradient of E_p with respect to the weights for that layer, are written as

$$\Delta_p w_{kj} = -\eta \frac{\partial E_p}{\partial w_{kj}} = \eta \delta_{pk} y_{pj},$$

where obviously $y_{pj} = f_j(v_{pj})$, η is the learning rate parameter and

$$\delta_{pk} = f_k'(v_{pk})(d_{pk} - y_{pk})$$

for output layer units or

$$\delta_{pk} = f_k'(v_{pk}) \sum_{i=1}^{M} \delta_{pi} w_{ik}$$

for hidden layer units.

So, the new weights are

$$w_{kj}(t+1) = w_{kj}(t) + \Delta_p w_{kj}(t+1).$$

A modification of this algorithm is obtained by introducing a momentum term into the weight adaptation equation. This equation is modified so that a portion of the previous weight change influences the current weight adjustment. This allows a low learning coefficient to be used and creates faster learning.

The new weights are calculated as follows:

$$w_{kj}(t+1) = w_{kj}(t) + \eta \delta_{pk} y_{pj} + \alpha \Delta_p w_{kj}(t),$$

where $0 < \alpha < 1$ is the momentum factor and $\Delta_p w_{kj}(t) = w_{kj}(t) - w_{kj}(t-1)$.

3. The Duffing model

Duffing's equation identifies a group of classical non-linear initial-value differential equations

$$\ddot{x} + c\dot{x} - \beta_1 x + \beta_2 x^3 = F \sin \omega\tau, \tag{1}$$

where c is the damping parameter, β_1 and β_2 are the linear and non-linear stiffness parameters respectively. The solution $x(\tau)$ gives the variation of displacement in time τ.

It is not difficult to verify [11] that this equation is the same which can be retrieved using a first approximation discretization, by means of the Galerkin method, a simple structural model such as a simply supported beam with an applied axial load, provided that the beam is considered to be so slender that the shortening of the beam axis cannot be neglected.

In fact, considering such a beam with span L, Young's modulus E, moment of inertia I, mass per unit length m, cross-sectional area A, which is subjected to a compressive load P and to an exciting transverse force $\bar{f}(z,t) = \bar{f}(z) \sin \bar{\omega} t$, the equation of motion can be written as

$$m\frac{\partial^2 u}{\partial t^2} + EI\frac{\partial^4 u}{\partial z^4} + P\frac{\partial^2 u}{\partial z^2} - \frac{EA}{2L}\left(\int_0^L \left(\frac{\partial u}{\partial z}\right)^2 \mathrm{d}z\right)\frac{\partial^2 u}{\partial z^2} = \bar{f}(z) \sin \bar{\omega} t. \tag{2}$$

This equation in terms of dimensionless variables becomes

$$\frac{\partial^2 v}{\partial \tau^2} + \frac{\partial^4 v}{\partial \zeta^4} + \sigma\frac{\partial^2 v}{\partial \zeta^2} - \kappa\left(\int_0^1 \left(\frac{\partial v}{\partial \zeta}\right)^2 \mathrm{d}\zeta\right)\frac{\partial^2 v}{\partial \zeta^2} = F(\zeta) \sin \omega\tau, \tag{3}$$

where

$$v = \frac{u}{L}, \quad \zeta = \frac{z}{L}, \quad \tau = \sqrt{\frac{EI}{m}}\,\frac{t}{L^2}, \quad \sigma = \frac{PL^2}{EI}$$

$$\kappa = \frac{AL^2}{2I}, \quad \omega = \sqrt{\frac{m}{EI}}\,L^2\bar{\omega}, \quad F(\zeta) = \frac{\bar{f}(\zeta)L^3}{EI}.$$

By writing the solution in first approximation as follows:

$$v(\tau,\zeta) = x(\tau) \sin \pi\zeta$$

and continuing as usual in the Galerkin method, Eq. (1) is obtained.

It is observed that the negative sign of the linear stiffness parameter β_1 indicates that for positive values the axial load is greater than the critical load. So the beam can oscillate around different equilibrium points alternatively and not just around the only equilibrium point existing for the sub-critical case $\beta_1 < 0$.

4. Training and testing of the neural network

A neural network may be considered as a non-linear input–output mapping. If the network produces a correct (or nearly so) input–output mapping even when the input is slightly different from the examples used to train the network, the network is said to generalize well.

The number of input–output pair examples in the training set is important to reach a good generalization: too many input–output examples cause a phenomenon known as overtraining. When the network is overtrained, it looses the ability to generalize between similar input–output patterns.

In preparing the data set (training and testing data) for the examined problem, the attention has been restricted to values corresponding to an amplitude of the forcing term between 3 and 9. In this range, bifurcation diagrams reveal two intervals relative to period 3 orbits. So, the number of output nodes is fixed at four: the minimum and the maximum frequency of the first interval are here indicated as *output 1* and *output 2* respectively, whereas the analogous values referring to the second interval are indicated as *output 3* and *output 4* respectively.

In addition, one hidden layer with six nodes has been considered.

The problem parameters have been fixed as follows:

$$\beta_1 = \beta_2 = 1, \quad c = 0.3.$$

Because the range of variation of the input parameter (i.e., the amplitude of the forcing term) is quite small, a reduced data set is sufficient to obtain a network with good predictive capabilities, provided that it is well designed.

To reproduce the observed data, a non-symmetric activation function (output between 0 and a certain value a) has been chosen as the hyperbolic tangent:

$$f(v) = a\tanh(bv).$$

The assumption $a = 1.6$ is necessary to contain the target values (desired output) into the range of the sigmoid function. Besides, by desiring to have at the origin, the slope of the activation function close to the unity, $b = 0.6$ remains fixed.

In order to increase the speed of convergence, $\eta = 0.2$ and $\alpha = 0.9$ have been fixed.

Figs. 1–4 show the good generalization of the designed network. The curves depicted in these figures represent the non-linear input–output mapping computed by the network as a result of learning the points labelled training data. In particular, Figs. 1 and 3 refer to the minimum frequency value for a single interval, whereas Figs. 2 and 4 refer to the maximum value. Fig. 5 gives information about the errors between desired and actual outputs.

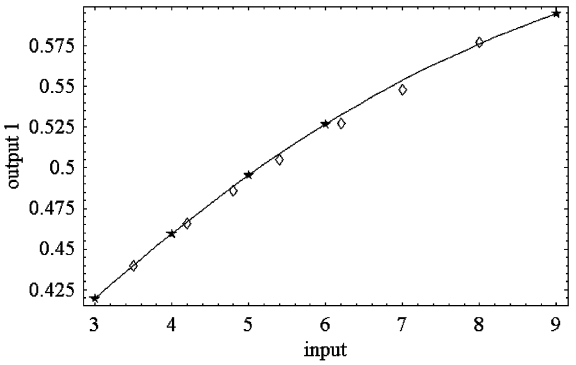

Fig. 1. The minimum frequency in the first interval: ★ training data, ◇ testing data.

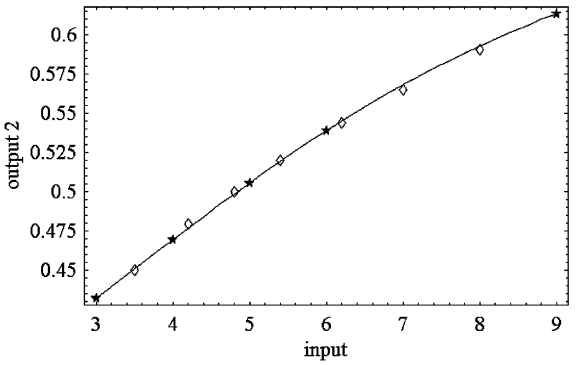

Fig. 2. The maximum frequency in the first interval: ★ training data, ◇ testing data.

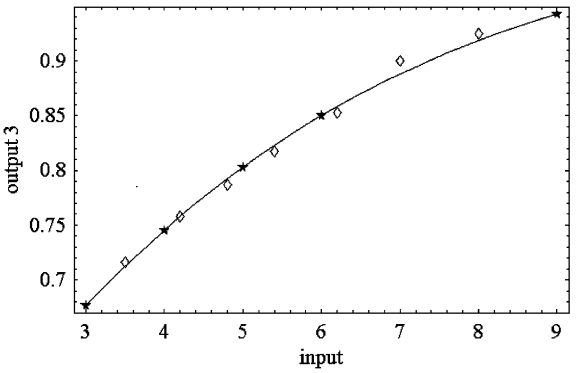

Fig. 3. The minimum frequency in the second interval: ★ training data, ◇ testing data.

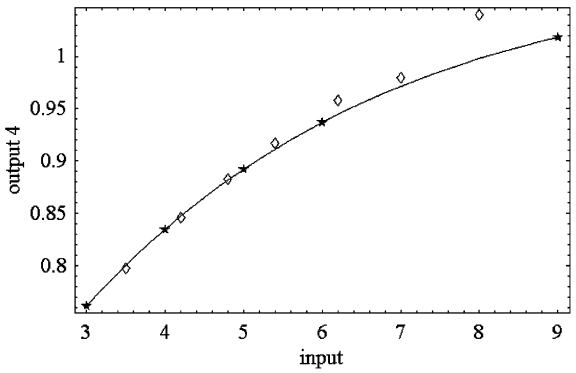

Fig. 4. The maximum frequency in the second interval: ★ training data, ◇ testing data.

5. Conclusions

This paper represents a first exploration of the possibility of using the neural networks tool for detecting the intervals where a periodic solution of assigned period exists.

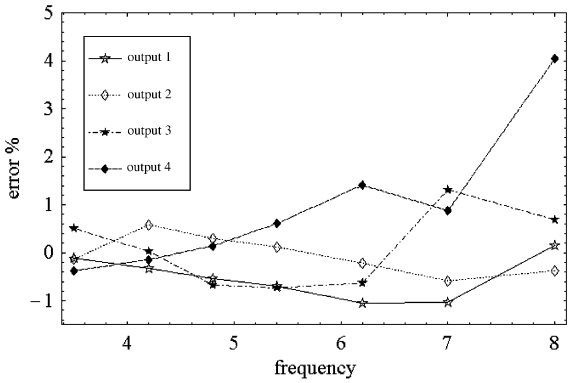

Fig. 5. Error distribution of the frequency.

A feedforward network trained with the back-propagation algorithm has been used. The type, shape and location of the activation function have been chosen with care to produce the desired output by avoiding problems such as saturation of some nodes. Numerical results show the good performance of the network, encouraging further development in order to enlarge the range of analysis.

References

[1] H.E. Nusse, J.A. Yorke, *Dynamics: Numerical Explorations*, Springer, Berlin, 1998.
[2] S. Haykin, *Neural Networks: A Comprehensive Foundation*, Prentice Hall, New York, 1999.
[3] I. Takahashi, Identification for critical flutter load and boundary conditions of a beam using neural networks, *Journal of Sound and Vibration* 228 (4) (1999) 857–870.
[4] N. Huber, Ch. Tsakmakis, A neural network tool for identifying the material parameters of a finite deformation viscoplasticity model with static recovery, *Computer Methods in Applied Mechanics and Engineering* 191 (2001) 353–384.
[5] J. Kudva, N. Munir, P.W. Tan, Damage detection in smart structures using neural networks and finite-element analysis, *Smart Materials and Structures* 1 (1992) 108–120.
[6] K. Worden, A.D. Ball, G.R. Tomlinson, Fault location in structures using neural networks, *Proceedings of 11th International Modal Analysis Conference*, Vol. 1, Orlando, FL, 1993, pp. 47–55.
[7] C.T. Leondes (Ed.), *Control and Dynamic Systems*, *Neural Network Systems Techniques and Applications*, Vol. 7, Academic Press, New York, 1998.
[8] K. Hirasawa, X. Wang, J. Murata, J. Hu, C. Jin, Universal learning network and its application to chaos control, *Neural Networks* 13 (2000) 239–253.
[9] J.W. Howse, C.T. Abdallah, G.L. Heileman, A synthesis of gradient and Hamiltonian dynamics applied to learning in neural networks, UNM Technical Report (EECE95–003), 1995.
[10] K.L. Janicki, W. Szemplinska-Stupnicka, Subharmonic resonances and criteria for escape and chaos in a driven oscillator, *Journal of Sound and Vibration* 180 (2) (1995) 253–269.
[11] F.C. Moon, P.J. Holmes, A magnetoelastic strange attractor, *Journal of Sound and Vibration* 65 (2) (1979) 275–296.