automatica

The Journal of IFAC the International Federation of Automatic Control

Please reply to:
Professor Hulbert Kwakernaak
Faculty of Applied Mathematics
University of Twente
P.O. Box 217, 7500 AE Enschede
The Netherlands

TW93/SBA/351/mq
April 14, 1993

Dr. E.M. Nebot
PLAPIQUI CONICET
Universidad Nacional del Sur
12 de octubre 1842
8000 Bahia Blanca
ARGENTINA

IFAC Congress 1993

Dear Dr. Nebot,

You have probably been informed that your paper "Identification of a Flexible Manipulator Using Neural Networks," with G. Sentoni and F. Masson, which you submitted for the 1993 IFAC World Congress, has been accepted for presentation at the Congress. I am writing you to let you know that your paper was favorably reviewed, and has been recommended for review for possible publication in the IFAC Journal AUTOMATICA. As you know, IFAC holds the copyright for the publication of papers at IFAC events in one of the two IFAC Journals until it is released.

Before a decision is taken about publication in AUTOMATICA, all papers undergo a comprehensive review procedure. Because the format and standard for Conference and Congress papers are different from those for an archive journal such as AUTOMATICA normally extensive rewriting and revision is needed if the paper is published.

To hasten this process, I am writing you now with the request to send me the final version of the paper as it is prepared for the Congress. Please send me five copies, with a covering letter that refers to this letter. If I do not receive these copies, I assume that you do not wish the paper to be considered for publication in AUTOMATICA under any circumstances. You may also send me copies of a version that has already been prepared for journal publication.

Telephone (+31)(53) 893457 Telex 44200 Telefax (+31)(53) 340733
E-mail Automatica@math.utwente.nl
Street address: Drienerlolaan 5, 7522 NB Enschede, The Netherlands

Published by
PERGAMON PRESS
Oxford • New York • Seoul • Tokyo

After receipt of the papers they will be sent out for preliminary evaluation. Please do not be disappointed if the result is not encouraging. If the paper emerges from the provisional review procedure with a positive recommendation most likely you will be requested to revise the paper extensively.

I am sending this letter to one of the authors of the paper only. Please inform your co-authors, if any.

I hope to receive your paper. Thank you very much.

Sincerely,

Huibert Kwakernaak, Deputy Editor-in-Chief

Cc: G. Axelby, Editor-in-Chief
R.E. Skelton

Identification of a Flexible manipulator using Neural Networks.

Eduardo M. Nebot, Guillermo Sentoni, Favio Masson.

*PLAPIQUI CONICET, Universidad Nacional del Sur

12 de octubre 1842, 8000 Bahia Blanca

E.mail: CONEBOT@CRIBA.EDU.AR

Abstract.

The need of model identification and / or adaptive control of flexible structures arise because of the lack of knowledge of the values of important internal parameters and their variation under different load conditions. This is also true for rigid systems. Furthermore, under certain tasks a more accurate model of the robot can be obtained considering the structural flexibility or the joint elasticity.

Due to the infinite dimensional nature of the flexible robot model involved, off line methods has been studied to determine set of parameter that fit the system under certain load conditions.

In this work we present a neural network approach to obtain the model of a flexible manipulator that is able to capture its high order dynamics. A technique using singular value decomposition is presented to optimize the network with respect to the number of inputs and nodes used. Finally an approach to accelerate the convergence of the standard Backpropagation algorithm is also presented.

Introduction

The modelling aspect of flexible structures has been studied by a large number of researchers. Generic models of these structures are investigated from the infinite discrete spectrum viewpoint for distributed parameter systems , Banks et. al. (1986). These models need to be converted to finite dimensional state space models or transfer function models for the purpose of control synthesis. Other approaches, Usoro (1986), make use of the finite element method for the mathematical modelling of lightweight flexible manipulators. Recursive algorithms have been also applied to these systems in on line applications, Rovner (1988), Nebot (1989). In these works it can be seen the complexity that high order system introduce to on line identification. This work presents an innovative approach to identify the flexible manipulator using neural networks.

A neural network system is a highly parallel dynamic system. The engineering community are involved in applying this massively parallel distributed information technique to image processing, speech recognition, control and more recently to system identification.

The most important factor in employing neural networks for any application include the choice of the architecture for the network and the selection of the appropriate learning algorithm. The reader can have a review of several architectures and teaching algorithm in the article by Lipmann (1987).

The use of neural network in the context of identification is supported by the result of Cybenko (1989) and Funahashi (1989) who have proved that any continuos function can be uniformly approximated by a neural network model with only one internal layer.

Neural Network architecture.

A neural network model is defined by three major components:

1. Network topology.
2. Computational characteristics of its elements.
3. Training algorithms.

In our experiments we have used the multilayer perceptron network which have the structure shown in figure 1.

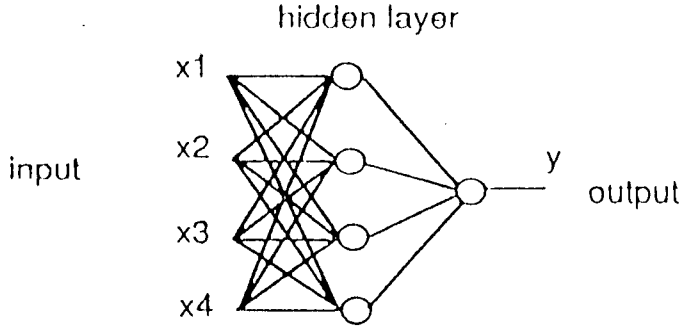

Figure 1. General Neural Network architecture.

The multilayer network is made up of one or more hidden layers between input and output layers. The layers are composed of computing units called nodes which are interconnected together as indicated in figure 1. The behavior of the network is set according to the value of the weights that joined the nodes. The inputs are propagated through the network and the output of each node is evaluated according to

$$x_i^k = f\left(\sum_{j=1}^{n_{k-1}} w_{ij}^k x_j^{k-1} + \theta_i^k\right)_{\substack{i=1..n_k \\ k=1..l}} \qquad (1)$$

where

x_i^k: output of the ith neuron of the k layer.

w_{ij}^k: weight value of connection between jth neuron of (k-1) layer and ith neuron of kth layer.

θ_i^k: Threshold of the ith neuron of the kth layer.

lth layer : output layer, 0th layer : input layer.

Being f (.) the node activation function, its argument can be written as follows:

$$u_i^k = \sum_{j=1}^{n_{k-1}} w_{ij}^k x_j^{k-1} + \theta_i^k \qquad (2)$$

The function f (.) must be diferentiable and must have a strictly positive first derivative. It is often chosen to be:

$$f(u) = \frac{1}{1+e^{-u}} \qquad (3)$$

The most popular and successful learning algorithm used to train multilayer neural networks in areas such as control and system modelling is currently the backpropagation routine, Rumelhart D. E., McClelland (1986). It is basically a gradient algorithm designed to minimize the mean square error between the actual output of the network and the desired output. The teaching algorithm is designed as an optimization problem :

$$\min_{w} \sum_{i=1}^{T} (d^n(i) - y^n(i))^2 \qquad (4)$$

where $y^n(i)$ is the nth neural network model output for the ith training sample, $d^n(i)$ is the desired corresponding output and T is the total number of training samples. The weights w must be chosen in order to minimize the difference between the desired output and the network output.

The backpropagation algorithm adjust the weight according with the following equations:

$$w_{ij}(t+1) = w_{ij}(t) + \eta \delta_j x_i + \alpha (w_{ij}(t) - w_{ij}(t-1)) \qquad 0 < \alpha, \eta < 1 \qquad (5)$$

where

$$\delta_j = y_j(1 - y_j)(d_j - y_j) \quad \text{if j is the output node}$$

$$\delta_j = x_j(1 - x_j) \sum_k \delta_k w_{jk}, \quad \text{if j is an internal hidden node} \qquad (6)$$

η and α are variable factors that can be adjusted to speed up the convergence of the algorithm.

Scaling Factor.

The neural network maps function of a non-bounded N dimensional input space to a bounded M dimensional output space. This mapping is due to the activation function. The sigmoid function used in this case maps an unbounded input to an output range [0 1]. The latter implies that this activation function could be used with unbounded inputs. However this use can be the cause of an ill conditioned weighting matrix. To overcome this problem it is necessary to scale the inputs between [0 1]. In practice scaling the inputs / outputs between [0.2 0.8] make the network work closer to the linear range of the activation function.

Optimization of the network.

Determining the optimum size of the network is a real involved task. Quite a few works present results in this area. Among them, Cybenko (1986) has shown that only one internal layer is necessary to approximate a continuos function. Mirchandani and Cao (1989), give some criteria to determine the maximum number of nodes in a given layer.

In this work we make use of Singular Value Decomposition, SVD, for the next three main objectives:

1. Determine redundancy in the inputs.
2. Determine an excessive number of neurons in a given layer.
3. Improve the final convergence of the backpropagation algorithm.

Singular Value Decomposition applied to Neural Networks.

It can be shown that given an M x N matrix A, where $M \geq N$, it can be written as a product of an M x N column orthogonal matrix U, an N x N diagonal matrix Σ with positive or zero elements, and the transpose of an N x N orthogonal matrix V. That is

$$A = U \Sigma V^T \tag{7}$$

with

$$A[M * N], U[M * N], \Sigma[N * N], V[N * N]$$

$$\Sigma = \begin{bmatrix} \sigma_1 & 0 & \dots & 0 \\ 0 & \sigma_2 & \dots & 0 \\ . & \dots & . & 0 \\ 0 & 0 & 0 & \sigma_N \end{bmatrix} \tag{8}$$

where U and V are orthogonal meaning that

$$U^T U = V^T V = I \tag{9}$$

In the case A is a square matrix, its inverse can be expressed as a function of U, V and Σ as follows:

$$A^{-1} = V \left[\text{diag} \frac{1}{\sigma_j} \right] U^T = V \Sigma^{-1} U^T \tag{10}$$

U and V are square matrices and their inverse are just their transpose. The only numerical problem that can arise is when one or more σ_i are very close

to zero. In that case the SVD is giving a clear diagnostic of the singularity of the matrix A. Usually the ratio of the maximum / minimum σ_i is defined as the condition number of a matrix. Higher values for this number imply ill conditioned systems.

If a matrix A has a rank k < N, it will imply that $\sigma_{k+i} = 0$, i = 0..N-k. In practice σ_{k+i} will be significant smaller than σ_i, i≤k. That means that there exists N-k linearly dependent columns in A. Being B the matrix without the extra column, the matrices A and B span the same subspace.

In relation with Neural Networks we can analyze the subspace spanned by the set of chosen inputs using SVD. Given the following two sets of inputs α_i, β_i, i=1..N, being N the number of trains

$$\alpha_1 = \begin{bmatrix} u_k & u_{k-1} & .. & u_{k-l} & y_{k-1} & \cdots & y_{k-j} \end{bmatrix} \qquad (11)$$

$$\beta_1 = \begin{bmatrix} u_k & u_{k-1} & .. & u_{k-h} & y_{k-1} & \cdots & y_{k-r} \end{bmatrix} \qquad (12)$$

being u and y the input and outputs of the network.

It is possible to determine the condition number of the input matrices and determine the redundancy in each set , if any. Furthermore, analyzing the values of the σ_i, it can be possible to adjust the number of input the Network should have in order to avoid high condition numbers.

This method was also applied to evaluate the number of nodes in a given layer. This study is done analyzing the propagated nodes output and obtaining the SVD of the generated matrix. Values of σ_i close to zero imply excessive number of neurons in this layer. The methodology has been applied in our experimentation with remarkable results.

Finally we make use of SVD to accelerate the convergence of the backpropagation algorithm. Working with a two layer network the following definition can be made:

$$\begin{aligned} \underline{u}_i &= \sum_{j=1}^{N_1} \underline{w}_{ij} x_j + \underline{\theta}_i \\ a_i &= f(\underline{u}_i) \\ \overline{u}_i &= \sum_{j=1}^{P} \overline{w}_{ij}\, a_j + \overline{\theta}_i = \sum_{j=1}^{P+1} \overline{w}_{ij}\, a_j \\ y_i &= f(\overline{u}_i) \quad i = 1...N \end{aligned} \qquad (13)$$

where $(\underline{w}_{ij}, \underline{u}_i, \underline{\theta}_i)$ and $(\overline{w}_{ij}, \overline{u}_i, \overline{\theta}_i)$ represent the parameter of the lower and upper layer respectively. At a later stage of the learning process we have that $f^{-1}(d_i^m) = b_i^m$ can be very close to $\overline{u}_i^m$. Then the following approximation can be written:

$$d_i^m - y_i^m = f(b_i^m) - f(\overline{u}_i^m) \approx f'(\overline{u}_i^m)(b_i^m - \overline{u}_i^m) \tag{14}$$

The mean square error can be reformulated as follows:

$$E_i = \sum_{m=1}^{M} f'(\overline{u}_i^m)^2 (b_i^m - \sum_{q=1}^{P+1} \overline{w}_{iq}^m a_q^m)^2 \tag{15}$$

Now we can form an M * (P+1) matrix A whose rows are the vectors $\{a^m\}$: $A = [a^1 \ a^2 \ \dots \ a^M]^T$, and a vector $b = [b_i^1 \ b_i^2 \ \dots \ b_i^M]^T$. Minimizing E_i can be done solving the following equation:

$$A w = b \tag{16}$$

where the unknown w is a P+1 dimensional vector whose components are $(\overline{w}_{iq}, \ q = 1,2,\dots,P+1)$. The matrix A is non square, and usually ill conditioned. Kung S, Hwang J (1988) propose a method based on algebraic projection analysis to solve for w. In this work we propose to work with SVD to overcome the numerical problem that arise from the high condition number of matrix A. If b is not in the range of of the singular matrix A the equation (16) will not have an exact solution, but an approximation of a vector w can be found solving:

$$w = V\left[\operatorname{diag}\left(\frac{1}{\sigma_j}\right)\right]U^T b \tag{17}$$

making $1/\sigma_j = 0$ for every σ_j very close to zero.

With this we have a robust numerical solution of the vector w that minimize $|Aw-b|$. The method was also applied in the experiments to show the improvement of the estimated model when the backpropagation algorithm has reached a local minimum.

Description of the experimental system.

In this work we have used experimental data obtained from a single link flexible manipulator. It was constructed from two thin cantilevered beams interconnected along their lengths by eight bridges that attach to their side plates through flexible brass hinges. With this structure there is flexibility in

the horizontal plane while maintaining vertical and torsional stiffness. The actuator consist of a DC motor mounted at the hub. The measurements available are : collocated hub position and velocity and noncollocated tip position that is generated through a ccd camera mounted at the hub. The experimental set-up is fully detailed in Nebot (1989).

The collocated transfer function can be approximated by the following equation:

$$\frac{\Theta(s)}{T(s)} = \frac{1}{I_T s^2} \prod_{i=1}^{\infty} \frac{s^2 + 2\zeta_i \Omega_i s + \Omega_i^2}{s^2 + 2\zeta_i \omega_i s + \omega_i^2} \qquad (18)$$

where Θ is the hub angle, T is the torque input and I_T is the total moment of inertia. From the transfer function one notes that ω , Ω and ζ are the frequencies of oscillation of the poles and zeros and damping coefficients respectively. I_T is the total moment of inertia. With high order modes the poles are very close to the zeros and they are very difficult to discriminate.

The non collocated transfer function have similar form, except for the numerator that have zeros with positive real part.

In this experiment the manipulator was excited with a composition of three chirp signal varying around the main three modes of the system. The neural network was trained with this signal. A second excitation was also used for comparison purposes. A linear chirp was generated varying from 0.1 to 10 hz and was input to the manipulator and to the network.

The size of the training samples was 512 spaced 25 msec each.

Results.

The system was trained with composed chirp signal which makes the hub position of the manipulator behave as shown in Figure 2. After applying SVD the following input set was chosen:

$$\text{Net Inp} = \begin{bmatrix} y_{k-3} & y_{k-2} & y_{k-1} & u_{k-1} & u_k \end{bmatrix}$$

The number of neurons of the hidden layer was also optimized following the methodology presented before. The minimum number of neurons were 3. Figure 3 shows the behavior of the estimated model and in figure 4 the relative porcentual error is presented. It can be appreciated that this error is

smaller than 10 % during most of the training samples. What is also important in the comparison in the frequency domain. Figure 5 and 6 shows the frequency response of the system and model respectively. It can be seen that the neural network model is reproducing with acceptable precision the frequency response of the manipulator. Figure 7 present the response of the hub position under a linear chirp varying in frequency from 0.1 hz to 10 hz. Although the network was not trained with this signal it can follow the system as shown in figure 8. The relative porcentual error is below 10 % as can be appreciated from figure 9.

In order to show the use of SVD to improve the convergence of the algorithm, a network with 7 nodes in the hidden layer was trained with 6700 iterations. The behavior of the model is shown in figure 10 with its error in Figure 11. The same network was trained with 21700 iterations giving the results shown in Figure 12 and 13. It can be appreciated that the error is considerable smaller than the one obtained with 6700 iterations.

After identifying the model with 6700 iteration the SVD was applied to find the weights solving the equation Aw=b. With this extremely fast procedure we obtained the results shown in figure 14 and 15 that indeed are comparable with the one obtained after 21700 iteration.

Finally the results corresponding to the identification of the tip position behavior is presented. Figure 16 is the tip response of the manipulator. The performance of the neural model is shown in figure 17 with the relative percentage error in Figure 18. Figure 19 - 21 show the same experiments but with the velocity information.

Conclusion.

We have presented a Neural Network approach to identify the collocated and noncollocated transfer function of a flexible manipulator. It has been shown with experimental results that the behavior of the model is reasonable good in the time and frequency domains. We have also presented techniques based on Singular Value decomposition to reduce the Neural Network computation units and to accelerate the convergence of the standard

Backpropagation algorithm. Future works will address the problem of controller design using these techniques.

References.

Banks H. J., Crowley J. H, Rosen I. G., "Methods for identification of material parameters in distributed models for flexible structures", Mat. Aplic. Comp. 5 1986, pp 139-168.

Cybenko G., "Approximation by superpositions of sigmoidal functions", Matematics of control signal and systems, 1989, pp 303-314.

Funahashi K., "On the approximate realization of continuous mapping by neural networks", Mathematics of control signal and systems, 1989, pp 183-192.

Ham M. F. , Greely S. U. , "Active damping control design for mast flight systems", ACC, Minessota, June 1987.

Kunn S. Y., Hwang J. N., "An algebraic Projection Analysis for optimal hidden units size and learning rates in back-propagation learning, 1988, 1363-1370, Caltech Libraries.

Lippman R. P. , "An introduction to computing with neural nets", IEEE ASSP Magazine, 1987.

Mirchandani G, Cao W, "On hidden nodes for neural nets ", IEEE transaction on Circuits and Systems, Vol 36, No 5, 1989.

Nebot E. M., Lee G. K. F., and Karim A., "Parameter Identification for a single link flexible manipulator". Proc. of IASTED Int'l Conference on Expert Systems, Zurich, Switzerland, 1989.

Rovner D. M. and G. F Franklin, "Experiments in load Adaptive control of a very flexible one link manipulator", Automatica, Vol 24, No 4, pp 541-548, 1988.

Rumelhart D. E., McClelland J. L., Parallel Distributed Processing: Exploration in the Microstructure of Cognition, Vol. 1 Foundations, Cambridge, MA:MIT Press, 1986.

Usoro P. B., Nadira R., Mahil S. S., "A finite element lagrange approach to model lightweight flexible manipulators", Journal of dynamic System, Measurements and control, Sept 1986, Vol 108

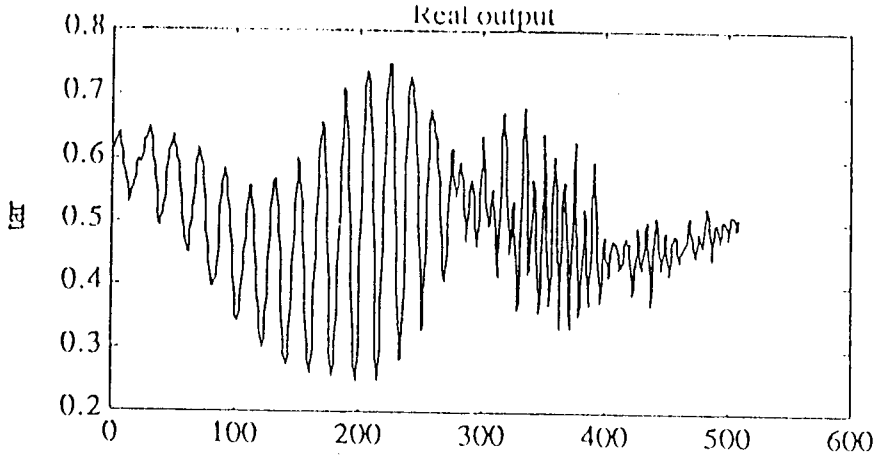

Figure 2.

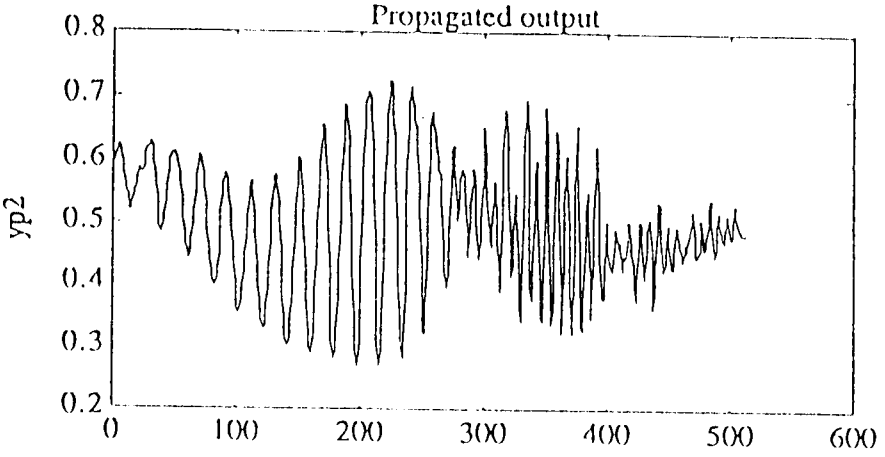

Figure 3.

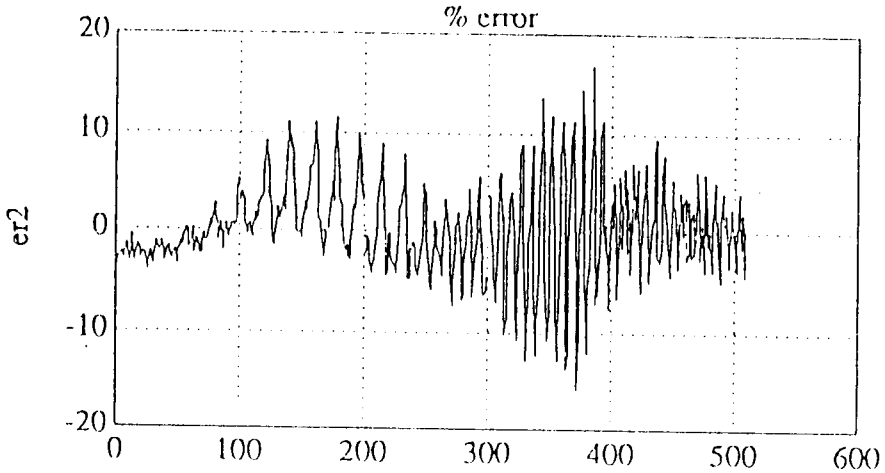

Figure 4.

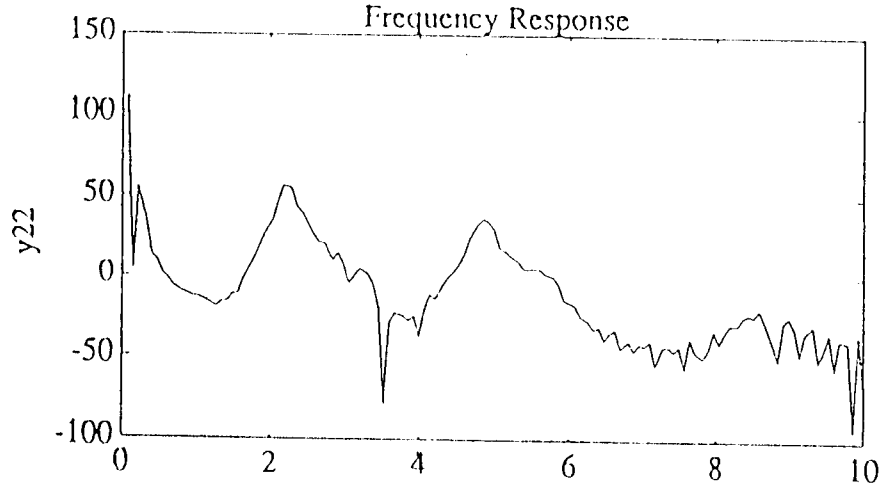

Figure 5.

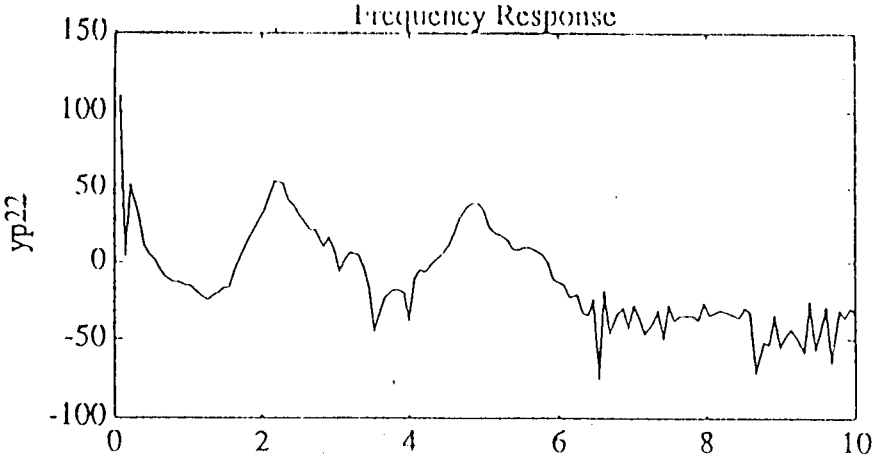

Figure 6.

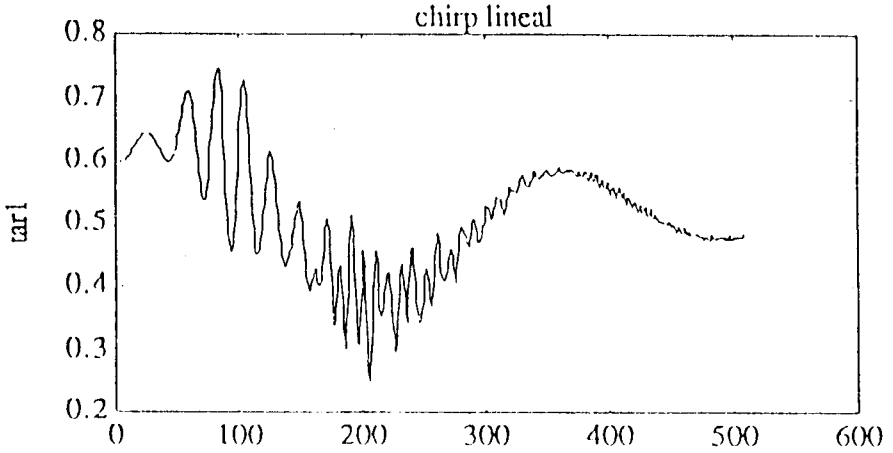

Figure 7.

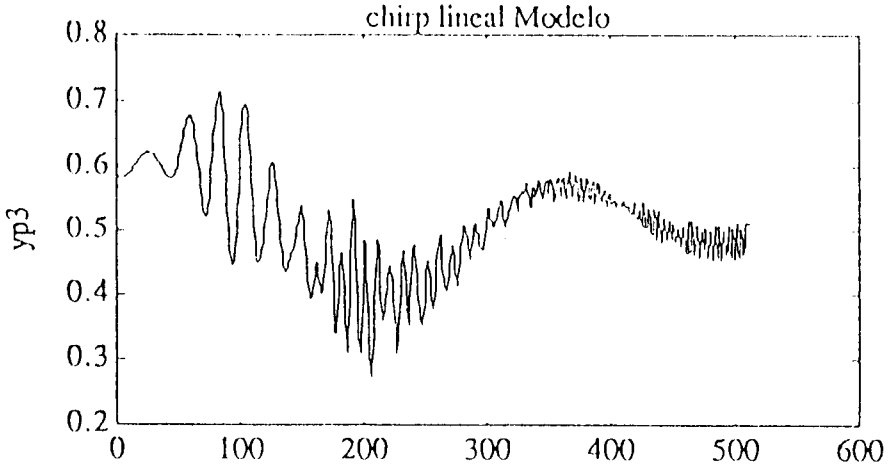

Figure 8.

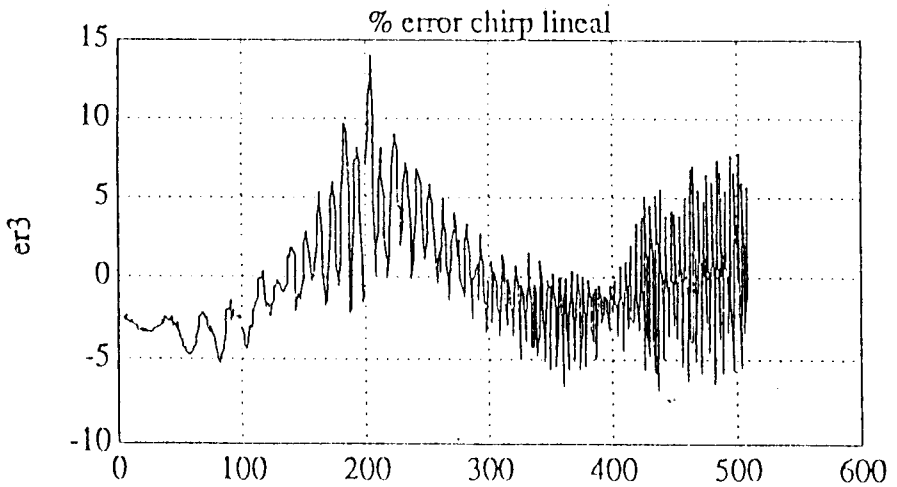

Figure 9.

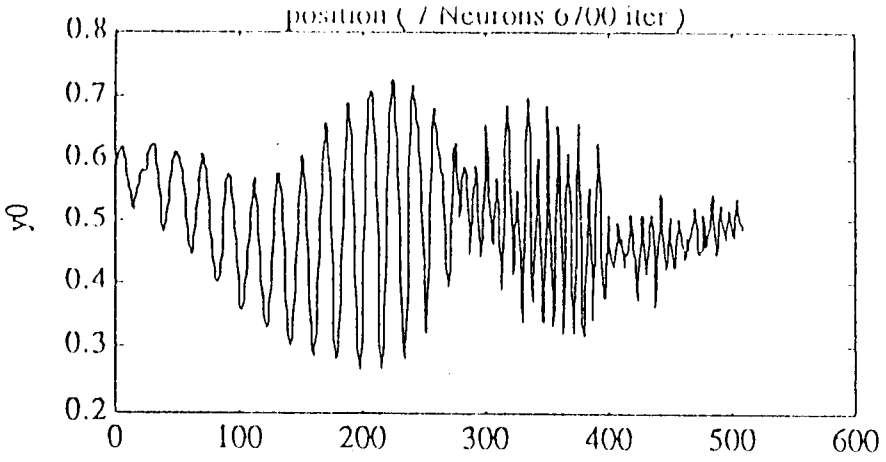

Figure 10.

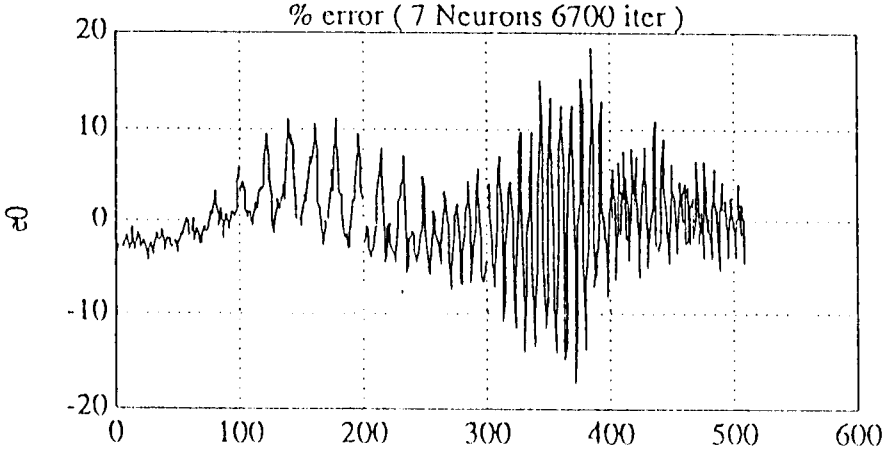

Figure 11.

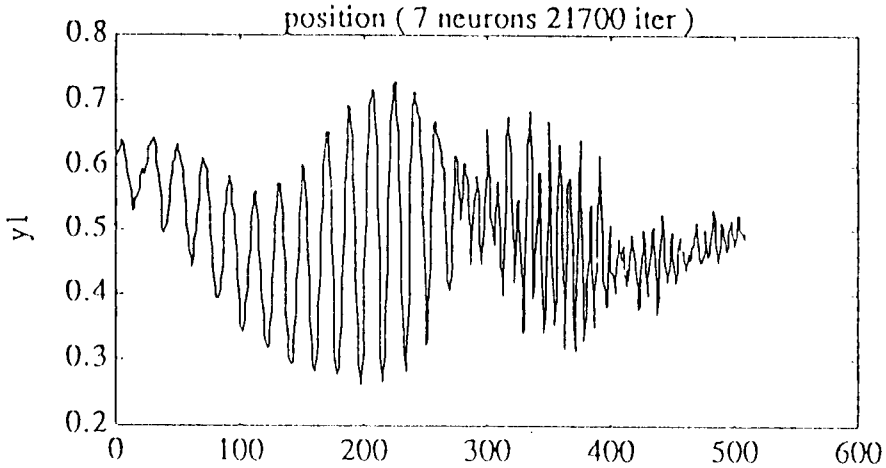

Figure 12.

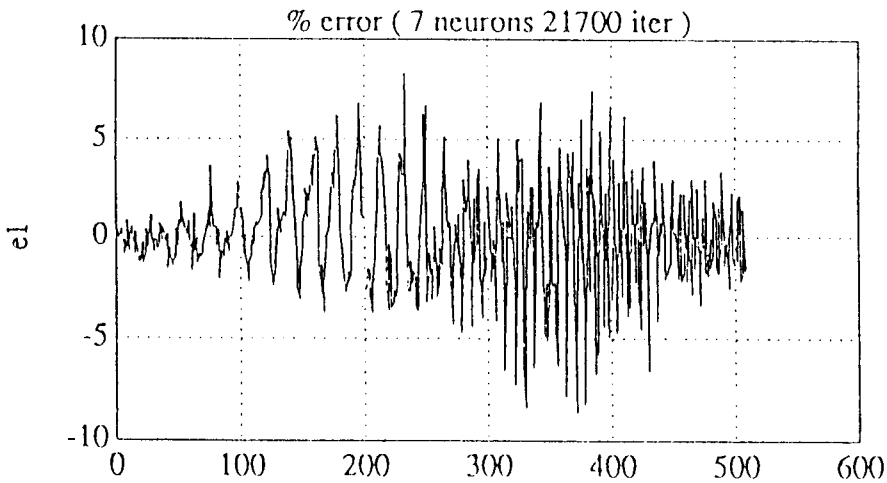

Figure 13.

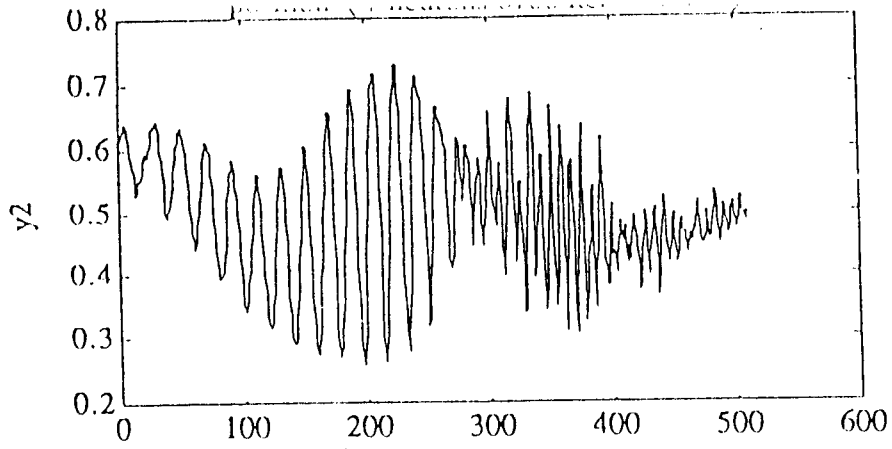

Figure 14.

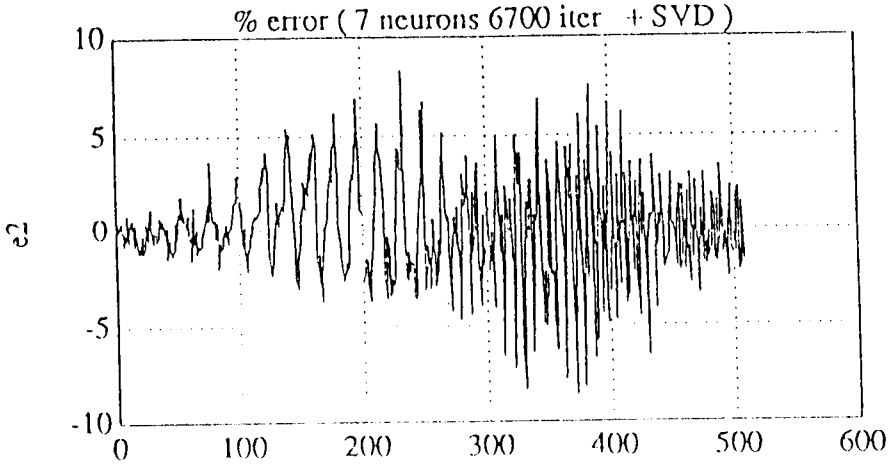

Figure 15.

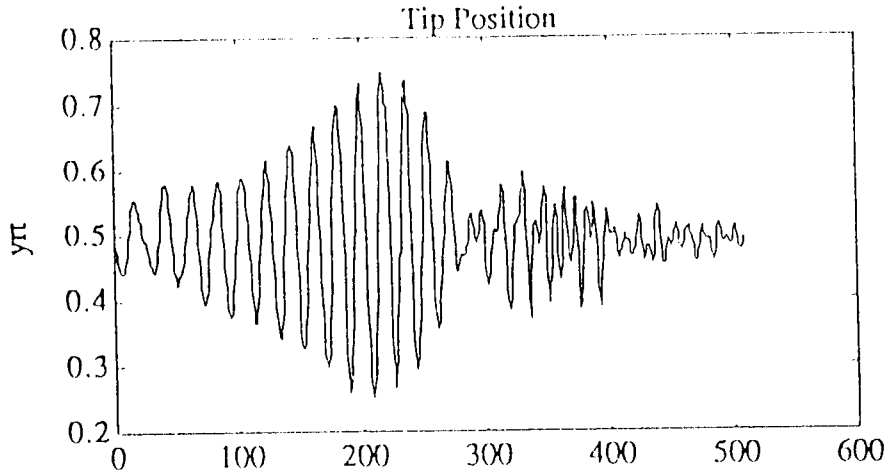

Figure 16.

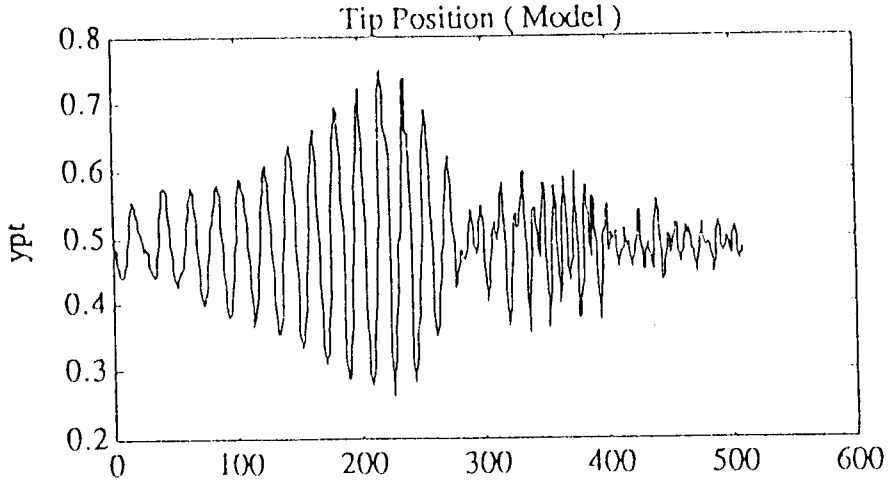

Figure 17.

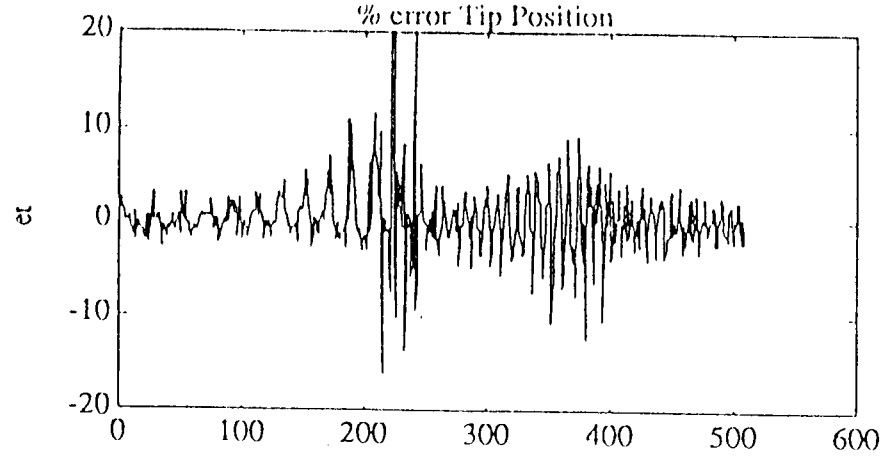

Figure 18.

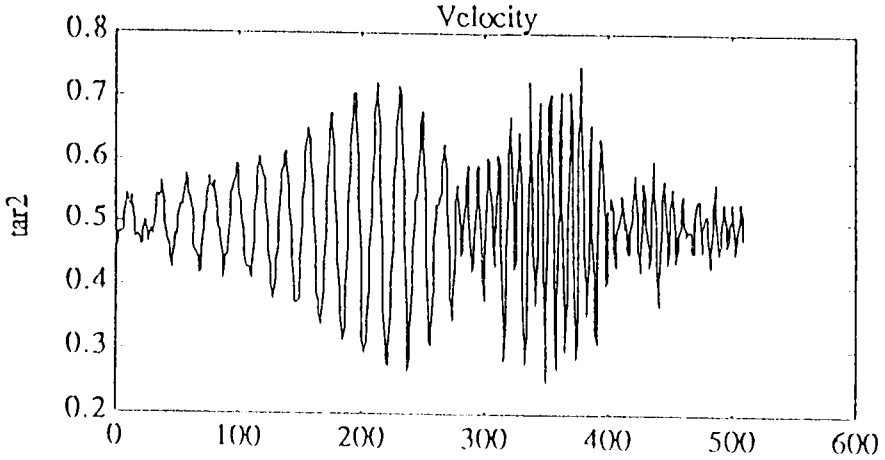

Figure 19.

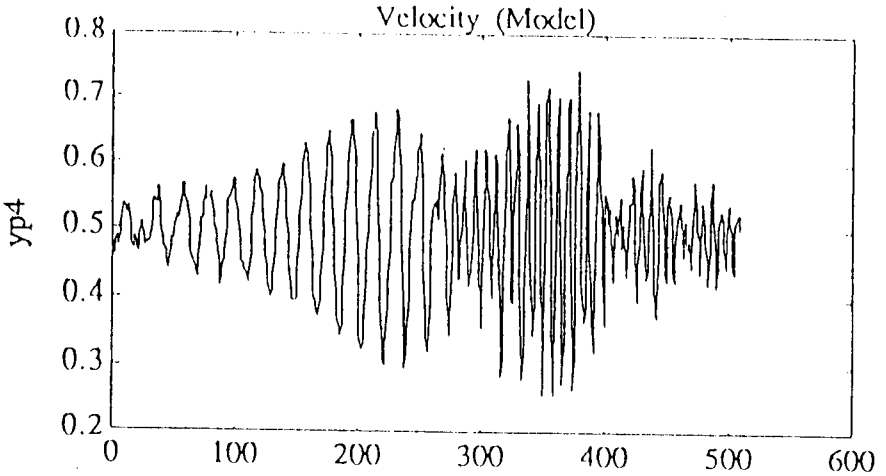

Figure 20.

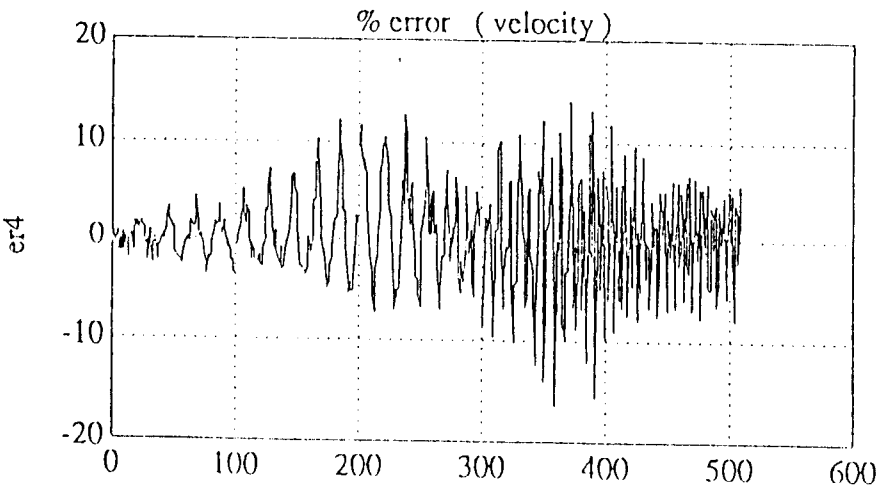

Figure 21.