ARCHIVES OF ACOUSTICS
35, 2, 191–202 (2010)

Active Noise Control Algorithm Based on a Neural Network and Nonlinear Input-Output System Identification Model

Tomasz KRUKOWICZ

Central Institute for Labour Protection –
National Research Institute
Czerniakowska 16, 00-701 Warszawa, Poland
e-mail: tokru@ciop.pl

(*received April 16, 2010; accepted Maj 18, 2010*)

The development of digital signal processors and the increase in their computing capabilities bring opportunities to employ algorithms with multiple variable parameters in active noise control systems. Of particular interest are the algorithms based on artificial neural networks. This paper presents an active noise control algorithm based on a neural network and a nonlinear input-output system identification model. The purpose of the algorithm is an active noise control system with a nonlinear primary path. The algorithm uses the NARMAX system identification model. The neural network employed in the proposed algorithm is a multilayer perceptron. The error backpropagation rule with adaptive learning rate is employed to update the weight of the neural network. The performance of the proposed algorithm has been tested by numerical simulations. Results for narrow-band input signals and nonlinear primary path are presented below.

Keywords: active noise control, neural networks, system identification, nonlinear phenomena.

1. Introduction

Active noise control (ANC) has been a dynamically developing branch of science for over 60 years. The popularity of ANC systems is mainly due to two factors. Firstly, ANC systems have some advantages over passive noise reduction techniques in the low-frequency range (below 500 Hz) in which efficient passive noise reduction solutions (sound absorbers) usually take up a lot of space. The second reason for ANC's growing popularity is the development of digital control technologies. Increasing computing capabilities of digital signal processors (DSP) allow for the implementation of multivariable control algorithms in ANC systems. Artificial neural networks (ANN) are a group of such techniques that have been widely explored for pattern recognition purposes. In the system theory, pattern recognition is equivalent to the identification of static systems. The objective of pattern recognition is to find a mapping Ψ between compact sets $U_i \in \Re^N$ and elements y_i of compact set $Y \in \Re^M$ (U_i is a pattern of class y_i). Mapping Ψ for the identification of a single-input–single-output (SISO) dynamical system is between pairs of two real-valued functions of time: input $u(t)$ and output $y(t)$ of a system, $t \in [0, \infty)$. Successful identification of a system occurs when the difference between model response and system response does not exceed an arbitrarily assumed accuracy (1) (Narendra, Parthasarathy, 1990; Nelles, 2000; Haykin, 1993):

$$|y_\Psi(t) - y_P(t)| \leq \varepsilon, \qquad t > t_0. \tag{1}$$

Since the purpose of an ANC system is to reduce acoustic pressure $p(t)$ in a specified point/area, arbitrarily assumed $\max\limits_t \left\{p^2(t)\right\}$ at $t > t_0$ is identical to accuracy ε. Therefore the ANC application can be treated as a dynamical system identification problem. Further investigation shows that the system identification problem is defined as follows (Chen, Billings, 1989; Narendra, Parthasarathy, 1990): for any $u := u(t) \in U, y := y(t) \in Y$, $t \in [a, b]$, where U, Y are real Banach spaces.

$$\|y_\Psi - y_P\| = \|\Psi(u) - P(u)\| \leq \varepsilon. \tag{2}$$

The norm $\|\bullet\|$ is defined basing on the system output space and depends on the chosen identification manner. Since modern ANC systems are mainly digital, further investigation concerns sampled-data systems.

A major advantage of ANNs is the approximation capability of nonlinear multivariable functions (Cybenko, 1989) (e.g. ANNs are capable to solve the pattern recognition problems). This fact was used to derive a group of ANN-based methods for the control and identification of a wide class of nonlinear dynamical systems (Elliott, 2001; Narendra, Parthasarathy, 1990; Nelles, 2000; Haykin, 1994). Nonlinear phenomena are a feature of primary and secondary paths of feedforward ANC systems. The presence of nonlinear phenomena in any element of ANC systems, based on commonly used adaptive transversal filters (such as FXLMS), may significantly decrease efficiency and may even lead to a loss of stability in the system. Investigations into the use of ANNs for ANC purposes performed by several researchers brought useful methods capable of maintaining some of the above-mentioned nonlinear phenomena (Snyder, Tanaka, 1995; Strauch, Mulgrew, 1999; Bouchard *et al.*, 1999; Morzyski, Makarewicz, 2003; Duch *et al.*, 2000). The aim of this article is to derive an efficient method for ANC systems with a nonlinear primary path. A known ANN-based ANC method is the Filtered-X Backpropagation Neural Network (FXBPNN) (Snyder, Tanaka, 1995; Zhou *et al.*, 2005). The method is an extension of the FXLMS algorithm (Hansen, Snyder, 1997; Engel, Nizioł, 1995) for systems with nonlinearities and is attained by the replacement of the finite impulse response filter with a forward sigmoidal neural network. Compared to the FXLMS algorithm, FXBPNN is distinguished by better performance when the primary path establishes a weak nonlinearity (Elliott, 2001).

The ANC method proposed in this paper is a modification of FXBPNN. In order to explain the difference between the proposed one and FXBPNN algorithms, a temporary assumption is made that the influence of the secondary path on the ANC system performance can be ignored. Since FXBPNN utilizes present values of reference signals, it can be classified as a Nonlinear Finite Impulse Response (NFIR)filter (Nelles, 2000). The proposed ANC algorithm is based on the Nonlinear Autoregressive Moving Average with Exogenous Input (NARMAX) system identification model (Chen, Billings, 1989; Nelles, 2000). The NARMAX identification model in general utilizes present and past reference signals and the past output signal:

$$y(n+1) = \Psi\left(y(n{-}1), y(n{-}2), \cdots, y(n{-}I), u(n), u(n{-}1), \cdots, u(n{-}O{+}1)\right), \tag{3}$$

I and O denote a dimension of input and output vectors respectively, while n is a discrete moment in time. Depending on the source of the output signal, the NARMAX model is described as series-parallel when the source is the system being identified, and parallel when the source is the identification model (Chen, Billings, 1989; Narendra, Parthasarathy, 1990; Nelles, 2000). The dynamical system must satisfy several conditions: it should be casual, continuous, bounded and time-invariant (Narendra, Parthasarathy, 1990). The utilization of the forward neural network for a discrete series-parallel NARMAX model (Elliott, 2001; Narendra, Parthasarathy, 1990; Nelles, 2000; Haykin, 1994) has been shown in Fig. 1.

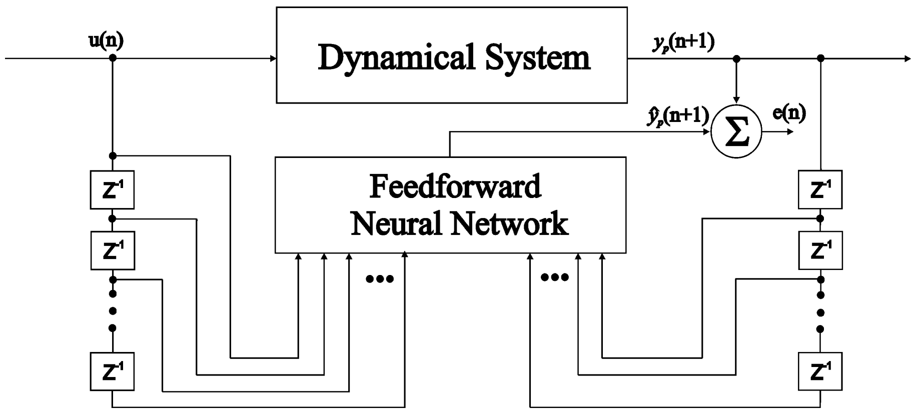

Fig. 1. System identification based on the NARMAX model with a forward neural network.

The feedforward neural network is driven by an actual value of input signal $u(n)$ and two separate tapped delay lines. The first one contains a set of past values of the input signals, while the second one contains a set of past values of the output signals of the system. The error signal $e(n)$ is created by addition of the system output and network output signals. The error signal is used to update ANN weights, according to the specific learning rule. The proposed ANC algorithm is named QNBPN as it involves the use of the NARMAX model, artificial neural networks (ANN) and a backpropagation (BP) learning algorithm. 'Q' stands for 'quasi', as the algorithm is neither parallel nor series-parallel.

2. Structure of the neural network

The neural network in the QNBPN algorithm is a three-layer, pure feedforward, sigmoidal perceptron (Fig. 2). Since there is no feedback connection inside the network, it can be assumed to be stable. Furthermore, a static error backpropagation learning algorithm can be applied to update network weights. Feedback connection in a neural network implies the application of a dynamic error backpropagation algorithm that characterizes a higher computational load.

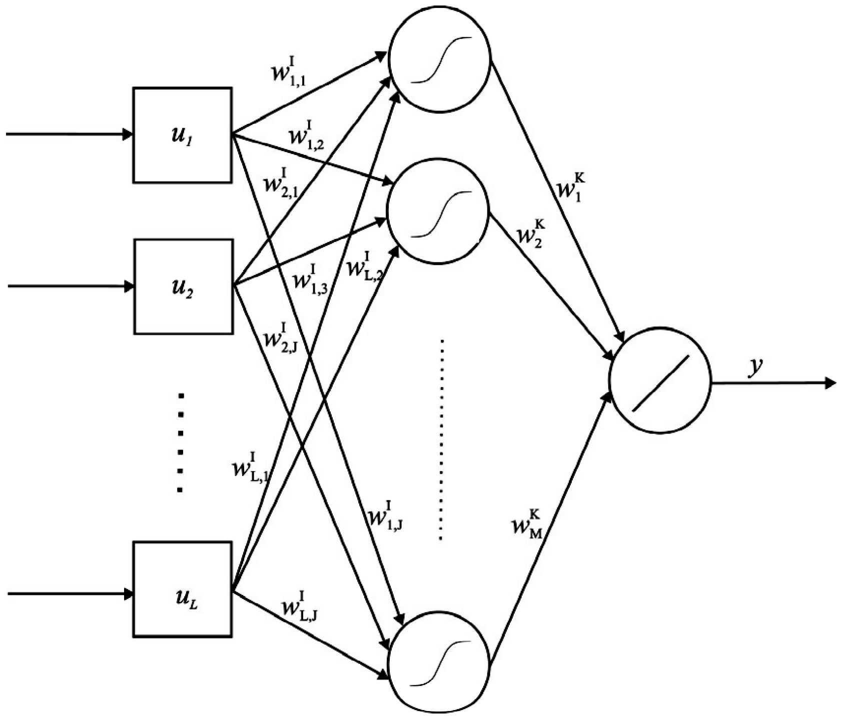

Fig. 2. Structure of the neural network.

The first layer gathers input signals. The second (nonlinear) and third (linear) layers perform nonlinear approximations (Haykin, 1994; Engel, Nizioł, 1995). Before explaining the ANC algorithm, forward signal propagation through the network will be presented. First, using an input vector of L elements $\mathbf{u} = [u_1, u_2, ..., u_L]$, weights matrix $\mathbf{W}$ of the hidden layer with J nonlinear neurons, the output vector of hidden layer is calculated:

$$\mathbf{y}_H = [y_1, y_2, ..., y_J] = f(\mathbf{v}_H) = f\left[(\mathbf{u}\mathbf{W}^{\mathrm{T}})\right], \tag{4}$$

f is the bipolar sigmoidal output function of nonlinear neurons, $\mathbf{v}$ denotes dot product of $\mathbf{u}\mathbf{W}^{\mathrm{T}}$

$$f(x) = \frac{1-e^{-x}}{1+e^{-x}} = \tanh(x), \tag{5}$$

the weights matrix of the hidden layer has the form shown below:

$$\mathbf{W} = \begin{bmatrix} \mathbf{w}_1^{\mathrm{T}} & \cdots & \mathbf{w}_J^{\mathrm{T}} \end{bmatrix} = \begin{bmatrix} w_{11} & \cdots & w_{J1} \\ \vdots & \ddots & \vdots \\ w_{1L} & \cdots & w_{JL} \end{bmatrix}. \tag{6}$$

Next, using an output vector of the hidden layer and a weight vector of the output layer $\mathbf{w} = [w_1, w_2, \ldots w_J]$, the output of the network is calculated:

$$y = av = a\left(\mathbf{y}_H \mathbf{w}^{\mathrm{T}}\right), \tag{7}$$

a is a specified constant, v is a dot product of $\mathbf{y}_H\mathbf{w}^{\mathrm{T}}$.

3. ANC algorithm

The utilization of the NARMAX model for ANC purposes is not directly available. There are two major difficulties. Firstly, there is a mutual compensation of both output signals. Secondly, a secondary path is present. This section shows a detailed solution for the above-mentioned problems, as well as a description of the proposed ANC algorithm. The proposed structure of ANC solution is presented in Fig. 3.

The reference signal $x(n)$ drives a nonlinear primary path P, estimated secondary path with impulse response parameters $\widehat{\mathbf{S}} = [s_1, s_2, ..., s_N]$, and the tapped delay line which is connected to the input of the neural network. For simplification of the derivation, it has been assumed that the secondary path is linear and can be estimated using a finite impulse response filter and an LMS algorithm (Engel, Niziołł, 1995). The nonlinear path transforms the reference signal into the desired signal $d(n)$ (desired corresponds to plant the output signal y_p in Eq. (2)):

$$d(n) = P\left[x(n), x(n-1), ..., x(n-I+1), y_p(n-1),\right.$$
$$\left. y_p(n-1), ..., y_p(n-O)\right], \tag{8}$$

I, O depend on the properties of the primary path. Since the desired signal is reduced by the filtered output of the neural network $y'(n)$, the error signal is:

$$e(n) = d(n) - y'(n). \tag{9}$$

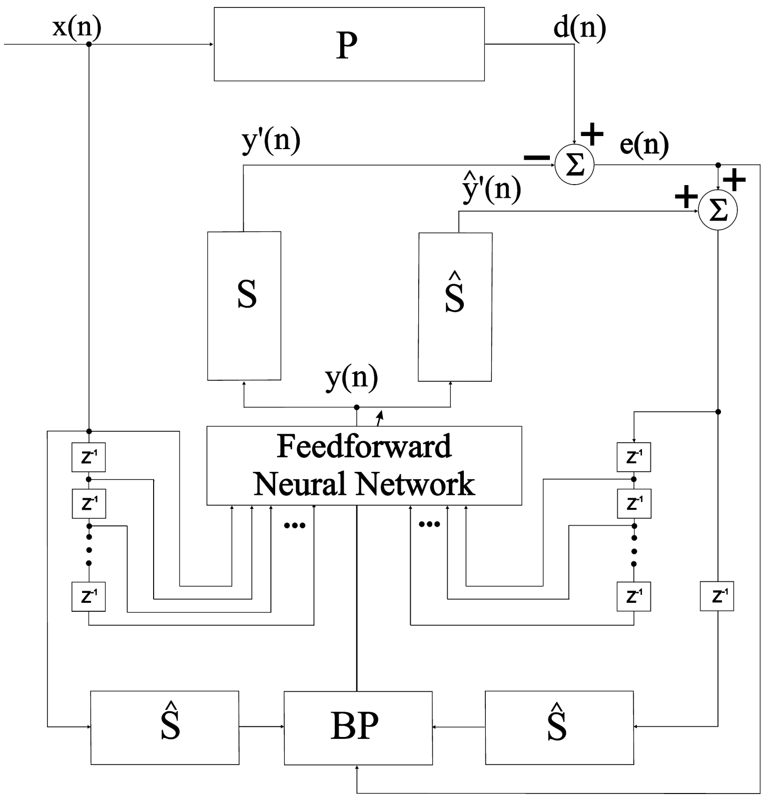

Fig. 3. Structure of the proposed algorithm.

Equation (9) is used to estimate the desired signal:

$$\widehat{d}(n) = e(n) + \hat{\mathbf{y}}' \tag{10}$$

and $\mathbf{y}(n) = [y(n), y(n-1), ..., y(n-S+1)]$. The network input vector is composed according to (3):

$$\mathbf{u}(n) = \left[\mathbf{x}\,(n)\,, \widehat{\mathbf{d}}(n-1)\right],$$

$$\mathbf{x}(n) = \left[x(n), x(n-1), ..., x\left(n - \frac{I}{2} + \alpha + 1\right)\right], \tag{11}$$

$$\hat{\mathbf{d}}(n) = \left[\widehat{d}(n), \hat{d}(n-1), ..., \hat{d}\left(n - \frac{I}{2} + \beta + 1\right)\right],$$

$|\alpha|$, $|\beta| \in \{0, 1, 2..., A\}$, $\alpha = -\beta =$ const. If I is odd, $A = \dfrac{I+1}{2}$, otherwise $A = \dfrac{I}{2}$. The weights update procedure is described below. The procedure is based on an error backpropagation algorithm. The objective of the procedure is to minimize the instantaneous square error:

$$E(n) = \frac{1}{2}e^2\left(n\right) = \frac{1}{2}\left(d(n) - y'_m(n)\right)^2. \tag{12}$$

This task is achieved by finding a minimum in a space of network weights, e.g.:

$$\frac{\partial E(n)}{\partial W} = 0, \tag{13}$$

where W is the total weight of the network. In order to update the weights belonging to a hidden layer, the direction of change of the output layer needs to be known first. Applying the chain rule to the left-hand side of (13) in accordance to Fig. 3:

$$\frac{\partial E(n)}{\partial \mathbf{w}(n)} = \frac{\partial E(n)}{\partial \mathbf{e}(n)}\,\frac{\partial \mathbf{e}(n)}{\partial \mathbf{y}'(n)}\,\frac{\partial \mathbf{y}'(n)}{\partial \mathbf{y}(n)}\,\frac{\partial \mathbf{y}(n)}{\partial \mathbf{v}(n)}\,\frac{\partial v(n)}{\partial \mathbf{w}(n)} = \nabla\frac{\partial v(n)}{\partial \mathbf{w}(n)}. \tag{14}$$

Using (12), firstly the factor on the right-hand side of (13) is equal to $e(n)$. Because of (10), the second factor of (14) is $\widehat{\mathbf{S}}$. The fourth factor is equal to a because of (7) and according to (4), the last factor is equal to $\mathbf{y}_H(n)$. ∇ is defined as the local gradient of the output layer.

$$\frac{\partial E(n)}{\partial \mathbf{w}(n)} = \nabla\left(n\right) y_H(n) = a\widehat{\mathbf{S}}\mathbf{e}^{\mathrm{T}}\left(n\right)\mathbf{y}_H(n). \tag{15}$$

The error vector has the form of: $\mathbf{e}(n) = [e(n), e(n{-}1), ..., e(n{-}N{+}1)]$. Now the output layer weights update formula can be written:

$$\mathbf{w}(n+1) = \mathbf{w}(n) + \eta(n)a\widehat{\mathbf{S}}\mathbf{e}^{\mathrm{T}}\left(n\right)\mathbf{y}_H\left(n\right), \tag{16}$$

$\eta(n)$ is the learning rate of the output layer. The gradient of the hidden layer is obtained by applying the chain rule to the left-hand side of (15), except $\mathbf{w}(n)$ which is replaced by $\mathbf{W}(n)$,

$$\begin{aligned}\nabla_H &= \operatorname{diag}\left[f'_H(\mathbf{v}_H(n))\right]\left[\mathbf{w}(n)\nabla(n)\right]^{\mathrm{T}} \\ &= \operatorname{diag}\left[f'_H(\mathbf{v}_H(n))\right]\left[\mathbf{w}^L(n)a\widehat{\mathbf{S}}\mathbf{e}^{\mathrm{T}}\left(n\right)\right]^{\mathrm{T}}.\end{aligned} \tag{17}$$

The weights update formula for the hidden layer can be written as:

$$\mathbf{W}(n{+}1) = \mathbf{W}\left(n\right) + \eta_H(n)\nabla_H\mathbf{u}\left(n\right). \tag{18}$$

$\eta_H(n)$ is the learning rate of the output layer. Using (11) in (18) we obtain:

$$\mathbf{W}(n{+}1) = \mathbf{W}\left(n\right) + \eta(n)\operatorname{diag}\left[f^i_H(\mathbf{v}_H(n))\right]\mathbf{w}a\mathbf{e}\left(n\right)\widehat{\mathbf{S}}[\mathbf{x}\left(n\right), \widehat{\mathbf{d}}\left(n{-}1\right)]. \tag{19}$$

According to Eq. (10), Eq. (19) can be rewritten as:

$$\mathbf{W}(n+1) = \mathbf{W}(n) + \eta_H(n)\mathrm{diag}\left[f_H^i(\mathbf{v}_H(n))\right] \cdot \mathbf{w}a\mathbf{e}(n)\,\widehat{\mathbf{S}}[\mathbf{x}(n), (e(n) + \widehat{\mathbf{S}}\mathbf{y}_n^{\mathrm{T}}(n))]. \tag{20}$$

The learning rate of the hidden and output layers is changing during the weights update process. The use of an adaptive learning rate improves the convergence of the algorithm (Duch *et al.*, 2000). Learning rate is an exponential function of the error signal:

$$\eta(n) = \eta_{\max} - (\eta_{\max} - \eta_{\min})\exp\left[-\gamma E(n)\right], \tag{21}$$

$\eta_{\max}$ and $\eta_{\min}$ are maximum and minimum values of the learning rate respectively. These values depend on the length of $\mathbf{u}$ and the number of neurons in a hidden layer (Osowski, 2000).

4. Numerical simulations

The QNBPN algorithm is compared with FXBPNN in this section. The structure of the neural network was as described in Sec. 2. Values describing the dimension of the input vector and the number of neurons in a hidden layer were $I = 64$ and $J = 64$ respectively. Sampling frequency was set to 16 kHz.

The secondary path was modelled on a 25th order band-pass FIR filter with centre frequency of 600 Hz (Fig. 4). The nonlinear primary path was created using the Wiener nonlinear model. The linear part of the Wiener model was modelled on a 25th order low-pass FIR filter with cutoff frequency of 800 Hz (Fig. 5). The nonlinear part of the Wiener model was $g(x) = 2.5x^3$. The reference signal was a sine wave of frequency $f = 50$ Hz. The error signal of the QNBPN algorithm is shown in Fig. 6.

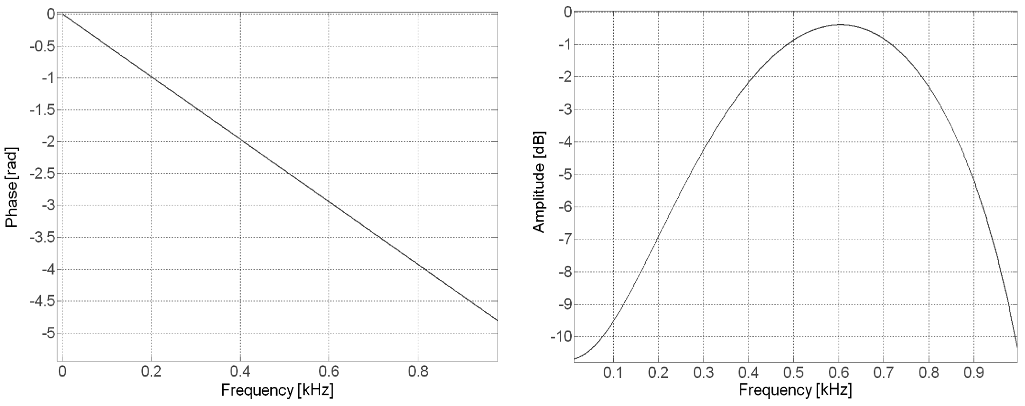

Fig. 4. Frequency response of secondary path: phase – left, amplitude – right.

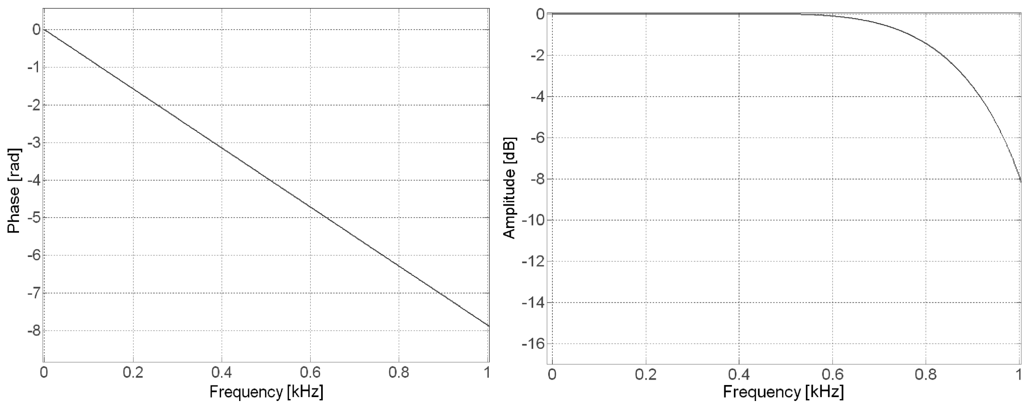

Fig. 5. Frequency response of linear part of primary path: phase – left, amplitude - right.

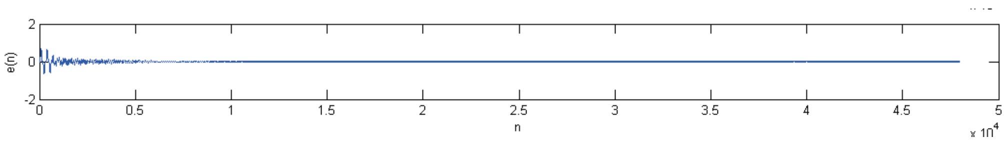

Fig. 6. Error signal of the QNBPN algorithm.

The power spectrum of the error signal is shown in Fig. 7. The FXBPNN algorithm was simulated using the same network as that used for the QNBPN algorithm. The network was driven by the reference signal only.

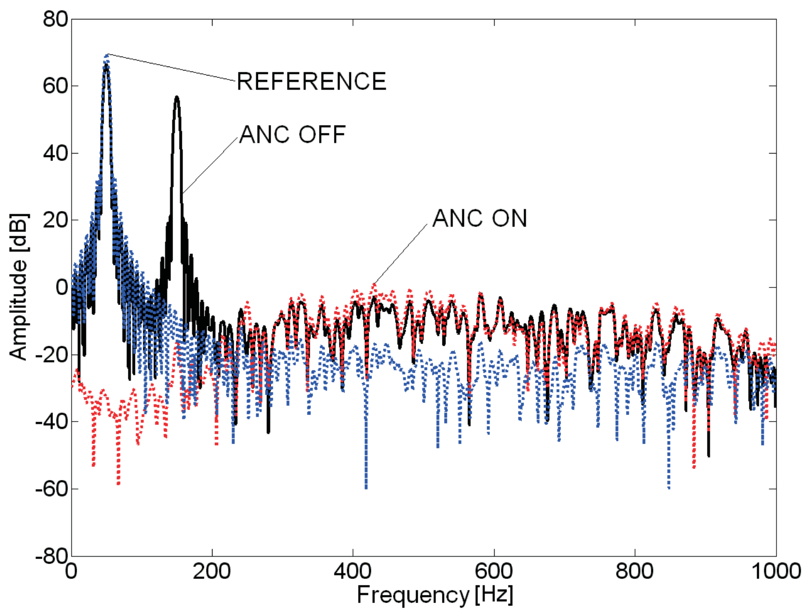

Fig. 7. Error power spectrum of the QNBPN ANC algorithm.

The error signal and its power spectrum in FXBPNN are shown in Fig. 8 and Fig. 9 respectively.

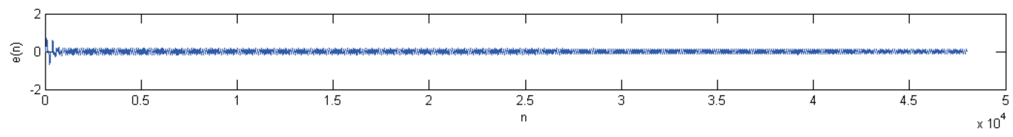

Fig. 8. Error signal of the FXBPNN algorithm.

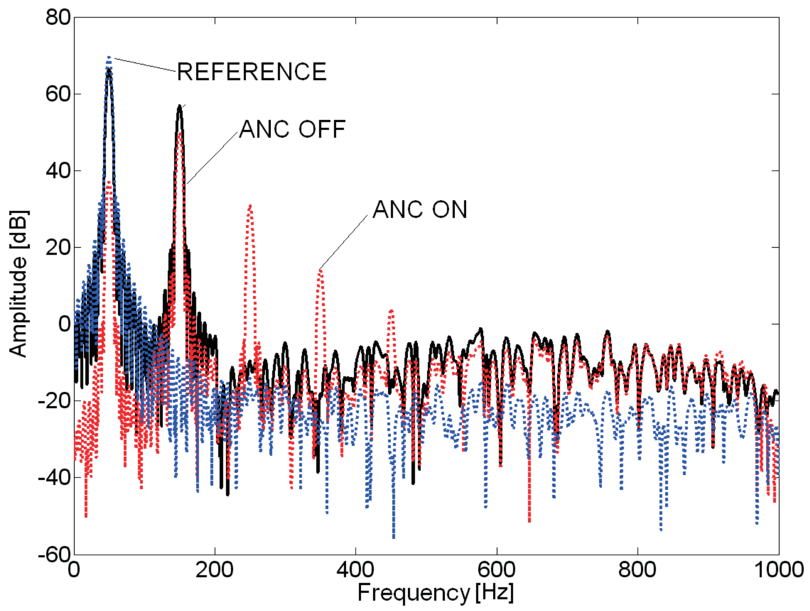

Fig. 9. Error power spectrum of the FXBPNN algorithm.

Both algorithms converged. However, the maximum value of the mean square error of the QNBPN algorithm was much smaller than the mean square error of the FXBPNN algorithm. Power spectrums show that the proposed algorithm was able to reduce harmonic frequency in contrast to FXBPNN.

5. Conclusion

This paper has described an active noise control algorithm based on feedforward neural networks and the NARMAX system identification model. The proposed algorithm (QNBPN) is able to converge when the primary path exhibits nonlinear behaviour. The efficiency of QNBPN is greater than that of FXPBNN. Since the size and the structure of the neural networks for the proposed and

FXBPNN algorithms are identical, the computational load of the proposed algorithm is not much higher than that of FXBPNN. A physical nonlinear primary path was not specified, therefore further laboratory experiments need to be carried out.

Acknowledgments

This publication has been prepared on the basis of the results of a research task carried out within the scope of the first stage of the National Programme “Improvement of safety and working conditions”, partly supported in 2008–2010 – within the scope of state services – by the Ministry of Labour and Social Policy, and – within the scope of research and development – by the Ministry of Science and Higher Education. The Central Institute for Labour Protection – National Research Institute is the Programme’s main coordinator.

References

1. Bouchard M., Paillard B., Dinh C. (1999), *Improved training of neural networks for the nonlinear active control of sound and vibration*, IEEE Trans. on Neural Networks, **10**, 2, 391–401.
2. Chen S., Billings S. (1989), *Representation of nonlinear systems: The NARMAX model*, International Journal of Control, **49**, 3, 1013–1032.
3. Cybenko G.V. (1989), *Approximation by Superpositions of a Sigmoidal Function*, Mathematics of Control, Signals and Systems, **2**, 4, 303–314.
4. Duch W., Korbicz J., Rutkowski L., Tadeusiewicz R. (2000), [in Polish: *Sieci neuronowe*], Exit, Warszawa.
5. Elliott S. (2001), *Signal Processing for Active Control*, Academic Press, London.
6. Engel Z., Nizioł J. (1995), [in Polish: *Perspektywy rozwoju aktywnych metod redukcji hałasu i wibracji*], Materiały II Szkoły Metody Aktywne Redukcji Drgań i Hałasu, pp. 11–24.
7. Hansen C.J., Snyder S.D. (1997), *Active Control of Noise and Vibration*, E&FN Spoon.
8. Haykin S. (1994), *Neural Networks, a Comprehensive Foundation*, Macmillan, New York.
9. Morzyski L., Makarewicz G. (2003), *Application of neural networks in active noise reduction systems*, International Journal of Occupational Safety and Ergonomics, **9**, 3, 257–270.
10. Narendra K., Parthasarathy K. (1990), *Identification and control of dynamical systems using neural networks*, IEEE Trans. on Neural Networks, **1**, 1, 4–27.
11. Nelles O. (2000), *Nonlinear System Identification: From Classical Approaches to Neural Networks and Fuzzy Models*, Springer.
12. Osowski S. (2000), [in Polish: *Sieci neuronowe do przetwarzania informacji*], Oficyna Wydawnicza Politechniki Warszawskiej, Warszawa.

13. Snyder S., Tanaka N. (1995), *Active control of vibration using a neural network*, IEEE Trans. on Neural Networks, **6**, 1, 819–828.

14. Strauch P., Mulgrew B. (1999), *Active control of nonlinear noise processes in a linear duct*, IEEE Trans. on Signal Processing, **46**, 2404–2412.

15. Zhou Y. *et al.* (2005), *Analysis and DSP implementation of an ANC system using a filtered error neural network*, Journal of Sound and Vibration, 285, 1–25.