Neural Network Based Estimation of Feedback Signals for a Vector Controlled Induction Motor Drive

M. Godoy Simões, Student Member, IEEE, and Bimal K. Bose, Fellow, IEEE

Abstract—Neural networks are recently showing good promise for application in power electronics and motion control systems. So far, they have been applied for a few cases, mainly in the control of converters and drives, but their application in estimation is practically new. The purpose of this paper is to demonstrate that such a technology can be applied for estimation of feedback signals in an induction motor drive with some distinct advantages when compared to DSP based implementation. A feedforward neural network receives the machine terminal signals at the input and calculates flux, torque, and unit vectors (cos θ and sin θ) at the output which are then used in the control of a direct vector-controlled drive system. The three-layer network has been trained extensively by Neural Works Professional II/Plus program to emulate the DSP-based computational characteristics. The performance of the estimator is good and is comparable to that of DSP-based estimation. The system has been operated in the wide torque and speed regions independently with a DSP-based estimator and a neural network-based estimator, and are shown to have comparable performance. The neural network estimator has the advantages of faster execution speed, harmonic ripple immunity, and fault tolerance characteristics compared to DSP-based estimator.

I. INTRODUCTION

ADJUSTABLE speed vector-controlled induction motor drives have found wide popularity for high-performance motion control applications. Since the control and feedback signal processing for the drives are very complex, a powerful microcomputer or digital signal processor (DSP) is invariably used for the computation. In the past few years, neural networks have been used in some power electronic applications, such as inverter current regulation [5], dc motor control [6] [7], flux estimation [8], and observer based control of induction machines [9]. Evidently, neural network technique is showing promise as a competitive method of signal processing for power electronics applications. Neural networks have the advantages of extremely fast parallel computation [10], [11]. Immunity from input harmonic ripple, and fault tolerance characteristics due to distributed network intelligence. It can perform dedicated signal processing function and can easily coexist as a peripheral chip with a system control DSP.

A neural network consists of many neurons or processing elements interconnected to constitute a parallel neuro-computing network. Each neuron, again, can be modeled as an op-amp summer-like configuration where the output is transmitted and squashed (or limited) through a nonlinear transfer function. So far, neural network has been applied for a few cases mainly in the control of converters and drives, but its application in estimation, particularly with time-varying input signals, is practically new. The present work explores feedforward neural network technique for estimation of feedback signals of a direct vector-controlled (DVC) induction motor drive.

Fig. 1 shows the block diagram of a DVC drive system where the estimation of feedback signals, such as rotor flux (ψr), unit vectors (cos θr, sin θr) and torque (Tr) are indicated by a DSP as well as a neural network. The estimated torque can be used in an additional feedback loop within the speed control loop, if desired. In a DVC drive system, the feedback signals are calculated from the machine terminal voltages and currents by using the following equations [3]:

\[
\psi_{ds} = \int (\psi_{ds} - R_e \cdot i_{ds}) \, dt
\]

\[
\psi_{qs} = \int (\psi_{qs} - R_e \cdot i_{qs}) \, dt
\]

\[
\psi_{qm} = \psi_{qs} - L_i \cdot i_{qm}
\]

\[
\psi_{dm} = \psi_{ds} - L_i \cdot i_{dm}
\]

\[
\psi_{qr} = \frac{L_r}{L_m} \cdot \psi_{qm} - L_r \cdot i_{qs}
\]

\[
\psi_{dr} = \frac{L_r}{L_m} \cdot \psi_{dm} - L_r \cdot i_{qs}
\]

\[
T_r = \frac{3P}{4} \cdot (\psi_{ds} \cdot i_{ds} - \psi_{qs} \cdot i_{qs})
\]

\[
\psi_r = \sqrt{\psi_{qr}^2 + (\psi_{dr})^2}
\]

\[
\cos \theta_r = \frac{\psi_{dr}}{\psi_r}
\]

\[
\sin \theta_r = \frac{\psi_{qr}}{\psi_r}
\]


M. G. Simões is on leave of absence from the University of São Paulo, Brazil. He is currently with the Department of Electrical Engineering, University of Tennessee, Knoxville, TN 37996 USA.

B. K. Bose is with the Department of Electrical Engineering, University of Tennessee, Knoxville, TN 37996 USA.

IEEE Log Number 9409293.
where

- $v_{dq}^s$ is the stator voltage in $d$-axis ($q$-axis),
- $i_{dq}^s$ is the stator current in $d$-axis ($q$-axis),
- $\psi_{dq}^s$ is the stator flux linkage in $d$-axis ($q$-axis),
- $\psi_{dm}^s$ is the airgap flux linkage in $d$-axis ($q$-axis),
- $\psi_{tr}^s$ is the rotor flux linkage in $d$-axis ($q$-axis),
- $R_s$ is the stator resistance,
- $L_{ls}$ is the stator leakage inductance,
- $L_{lr}$ is the rotor leakage inductance,
- $L_m$ is the magnetizing inductance,
- $L_r$ is the rotor inductance, and
- $P$ is the number of poles.

All the signals with superscript $s$ indicate that they are in stationary reference frame. The integrations in (1) and (2) can be merged with the low-pass filter (LPF), as indicated in Fig. 1, by using very low corner frequency, and can easily be implemented by dedicated hardware (op-amps). The reason for merging the integration function with the LPF hardware is that the feedforward neural network, used in the present application, is basically a pattern recognition network and cannot perform directly any dynamic function. In Fig. 1, both the DSP and neural network receive the variable frequency variable magnitude signal waves $\psi_{tr}^s$, $\psi_{dq}^s$, $i_{dq}^s$, and $v_{dq}^s$, and then compute (3)–(10) to estimate rotor flux, unit vectors, and torque, as shown. The DSP-based estimator output is used for comparison of the neural network based estimator performance.
II. NEURAL NETWORK ESTIMATION PRINCIPLE

The structure of the proposed feedforward neural network used for estimation is indicated in Fig. 2. The network has three layers, i.e., input layer, hidden layer, and the output layer. The circles in the network represent the neurons. The input and output layers have neurons equal to the respective number of signals, whereas the hidden layer in the present design has 20 neurons. The topology can be defined as four-twenty-four network. The network is fully connected, i.e., the output of each neuron is connected to all the neurons in the forward layer through a weight which is not shown in the figure (see Fig. 4). Besides, a bias signal is coupled to all the neurons of the hidden and output layers (not shown) through a weight.

The input layer neurons have linear transfer characteristics, as indicated. On the other hand, the hidden layer and output layer neurons have hyperbolic-tan type nonlinear transfer function given as

\[ \Phi(I) = \frac{e^{\alpha I} - e^{-\alpha I}}{e^{\alpha I} + e^{-\alpha I}} \]  

where \( \alpha \) is the gain. The transfer function is bipolar, monotonic, differentiable, and has the largest gain (\( \alpha \)) at zero signal. Fig. 3 shows the plot of the transfer function with adjustable slope, and Fig. 4 shows the structure of the neural element that incorporates the transfer function.

The weight \( w \) in series with the transfer function \( \Phi \) helps adjusting its slope, as indicated in Fig. 3. The nonlinear transfer function associated with the neurons gives nonlinear mapping property of the network and helps performing highly nonlinear computations, such as multiplication, division, and square-rooting, besides addition and subtraction, as indicated in (3)-(10). The input variable frequency variable magnitude near-sinusoidal signals are converted to per-unit form through the normalizer gains, and then after computation, the output is brought back to actual values through the denormalizer gains, as indicated.

III. TRAINING PROCEDURE

The feedforward neural network is usually trained by a back-propagation training algorithm first proposed by Rumelhart, Hinton, and Williams in 1986. The distributed weights in the network contribute to the distributed intelligence or “associative memory” property of the network. With the network initially untrained, i.e., with the weights selected at random, the output signal pattern will totally mismatch the desired output pattern for a given input pattern. The actual output pattern is compared with the desired output pattern and the weights are adjusted by the supervised back-propagation training algorithm until the pattern matching occurs, i.e., the pattern errors become acceptably small. For the input pattern \( p \), the squared output error for all output-layer neurons of the network is given as

\[ E_p = \frac{1}{2} (d_p - y_p)^2 = \frac{1}{2} \sum_{j=1}^{S} (d_{pj} - y_{pj})^2 \]  

(12)
where $d_p^o$ is the desired output of the $j$th neuron in the output layer, $y_p^o$ is the corresponding actual output, $S$ is the dimension of the output vector, $y_p^o$ is the actual net output vector, and $d_p^o$ is the corresponding desired output vector. The total squared error $E$ for the set of $P$ patterns is then given by

$$E = \sum_{p=1}^{P} E_p = \frac{1}{2} \sum_{p=1}^{P} \sum_{j=1}^{S} (d_{p}^{o} - y_{p}^{o})^2.$$  \hfill (13)

The weights are changed to reduce the cost functional $E$ to a minimum value by gradient descent method. The weight update equation is then given as

$$W_{ij}(t+1) = W_{ij}(t) + \eta \left( \frac{\partial E_p}{\partial W_{ij}(t)} \right)$$  \hfill (14)

where $\eta$ is the learning rate, $W_{ij}(t+1)$ is the new weight, and $W_{ij}$ is the old weight. The weights are iteratively updated for all the $P$ training patterns. Sufficient learning is achieved when the total error $E$ summed over the $P$ patterns falls below a prescribed threshold value. The iterative process propagates the error backward in the network and is therefore called the back-propagation algorithm. To be sure that the error converges to a global minimum but does not get locked up in a local minimum, a momentum term $\beta[W_{ij}(t) - W_{ij}(t-1)]$ is added to the right of (14). Further improvement of the back-propagation algorithm is possible by making the learning rate step-size adaptive, i.e.,

$$\eta(t+1) = u\eta(t), \quad \text{with} \quad u < 1$$  \hfill (15)

so that the oscillation becomes minimal as it settles to the global minimum point. The network training is highly automated and is usually performed off-line through a PC-based simulator program, such as NeuralWorks Professional II/PLUS [13]. During the iterative procedure for training the network, the neural network simulator program stops when the error $E$ falls below a prescribed threshold value or when a limit number of iterations is reached. The particular simulator program has a graphical interface where all the set-ups are done via dialog boxes. It is possible to add some visual instruments like an rms error gauge, weight histogram, and confusion matrices for helping the user to see whether the training is converging. The simulator can use ASCII, binary or commented ASCII files for training. The network topology can be edited to change the connections, delete weights, or to implement different transfer functions. The training procedure used in the present project can be summarized as follows:

- Simulate the induction motor drive system, as shown in Fig. 1, by PC-SIMNON.
- Generate the input/output data table for different operating conditions with the help of PC-MATLAB. The
input data correspond to the machine terminal variables \( \psi_{d}^* \), \( \psi_{q}^* \), \( i_{d}^* \), and \( i_{q}^* \), and the output data correspond to the desired estimated signals derived by solving (3)-(10).

- Convert the data table in per unit form and feed to the NeuralWorks Simulator program located in PC.
- The simulator assigns small but random weights initially to the network.
- An input/output data pair is selected from the table. For the given input data pattern, calculate the network output and compare with the desired data output to derive the error pattern.
- From the error pattern, compute and adjust the network weights by the back-propagation algorithm so that the new error is small.
- Repeat the above steps with each set of input/output data patterns until the rms error for the entire training set converges below the desired threshold value.
- After completion of the training, test the network performance with arbitrary input pattern to ensure successful training.

The initial learning rate \( \eta \) and the momentum gain \( \beta \) in the present project were set to 0.1 and 0.4, respectively. The neural network simulator program has an automatic scheduling to lower these coefficients as the network converges. The convergence threshold error level was set to 0.01, and the maximum number of iteration steps was 15 million. The design and training of a neural network for satisfactory performance requires a very time consuming iterative procedure with a large training data table. Fortunately, the simulator program highly automates the training procedure. The selection of hidden layer neurons may require several stages of iteration. If the number is small, the error will not converge to the satisfactory level. Again, if the number is too large, the network will tend to memorize (look-up table function)
rather than learn. After satisfactory training with the help of the simulator program, the weights are down-loaded to the prototype network.

IV. ESTIMATOR PERFORMANCE

In the beginning, it was decided to estimate the torque ($T_e$) only with a simple four-five-one network to validate the feasibility of estimation. A 5-hp DVC induction motor drive system, shown in Fig. 1, with pure inertia load was simulated with the flux and speed control loops open but with a constant $i_{s_y}$ (i.e., constant flux). A triangular bi-directional $i_{s_y}$ profile was injected to generate the proportional torque profile, as indicated in Fig. 5. As a result, the machine speed varies correspondingly generating variable frequency variable magnitude $\psi_{a}, \psi_{d}, i_{a}, i_{d},$ and $i_{s_y}$ waves.

An input/output data table (with 10000 points) was used to train the network. The network performance, although somewhat crude in comparison with the actual torque, indicates...
the correct trend for estimation. The rms threshold error was reasonably low in spite of the large output ripple which was found to be present even at constant torque condition (see torque step in Fig. 10(a)). The ripple amplitude can be attenuated to some extent with a low pass filter of small time constant. Fig. 6 shows improvement of the estimator performance when the hidden layer neurons were increased from five to ten.

Being encouraged with the feasibility of estimation and performance improvement with a larger number of hidden layer neurons, it was decided to design the full-fledged estimator with four-twenty-four network, as shown in Fig. 2. There is no unique way to determine the number of hidden layers and the number of neurons in each hidden layer. It may be mentioned also that the neural network technique gives only approximate estimation. Of course, the estimation accuracy can be improved by increasing the number of hidden layers, the number of neurons in each hidden layer, or increasing the training time. To some extent, it involves trial-and-error iteration procedure. After the neural network is trained, it is expected that the input–output characteristics of the network are to be the same as those of the input–output characteristics of the physical system being modeled. This is true if the system has been trained dynamically over the full range with a sufficient number of training examples. Since the network does not use any delayed input or delayed feedback, the outputs are dependent upon the instantaneous inputs. Such a gain-like characteristic does not introduce any additional dynamic stability problem other than the intrinsic system stability. Fig. 7(a)–(d) show, respectively, the torque, flux, cosθ, and sinθ, output of the estimator after successful training of the network with a very large number of data sets. As before, the drive system operates with variable frequency but with a constant PWM carrier frequency of 15 kHz. Therefore, the input $i_d^*$ and $i_q^*$ current waves are reasonably harmonic-free. The ripple generated by the network is evident on torque and flux, but the unit vector signals are clean. Again, the estimator ripple frequency has some correspondence with the motor fundamental frequency. For example, at a time below 0.2 s and above 0.8 s in Fig. 7, the fundamental frequency is very low, showing the corresponding low ripple frequency on the estimated torque. The overall performance of the neural network based estimator appears to be very encouraging. Fig. 8 shows the estimator performance for the same conditions as above, except the switching frequency is low (2 kHz). The harmonic ripple immunity feature for neural network based estimation can be observed in the results because the raw input waveforms at 2 kHz contain considerable amount of ripple which do not reflect at the output.

This noise-immune performance of the neural network is particularly important for drive system feedback signals estimation with low switching frequency, such as OTO inverters. Otherwise, any attempt to reduce the ripple by a low pass filter will cause fundamental frequency sensitive phase lag which may not be acceptable. Fig. 9 shows the validity of flux estimation in the constant torque as well as in field-weakening regions.

V. SYSTEM SIMULATION STUDY AND PERFORMANCE COMPARISON

Once the neural network based estimator performance was found to be satisfactory, it was decided to operate the drive system of Fig. 2 with closed loop flux and speed control independently with a DSP-based estimator and a neural network based estimator to compare the system performances. The inverter switching frequency was maintained at 15 kHz. Two sets of tests were carried out: (a) operate the system at constant speed but at stepped load torque, and then (b) at free acceleration/deceleration mode with pure inertia load. The performances at both the conditions with the neural network based estimator were found to be excellent. Fig. 10 shows the performance comparison with stepped load torque with the set speed of 1195 r/min where the left column indicates the DSP-based performance whereas the right side indicates the neural network based performance. The performances match very well, but the ripple with neural network based estimator was somewhat large. The average accuracy of estimation was found to be better than 1%. Evidently, the torque and flux
ripples have little effect on the drive because of inertia filtering. Fig. 11 shows the performance comparison of all the signals when the speed was cycled with a bi-directional profile in the constant torque mode. The performances were found to match very well at all the speeds. The drive also performs well in the field-weakening range, although no results are shown.

After validation of neural network based estimation, it has to be implemented in a practical drive system. A neural network can be implemented by dedicated hardware (such as analog IC chip Intel 80170NX or digital IC chip Micro Device MD1220NBS), a concurrently operating processor (such as INMOS transputer type T800), or can be integrated with the central processor system. Dedicated hardware implementation has the inherent fault tolerance advantage because a few missing connections or erroneous weights will not substantially affect the accuracy of estimation. Besides, parallelism of dedicated hardware computation makes it extremely fast compared to central processor based computation. Of course, concurrent processor based computation is a compromise in speed between the above implementation methods. In summary, the neural network approach provides the advantages of learning input-output relations, fast execution speed, fault
VI. CONCLUSION

The paper successfully demonstrates the application of neural networks in the estimation of feedback signals for a vector-controlled induction motor drive system. A three-layer feedforward neural network of the structure four-twenty-four has been trained with the Neural-Works Professional II/PLUS Simulator program using the variable frequency variable magnitude input/output data from a simulated drive system, and the performance of the estimator was found to be excellent in the wide torque and speed regions. Although the estimator performance was demonstrated for a direct vector-controlled induction motor drive, it can be extended to scalar or vector control (direct or indirect) of any type of drive system. The estimator can be implemented either by dedicated hardware or by microprocessor software. The dedicated hardware estimator can easily coexist with the system control DSP and relieve its computational burden. The neural network has distinct advantages when compared to the conventional DSP-based estimator and promises to be the future choice for application in industrial drives.

REFERENCES


Bimal K. Bose (S’59-M’60-SM’78-F’89) received the B.E. degree from Calcutta University, India, the M.S. degree from the University of Wisconsin, Madison, and the Ph.D. degree from Calcutta University in 1956, 1960, and 1966, respectively.

Early in his career, he served as a faculty member at Bengal Engineering College, India, for 11 years. In 1971, he joined the faculty at Rensselaer Polytechnic Institute, Troy, NY. In 1976, he joined General Electric Corporate Research and Development, Schenectady, NY, as an Electrical Engineer, where he worked for 11 years. He currently holds the Condra Chair of Excellence in Power Electronics at the University of Tennessee, Knoxville, where he is responsible for organizing the power electronics teaching and research program for the last seven years. He is also the Distinguished Scientist (formerly Chief Scientist) of EPRI-Power Electronics Applications Center, Knoxville, Honorary Professor of Shanghai University, China, and Senior Advisor of Beijing Power Electronics Research and Development Center, China. His research interests are power converters, ac drives, microcomputer control, and application of expert systems, fuzzy logic, and neural networks in power electronics. He has published more than 100 papers, authored and edited four books, and holds 18 U.S. patents.

Dr. Bose, for his research contributions, was awarded a Premchand Roychhand scholarship and Mouat gold medal by Calcutta University in 1968 and 1970, respectively. He has received the Publication Award, the Silver Patent Medal, and the Centennial Invention Disclosure recognition from General Electric, and a number of IEEE Prize Paper Awards. In 1993, he was awarded the IEEE Industry Applications Society’s Outstanding Achievement Award “for contribution in the application of electricity to industry.” In 1994, he was awarded the IEEE Industrial Electronics Society’s highest Dr. Ing. Eugene Mittelmann Achievement Award “in recognition of outstanding contributions to research and development in the field of Power Electronics and a lifetime achievement in the area of motor drives,” and the IEEE Region 3 Outstanding Engineer Award “for contribution to power electronics and drives technology.”

M. Godoy Simões (S’89) received the B.S. and M.S. degrees from the Escola Politécnica da Universidade de São Paulo, Brazil, in 1985 and 1990. He has been an assistant professor at the Universidade de São Paulo since 1989, and is currently on leave of absence to pursue the Ph.D. degree at the University of Tennessee, Knoxville. His research interests include fuzzy logic and neural networks applications to power electronics, drives, and machines control.