Journal of Computer Science 6 (12): 1457-1464, 2010
ISSN 1549-3636
© 2010 Science Publications

Accurate Wavelet Neural Network for Efficient Controlling of an Active Magnetic Bearing System

[1]Youssef Harkouss, [1]Souhad Mcheik and [2]Roger Achkar
[1]Department of Electronics, Faculty of Engineering, Lebanese University,
Branch III, P.O. Box 14, 6573, Al Hadath, Beirut, Lebanon
[2]Department of Information Processing, Compiegne University of Technology,
Heudiasyc Laboratory (UMR CNRS 6599), BP 20529, 60205 Compiegne Cedex, France

Abstract: Problem statement: The synthesis of a command by the neural network has an excellent advantage over the classical one such as PID. This study presented a fast and accurate Wavelet Neural Network (WNN) approach for efficient controlling of an Active Magnetic Bearing (AMB) system. **Approach:** The proposed approach combined neural network with the wavelet theory. Wavelet theory may be exploited in deriving a good initialization for the neural network and thus improved convergence of the learning algorithm. **Results:** We tested two control systems based on three types of neural controllers: Multiplayer Perceptron (MLP) controller, RBF Neural Network (RBFNN) controller and WNN controller. The simulation results show that the proposed WNN controller provides better performance comparing with standard PID controller, MLP and RBFNN controllers. **Conclusion:** The proposed WNN approach was shown to be useful in controlling nonlinear dynamic mechanical system, such as the AMB system used in this study.

Key words: Active magnetic bearing system, rotor unbalance, wavelet neural network, radial basis function, LM algorithm, hidden neurons, nonlinear dynamic, oscillations

INTRODUCTION

The design of an AMB (Schweitzer, 1994) is expected to satisfy the static and dynamic requirements in the best possible way. The AMB has several advantages over traditional contact type bearing systems (Shafai *et al*., 1994; Knospe and Collins, 1996). An AMB is used in modern industry where the mechanical systems reach their limits. The AMB systems permit the rotor to turn without any friction, also without any contact with the stator. They are widely used in applications that require high rotation speeds and minimum energy loss. Such systems operate at extreme environment conditions (very low or very high temperature degrees). The AMB systems are nonlinear dynamic systems.

Decentralized PID control systems are the industrial standard for most AMBs. However, these controllers lack systematic design tools and require extensive manual tuning to achieve desired performance for specific applications. A lot of researches about the application of robust control theory in the AMB digital control are reported (Sivrioglu and Nonami, 1999; Xu and Nonami, 2003).

Recently, a big effort has been made to elaborate a new technical command using a feed forward neural network. An adaptive control of a magnetic bearing by neural network has lead to better results. The advantages of such control could be illustrated by the following: A neural network has high immunity levels against the noise, high ability to learn from given examples toward the adaptation with new situations not previously known and high ability to overcome system’s modeling uncertainty. The neural network is also used as an efficient universal tool for function approximation (Haykin, 1998) and as an accurate control mechanism of nonlinear dynamic systems.

In general, the neural networks are directly used in dealing with an identification and control problem. Narendra and Parthasarathy (1990); Jin *et al*. (1995) and Wahab and Mohamed (2008) apply neural network in identification and control of nonlinear systems.

This study presents a comparative study between The WNN, the MLP and the RBFNN in controlling an AMB system. The goal of this study is to show that the control of the AMB by WNN provides an improvement of the response compared to the other control systems used PID, RBFNN or MLP.

Corresponding Author: Youssef Harkouss, Department of Electronics, Faculty of Engineering, Lebanese University, Branch III, P.O. Box 14, 6573, Al Hadath, Beirut, Lebanon

The active magnetic bearing: There are two families of magnetic bearing: passive magnetic bearing and active magnetic bearing. The passive magnetic bearing contains a permanent magnet. In this type of magnetic bearing, we cannot modulate the current vector to control the electromagnetic forces. But in the AMB, we can easily modulate the current vector.

The bloc representing the model of the AMB system in the control system (Fig. 3) is the complete model of an active magnetic broche of small size. This broche is made up of two control plans. Every one includes two axes Y and Z ((Y1, Z1) and (Y2, Z2)). It is also made up of a stubborn that controls the X axis of a motor. This motor maintains the speed of rotation of the rotor (Fig. 1). The model of the broche takes into consideration the disturbances which will be applied to both plans of control. The mathematical equations of the model are developed in (Achkar *et al*., 2006). In fact, this model reflect the real behavior of the system.

In Fig. 2, we present the effect of the rotor unbalance (there is no alignment between the centre of gravity G of the rotor and the geometric centre O). This non-coincidence comes from the intrinsic disturbances. This effect is translated by the existence of δy and δz on every control plan. These disturbances will be taken into considerations in the development equations of the complete model. δy and δz are parameters having a great influence on the existence of the rotor unbalance. δy has an influence on the Y axis and δz has an influence on the Z axis. Centre of gravity of the rotor has equal distance of both control plans, it implicates that parameter δx = 0.

Control system architecture: The bloc diagram of the system controlled by PID without neural network is shown in Fig. 3. In this study we tested two control methods. In the first method the neural network was added to the system controlled by PID by taking the current vector $\vec{I}$ as input and the vector $\vec{Y}$ as output (Fig. 4). In this figure the gain bloc represents the weight of the rotor. Upon adding Neural Network (NN) bloc to the system, the temporal responses of axes position, the response time, the time boarding improved even the overshoot reduced. The difference result vector ($\vec{e}$) (Eq. 1) between the desired positions vector $\overrightarrow{q_d}$ and the system's response vector $\overrightarrow{q_s}$ becomes the input to the PID bloc. The corrected error vector ($\vec{E}$) is multiplied by the mass of the rotor to form the vector $\overrightarrow{V_h}$ (Eq. 2). The bloc is an State-Input Linearization bloc (Achkar *et al*., 2005). The output of this bloc controls the modeling bloc of an AMB system (Charara *et al*., 1996). The vectors $\vec{E}$, $\vec{e}$, $\overrightarrow{V_h}$ and $\vec{I}$ are given by:

$$\vec{e} = \overrightarrow{q_d} - \vec{q}_s \tag{1}$$

$$\vec{E} = k_p \times \vec{e} + k_d \times \frac{\overrightarrow{de}}{dt} + k_i \times \int \vec{e}dt \tag{2}$$

$$\vec{V}_h = m * \vec{E} \tag{3}$$

$$\vec{I} = H * \vec{F} = H * (\overrightarrow{V_h} + \vec{Y}) \tag{4}$$

The way of placing the NN bloc in the control system was to eliminate the error represented by Eq. 2.

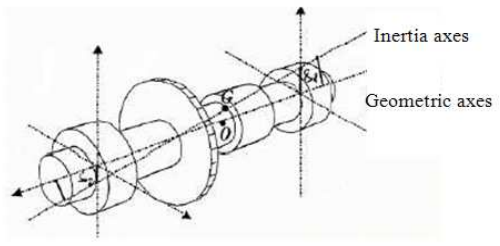

Fig. 1: Representing of AMB with asynchronous motor

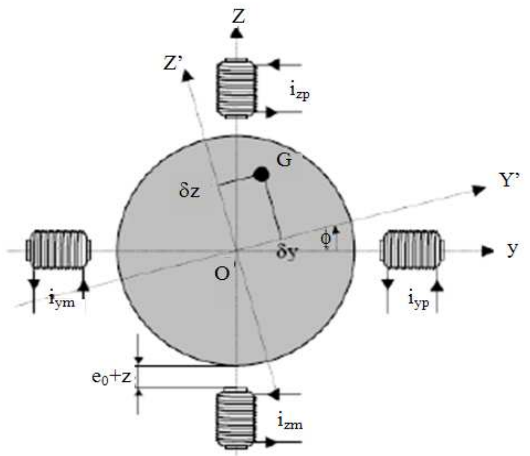

Fig. 2: Disturbed system plan control

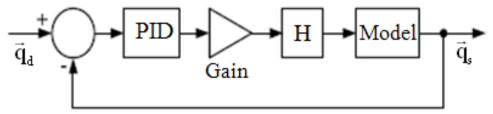

Fig. 3: Bloc diagram of the closed loop system without neural network

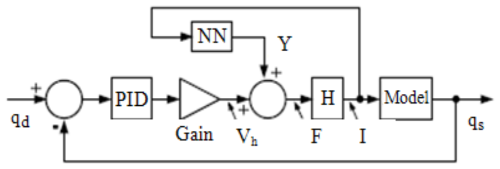

Fig. 4: Bloc diagram of the closed loop system with neural network (first method)

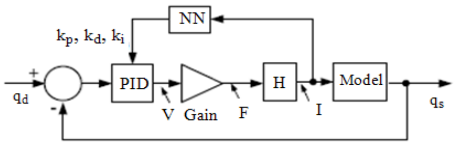

Fig. 5: Bloc diagram of the closed loop system with neural network (second method)

The $\vec{F}_t$ in Eq. 5 demonstrates all the uncertainties of the system modeling (Charara *et al*., 1992; Charara and Caron, 1992):

$$\vec{I} = M(\vec{q}) \times \ddot{\vec{q}} + \vec{h}(\dot{\vec{q}}, \vec{q}) + \overrightarrow{F_t} \quad (5)$$

As mentioned above, the neural network can overcome all the uncertainties of the modeling of non linear dynamic system. We can easily demonstrate that:

$$m * \left(k_p \times \vec{e} + k_d \times \frac{\overrightarrow{de}}{dt} + k_i \int \vec{e}dt \right) = \vec{F} - \vec{Y} \quad (6)$$

We can conclude that, in order to minimize the error represented by Eq. 2, it is sufficient to minimize the vector $\overrightarrow{V_h} = \vec{F} - \vec{Y}$. Then the goal of the NN bloc added to the control system is to minimize this vector $\overrightarrow{V_h}$.

The bloc diagram of the second control method is shown in Fig. 5. In this method the NN bloc is used to compute the different parameters k_p, k_d and k_i of the PID controller. The major problem of the first method is the existence of oscillations in the axes's responses. The origin of these oscillations comes from the using of non optimized parameters values k_p, k_d and k_i of the PID controller. To eliminate the oscillations and add more stability to the system, it has been necessary to propose the use of second NN control method. The aim is to find the optimized PID parameters values by a systematic way. This effect was definitely explained in (Achkar, 2008; Mcheik *et al*., 2009). In our study, we placed the MLP by WNN first and then by RBFNN, in the two control methods. The training was online. The training of the MLP (Achkar *et al*., 2006) was also online and in two stages direct and backpropagation.

Wavelet neural network and learning scheme:

Wavelet neural network: Interest in wavelet analysis has been grown very rapidly in recent years. The wavelet theory is a rapidly developed branch of mathematics, which has offered very efficient algorithms for approximating, estimating, analyzing and compressing nonlinear functions. WNNs are widely applied to a variety of practical problems. Due to the similarity between discrete inverse wavelet transform and one-hidden-layer feedforward neural network (i.e., three-layer network), the idea of combining both wavelets and neural networks has been proposed by Zhang and Benveniste (1992) and Zhang (1997). A WNN is an adaptive discretization of the continuous inverse wavelet transform. It can also be considered as a one-hidden-layer feedforward neural network with wavelets as activation functions of its hidden neurons.

A bloc diagram of a multiple-input-multiple-output WNN is shown in Fig. 6. The expression of the kth ($1 \le k \le q$) output of the WNN can be written as:

$$y_k = (X_j) = \sum_{i=1}^{h} w_{ki} \psi_i (X_j)$$

where, q is the number of linear output neurons.

In practice, this equation is modified as follows:

$$y_k = (X_j) = \sum_{i=1}^{h} w_{ki} \psi_i (X_j) + U_k^T X_j + \theta_k$$

Or:

$$y_k = (X_j) = \sum_{i=1}^{h} w_{ki} \psi_i (X_j) + \sum_{l=1}^{p} u_{kl} x_{jl} + \theta_k$$

Where:

w_{ki}	= The linear weight between ith hidden neuron and kth output neuron
$u_k = (u_{k1}, u_{k2}, \ldots u_{kp})^T$	= The vector that represents the direct linear connections between input layer and kth output neuron
$X_j = (x_{j1}, x_{j2}, \ldots x_{jp})^T$	= The jth input vector of WNN and h is the number of hidden neurons in the network
θ_k	= The bias parameter of the kth linear output neuron

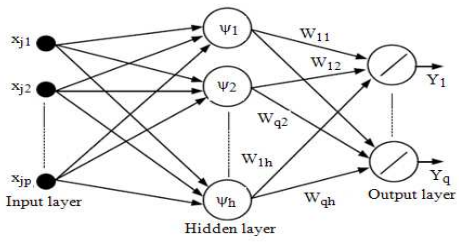

Fig. 6: Structure of wavelet neural network

$\psi_i(.)$ = The wavelet activation function for the ith hidden neuron and it is calculated using the following equation (Oussar *et al*., 1998)

$$\psi_i(X_j) = \prod_{l=1}^{h} \phi\left(\frac{x_{jl} - b_{il}}{a_{il}}\right) \tag{9}$$

where, i = 1,…,h
a_{il} = the dilation parameter for the ith hidden neuron
b_{il} = the translation coefficient for the ith hidden neuron and the lth input neuron
ϕ = the mother wavelet

In this study, the mother wavelet function ϕ has been chosen as (Zhang, 1997):

$$\phi(x) = -xe^{-0.5x^2} \tag{10}$$

Y = (y_1, y_2,…,y_q) is the output vector of the WNN. All the network parameters (θ_k, u_{kl}, w_{ki}, a_{il}, b_{il}) must be adapted on the training data.

Learning algorithm: In this study the Levenberg-Marquardt (LM) algorithm is used to train the WNN. The LM algorithm is based on the standard Newton's method and has been shown to be a very powerful optimization technique (Ampazis and Perantonis, 2002; Hagan and Menhaj, 1994). In this application, LM algorithm is clearly superior to the classical BP (Backpropagation) and CG (Conjugate gradient) approaches. Training the WNN is to determine the α = {θ_k, u_{kl}, w_{ki}, a_{il}, b_{il}} (l = 1,…p, i = 1,…h, k = 1,…,q) such that the error function $E_{net}(\alpha)$ represented by Eq. 11 is minimized (first control method):

$$E_{net}(\alpha) = \sum_{j=1}^{N} e_j^2 \tag{11}$$

and:

$$e_j = \frac{1}{2}\sum_{k=1}^{q} v_{hk}^2(j) \tag{12}$$

N = The number of training inputs
v_{hk} (j) = The kth element of the vector V_h = (v_{h1}, v_{h2},…,v_{hq}) corresponding to the jth training input

Here the variables to be optimized are the parameters in vector α. The error function $E_{net}(\alpha)$ is minimized by refining these parameters using the LM algorithm. The nth correction of these parameters is described as:

$$\alpha(n+1) = \alpha(n) - (J^TJ + \lambda 1)^{-1}J^Te_{net} \tag{13}$$

where e_{net} = (e_1, e_2,…,e_N)^T . 1 is the identity matrix and λ is a constant parameter. The Jacobian matrix J is given by:

$$J = \begin{bmatrix} \frac{\partial e_1}{\partial \theta_k} & \frac{\partial e_1}{\partial u_{kl}} & \frac{\partial e_1}{\partial w_{ki}} & \frac{\partial e_1}{\partial \alpha_{il}} & \frac{\partial e_1}{\partial |b_{il}} \\ \frac{\partial e_2}{\partial \theta_k} & \frac{\partial e_2}{\partial u_{kl}} & \frac{\partial e_2}{\partial w_{ki}} & \frac{\partial e_2}{\partial \alpha_{il}} & \frac{\partial e_2}{\partial b_{il}} \\ \frac{\partial e_N}{\partial \theta_k} & \frac{\partial e_N}{\partial u_{kl}} & \frac{\partial e_N}{\partial w_{ki}} & \frac{\partial e_N}{\partial \alpha_{il}} & \frac{\partial e_N}{\partial b_{il}} \end{bmatrix}$$

The formulas for each element in the Jacobian matrix (calculation for the first control method) are:

$$\frac{\partial e_j}{\partial \theta_k} = -v_{hk}(j),$$

$$\frac{\partial e_j}{\partial u_{kl}} = -v_{hk}(j)x_{jk}$$

$$\frac{\partial e_j}{\partial w_{ki}} = -v_{hk}(j)\psi_i(X_j)$$

$$\frac{\partial e_j}{\partial b_{il}} = \left(\frac{1}{\alpha_{il}}\right)\psi_i(X_j)\left(\frac{1}{z_{il}} - Z_{il}\right)(\textstyle\sum_{k=1}^{q} v_{hk}(j)w_{ki}$$

$$\frac{\partial e_j}{\partial \alpha_{il}} = \frac{\partial_{e_j}}{\partial b_{il}} Z_{il}$$

$$Z_{il} = \frac{x_{jl} - b_{il}}{\alpha_{il}}, \text{and } i = 1,...h, k = 1,...q, l = 1,..., p, j = 1,...N$$

Note that in the second control method the training process tends to minimize the error function represented by Eq. 11 but e_j is given by:

$$e_j = \frac{1}{2}\sum_{k=1}^{q} v_k^2(j) \tag{14}$$

where, $v_k(j)$ is the kth element of the vector V = (v_1, v_2,…,v_q) corresponding to the jth training input.

A good initialization of the WNN parameters is very important. The good initialization yields a fast training procedure. In this study, we adopt the initialization proposed by Oussar *et al*. (1998). Zhang and Benveniste (1992) also used a similar but more complicated method for parameter initialization.

RBF neural network and learning scheme: Figure 6 shows the structure of a typical RBFNN (Haykin, 1998), but the activation function for the hidden layer is replaced by a radial and symmetric Gaussian function (Eq. 16). The kth ($1 \le k \le p$) output of the RBFNN is:

$$y_k(X_j) = \sum_{i=1}^{h} w_{ki}\phi_i(X_j) \tag{15}$$

$$\phi_i(X_j) = e^{\frac{\left\|x_j - T_i\right\|^2}{\sigma_i^2}} \tag{16}$$

Where:

X_j	= The jth input vector of the RBFNN
$T_i = (t_{i1}, t_{i2},\ldots, t_{ip})^T$	= The center vector of the ith hidden neuron
σ_i	= The width parameter of the ith hidden neuron which is related to the spread of this function around its center

All the network parameters (w_{ki}, t_{il}, σ_i) must be adapted on the training data. The training procedure of the RBFNN consists in determining the centers of the hidden neurons by an unsupervised technique and the weights of connections of the hidden-output layer and the width parameters by a supervised technique. The fact that the performance of an RBFNN critically depends upon the chosen centers, we proposed to implement a non supervised standard algorithm the rival penalized competitive learning algorithm (Xu *et al*., 1993) to best determine the centers of the hidden activation functions. Training the RBFNN is to determine the $\alpha = \{w_{ki}, t_{il}, \sigma_i\}$ ($l = 1,\ldots p$, $i = 1,\ldots h$, $k = 1,\ldots,q$) such that the error function $E(\alpha)$ represented by Eq. 11 is minimized. The training of the RBFNN was done by epoch, where every epoch contains N training inputs. After every epoch, the rival penalized algorithm is executed to choose the pertinent set of centers. Then the width parameters are calculated according to simple formula presented in (Hassoun, 1995). It is only a rough guide that provides a starting point for the width calculation by the training algorithm. At every training input, a gradient descent algorithm (Haykin, 1998) is used iteratively to train the weights of different connections and the width parameters in the opposite direction of the respective partial derivative of the error.

In the second control method the training process tends also to minimize the error function represented by Eq. 11 where e_j is given by Eq. 14.

MATERIALS AND METHODS

Figure 4 shows the bloc diagram of the first control system. In this case, the rotor unbalance is modeled with three components δx, δy and δz. These components represent the disturbance parameters of the centre of inertia compared to the centre of gravity. δy will influence the positions Y1 and Y_2 and δz will influence the positions Z_1 and Z_2. The fact that the centre of gravity of the rotor has equal distance of both control plans, it implicates that parameter $\delta x = 0$. The control of the AMB by NN requires two steps: Learning process and online treatment. In this method, the number of inputs of each network is $p = 10$ and the number of output neurons is $q = 5$. In the Table 1, we list for each network: The number of hidden neurons, the error calculated at the end of the training process and the number of epochs needed in the training process. The learning parameter (η) used in the training process of the RBFNN is 0.01 ($\eta = 0.01$ for the MLP) and the constant parameter λ of the LM algorithm is 0.001. The number of examples per epoch is $N = 501$.

Figure 5 shows the bloc diagram of the second control method. The learning parameter (η) used in the training process of the RBFNN and MLP is 0.01 and the constant parameter λ of the LM algorithm is 0.001. The number of examples per epoch is also $N = 501$. In the Table 2, we list also for each network: The number of hidden neurons, the error calculated at the end of the training process and the number of epochs needed in the training procedure. The number of inputs of each network is $p = 10$ and the output vector of the NN bloc is given by ($q = 15$):

$$Y = (K_{px}, K_{pY1}, K_{pZ1}, K_{pY2}K_{pZ2}, \\ K_{dx}, K_{dY1}, K_{dZ1}, K_{dY2}K_{dZ2}, \\ K_{ix}K_{iY1}, K_{iZ1}, K_{iY2}K_{iZ2})^T$$

Table 1: Optimized parameters of each network

Network	No. of hidden neurons	No. of epochs	Error
MLP	8	28	6×10^{-6}
RBFNN	8	28	6×10^{-6}
WNN	8	28	4.2×10^{-6}

Table 2: Optimized parameters of each network

Network	No. of hidden neurons	No. of epochs	Error
MLP	6	50	8×10^{-6}
RBFNN	6	50	9×10^{-6}
WNN	6	50	4×10^{-6}

RESULTS

The execution of different simulations was done using the simulator MATLAB. Figure 7-12 show the simulation results of the first control method. Figure 7 shows the error function as a function of the number of epochs for all the networks.

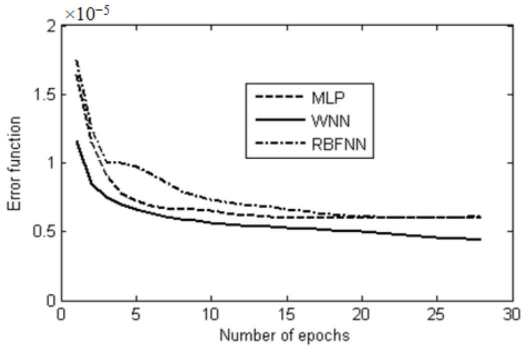

Fig. 7: Error function of different networks

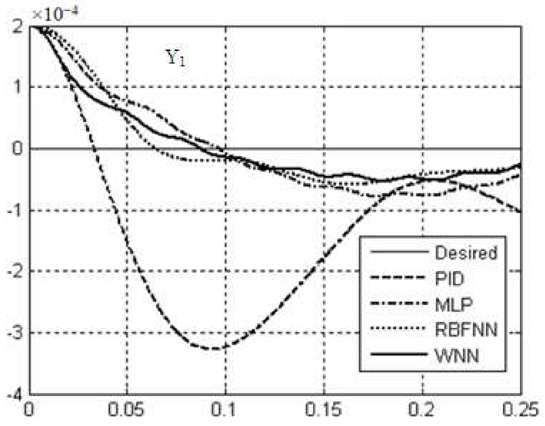

Fig. 8: Response on the x axis

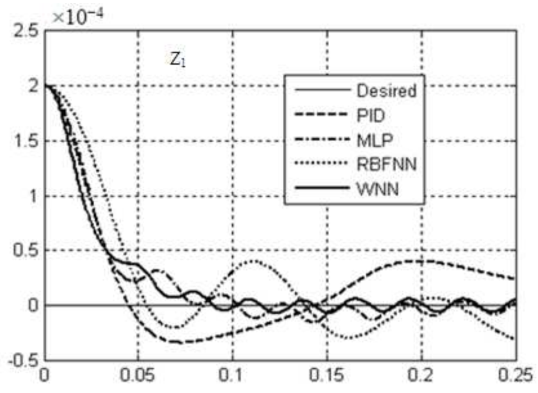

Fig. 9: Response on the Z_1 axis

After optimization of the parameters of MLP, RBFNN and WNN, a simulation giving the temporal answers according to time was made for the three axes (Fig. 8-12). Figure 13-18 shows the different simulation results of the second control method.

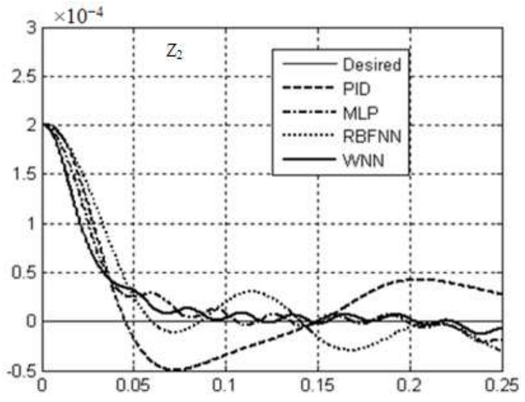

Fig. 10: Response on the Z_2 axis

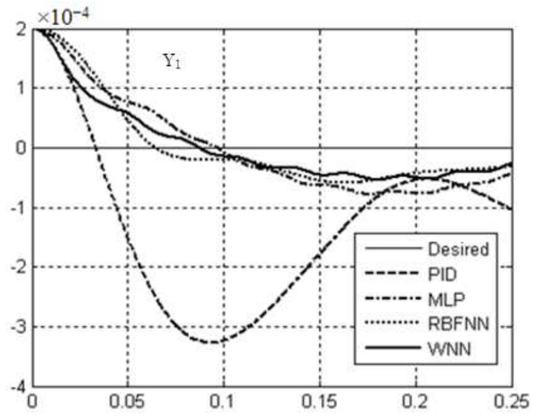

Fig. 11: Response on the Y_1 axis

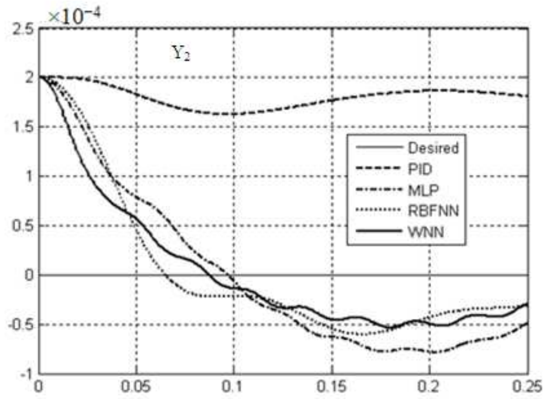

Fig. 12: Response on the Y_2 axis

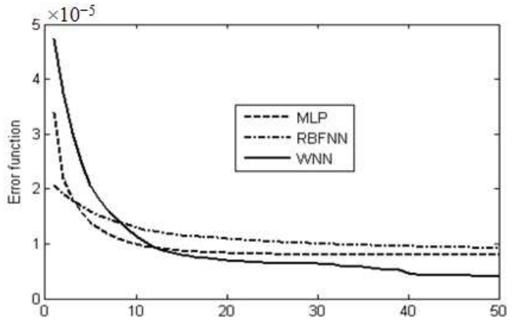

Fig. 13: Error function of different networks

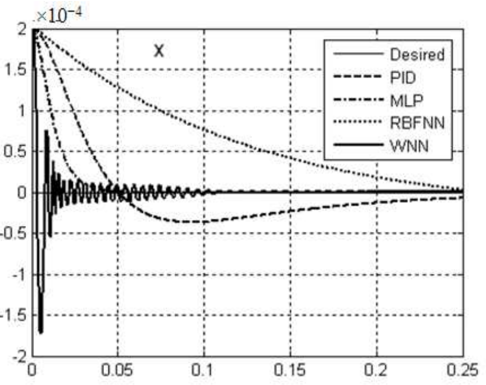

Fig. 14: Response on the x axis

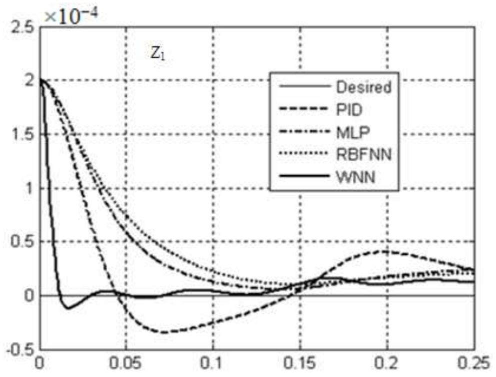

Fig. 15: Response on the Z_1 axis

DISCUSSION

From the values in Table 1 and 2, it is obvious that the WNN consistently produces lower error value than the MLP and RBFNN, suggesting that the WNNs have better prediction accuracy and adaptability. In this study, we also find that the RBFNN doesn't have better average performance than the MLP. Besides better accuracy and adaptability, the WNNs are easier to apply than the RBFNNs.

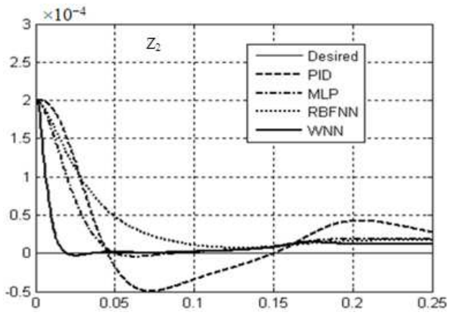

Fig. 16: Response on the Z_2 axis

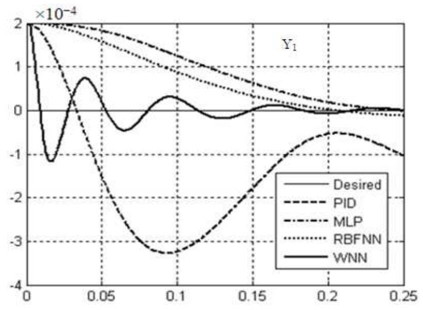

Fig. 17: Response on the Y_1 axis

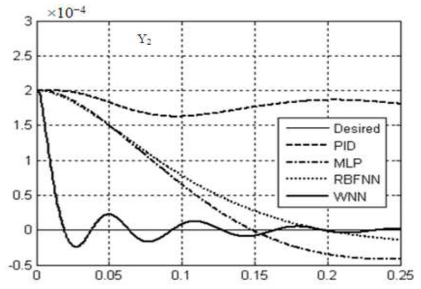

Fig. 18: Response on the Y_2 axis

To apply the RBFNN, in addition to specifying the number of radial basis functions (hidden neurons), some additional algorithms are needed to determine the center and width coefficient of each radial basis function. Simulation results have shown that the WNN controller has a good initialization of its parameters during the learning process and then a faster learning speed with a smaller error. From Fig. 7-18 it can be seen that the WNN control system gives better results in comparison with PID, MLP or RBFNN control systems. The WNN control scheme is then more suitable to control the AMB system under the possible occurrence of uncertainties.

CONCLUSION

In this study, we presented the effectiveness of a neural command applicable on an AMB system. The fact that this system cannot be stable in the three space directions, the control of the rotor position was necessary. We tested two control systems based on three types of neural controllers. The simulation results showed that the WNN controller provides a better performance comparing with standard PID controller, MLP and RBFNN controllers. As a perspective work, the synthesis of a command by WNN according to the two methods previously mentioned may be done in real time on an experimental platform. The simulated results and the experiment ones may be compared. The application of the WNN command in real time permit us to eliminate the effect of the rotor unbalance and the reduction of the energy consummation with conservation of the high performance speed rotation.

REFERENCES

Achkar, R., 2008. Contribution à l'étude et à la validation expérimentale de commandes neuronales d'un palier magnétique actif. Ph.D. Thesis, Compiegne University.

Achkar, R., C. Nasr and A. Charara, 2005. Control of an active magnetic bearing using MLP. Proceeding of the Conference on the Artificial Neural Networks, (ANN'05), IEEE Xplore Press, Missouri, USA., pp: 811-820.

Achkar, R., C. Nasr, J. De Miras and A. Charara, 2006. A new method of controlling AMB through neural network. IEEE CCA/CACSD/ISIC.

Ampazis, N. and S.J. Perantonis, 2002. Two highly efficient second-order algorithms for training feedforward networks. IEEE Trans. Neural Networks, 13: 1064-1074. PMID: 18244504

Charara, A. and B. Caron, 1992. Magnetic bearing: Comparison between linear and nonlinear functioning. Proceeding of the 3rd International Symposium on Magnetic Bearings, (MB'92), IEEE Xplore Press, USA., pp: 451-463.

Charara, A., B. Caron and G. Lemarquand, 1992. Modeling and non-interactive control of an active magnetic bearing. Int. J. Applied Electromagnet. 2: 359-368.

Charara, A., J. De Miras and B. Caron, 1996. Nonlinear control of a magnetic levitation system without premagnetization. IEEE Trans. Control Syst. Technol., 4: 513-523. DOI: 10.1109/87.531918

Hagan, M.T. and M.B. Menhaj, 1994. Training feedforward networks with the Marquardt algorithm. IEEE Trans. Neural Networks, 5: 989-993. DOI: 10.1109/72.329697

Hassoun, M.H., 1995. Fundamentals of Artificial Neural Networks. 1st Edn., MIT Press, Cambridge, MA., ISBN: 10: 026208239X, pp: 511.

Haykin, S., 1998. Neural Networks: A Comprehensive Foundation. 2nd Edn., Prentice Hall, New York, ISBN: 0132733501, pp: 842.

Jin, L., P.N. Nikiforuk and M.M. Gupta, 1995. Fast neural learning and control of discrete-time nonlinear systems. IEEE Trans. Syst. Man. Cybernet., 25: 478-488. DOI: 10.1109/21.364860

Knospe, C.R. and E.G. Collins, 1996. Introduction to the special issue on magnetic bearing control. IEEE Trans. Control Syst. Technol., 4: 481-598. DOI: 10.1109/TCST.1996.531914

Mcheik, S., R. Achkar and C. Nasr, 2009. Control of an active magnetic bearing by RBF neural network. M.S. Thesis, Lebanese University.

Narendra, K.S. and K. Parthasarathy, 1990. Identification and control of dynamical systems using neural networks. IEEE Trans. Neural Networks, 1: 4-27. PMID: 18282820

Oussar, Y., I. Rivals, L. Personnaz and G. Dreyfus, 1998. Training wavelet networks for nonlinear dynamic input-output modeling. Neurocomputing, 20: 173-188. DOI: 10.1016/S0925-2312(98)00010-1

Schweitzer, G., 1994. Active Magnetic Bearings: Basics, Properties and Applications of Magnetic Bearings. Vdf Hochschulverlag AG, ETH Zurich, ISBN: 10: 3728121320, pp: 252.

Shafai, B., S. Beale, P. LaRocca and E. Cusson, 1994. Magnetic bearing control systems and adaptive forced balancing. IEEE Control Syst., 14: 4-13. DOI: 10.1109/37.272775

Sivrioglu, S. and K. Nonami, 1999. Gain scheduled sliding mode control of active magnetic bearing system with gyroscopic rotor. Trans. Jap. Soc. Mech. Eng., 65: 2665-2671.

Wahab, N.I.A. and A. Mohamed, 2008. Transient stability assessment of a power system using probabilistic neural network. Am. J. Applied Sci., 5: 1225-1232. http://www.scipub.org/fulltext/ajas/ajas591225-1232.pdf

Xu, L., A. Krzyzak and E. Oja, 1993. Rival penalized competitive learning for clustering analysis, RBF net and curve detection. IEEE Trans. Neural Networks, 4: 636-649. PMID: 18267764

Xu, Y. and K. Nonami, 2003. A fuzzy modeling of active magnetic bearing system and sliding mode control with robust hyperplane using μ-synthesis theory. JSME Int. J. Ser. C, Mech. Syst. Mach. Elements Manuf., 46: 409-415.

Zhang, Q. and A. Benveniste, 1992. Wavelets networks. IEEE Trans. Neural Network, 3: 889-898. DOI: 10.1109/72.165591

Zhang, Q., 1997. Using wavelet network in non parametric estimation. IEEE Trans. Neural Networks, 8: 227-236. DOI: 10.1109/72.557660