Sliding mode neural network control of an induction motor drive

Gustavo G. Parma*,†Benjamim R. Menezes‡, Antônio P. Braga§ and Marcelo A. Costa*

Departamento Engenharia Eletrônica (UFMG), C.P. 209, 30.161-970, Belo Horizonte, MG, Brazil

SUMMARY
A new approach for induction motor drive control is presented in this paper. The new scheme is based on the direct application of an artificial neural network, trained with sliding mode control, into the feedback control system. Neural network learning is implemented with an on-line adaptation algorithm that inherits robustness and high speed of learning from Sliding Mode Control. The results showed that proportional and integral or proportional, integral and differential controllers used in classical motor drives can be replaced with a neural network with on-line learning. Copyright © 2003 John Wiley & Sons, Ltd.

1. INTRODUCTION

In control applications artificial neural networks (ANNs) usually play the role of an observer, a reference model or a specialized control subsystem. In induction machines, torque and speed control can be implemented by identifying first system dynamics, which is a very challenging task. Although the induction machine model is well known, it is highly non-linear with many of its parameters varying with time and also with operating conditions. Many conventional induction machine control approaches, such as field-oriented control (FOC) [1–3] aim at linearizing the dynamical model. FOC systems have been extensively and successfully applied, but they suffer from sensitivity to parameter variations. In order to overcome this constraint, numerous techniques have been proposed, but they are normally complex and require complete knowledge of machine parameters. In order to overcome some of these difficulties, ANNs have also been applied to the identification and control of induction machines [4].

*Correspondence to: Gustavo G. Parma, Departamento Engenharia Eletrônica (UFMG), C.P. 209, 30.161-970, Belo Horizonte, MG, Brazil.
†E-mail: parma@cpdee.ufmg.br
‡E-mail: brm@cpdee.ufmg.br
§E-mail: apbraga@cpdee.ufmg.br
*E-mail: mcosta@cpdee.ufmg.br

Contract/grant sponsor: FAPEMIG
Contract/grant sponsor: CNPq
Contract/grant sponsor: CAPES
The approach presented in this paper replaces the conventional controllers in FOC systems by ANNs. The main idea is to reduce the number of controllers in order to speed up tuning and add robustness to parameter variation. In order to achieve these features in motor drive control, the on-line learning algorithm is expected to be fast, robust and precise. Our alternative to this is the use of sliding mode control (SMC) [5] in ANNs learning, as described previously by Parma et al. [6]. The algorithm applies control action to the ANN weight update equations, resulting in a robust and fast learning algorithm, fulfilling the basic control requirements of the problem.

Other works that apply SMC to train ANNs have also appeared in the literature. Sabanovic and Jezernik et al. [7, 8] show how SMC can be used to train networks for on-line learning in robot control applications. Parma et al. [6, 9] presented general approaches for on-line and off-line training learning of multi-layer perceptrons and showed that such a training scheme inherits robustness and high speed of learning from SMC. The speedup achieved in learning is a consequence of taking into consideration the derivative of error estimate in the design of the sliding surface, that is also responsible for convergence and stability. Topalov and Kaynak [10] presented a variation of the previous algorithm that prevents network nodes from saturation due to possible large weight updates.

2. SLIDING MODE CONTROL OF ANN TRAINING

Sliding mode control [5] is based on variable structure systems (VSS) in which the control input is switched between two control signals in order to maintain the system on a sliding control surface. Once the control restrictions are achieved the sliding surface itself guarantees system stability. These restrictions define the existence or not of the sliding manifold $S$. In relation to the classical control techniques, SMC is simpler to implement, since only two input control values are required. Despite its similarity with on-off control, system stability is guaranteed by the sliding surface. Another feature of SMC is that control actions are not governed by plant parameters, but by the dynamics imposed by the sliding surface.

In the control scheme presented in this paper, SMC acts indirectly in the plant, by controlling first the neural network learning process. The ANN, trained with error information, acts as the motor drive controller, but, that itself is controlled by SMC. In contrast with other control techniques, the use of SMC demands only that parameter variations and external disturbances are bounded. SMC is designed taking into account the surface $S$ and the control action gains (learning rates), which are defined by the information available for training: training data and network topology. No additional information is needed to design the learning controller. A schematic view of learning with SMC is presented in Figure 1. It can be observed that the proposed algorithm acts like a feedback controller that includes some information from the standard backpropagation algorithm and the sliding mode controller. The weight updates are in fact the control actions applied by sliding mode control backpropagation (SMC-BP) to the network weights.

2.1. The algorithm

In order to fulfil the on-line requirements to control the machine, an algorithm based on SMC [5] is used in this paper. The algorithm applied is generic and can be used to any configuration of
multi-layer ANNs with MCP nodes [11]. This algorithm was first introduced in Reference [6] where further information can be found.

2.1.1. The algorithm. The following symbols are used:

- \( T \): input vector augmented by the bias term;
- \( Y_H \): output vector of the hidden layer;
- \( Y \): network output vector;
- \( Z \): weight matrix for the connections between input and hidden layers;
- \( Z_{ih} \): weight between input node \( h \) and hidden node \( i \);
- \( W \): weight matrix for the connections between hidden and output layers;
- \( W_{ji} \): weight between hidden node \( i \) and output node \( j \);
- \( f_H(\cdot) \): activation function for the hidden nodes;
- \( f(\cdot) \): activation function for the output nodes.

Considering the symbols and definitions above, for a two-layer ANN we obtain the following relations:

\[
Y_H = f_H(R), \quad Y = f(V), \quad R = Z \cdot T, \quad V = W \cdot Y_H,
\]
where \( R \) and \( V \) are, respectively, the linear output vectors of the hidden and output layers. The following equations are used by the algorithm:

For the input layer:

\[
S_j = X_{2j} + C \cdot X_{1j}, \quad C > 0
\]

\[
X_{1j} = Y_{dj} - Y_j, \quad X_{2j} = \frac{\partial X_{1j}}{\partial t}
\]

where \( Y_{dj} \) is the desired output for the output node \( j \).

For the hidden layer, the following sliding surface is defined:

\[
S_H = X_{2H} + C_H \cdot X_{1H}, \quad C_H > 0
\]

\[
X_{1H} = \frac{1}{2} \sum_{k=1}^{p} (Y_{dk} - Y_k)^2, \quad X_{2H} = \frac{\partial X_{1H}}{\partial t}
\]

where \( p \) is the number of network outputs.

The following weight update equations are defined [6, 9] (for \( \alpha > 0 \) and \( \beta = 0 \)):

\[
W_{ji} = \frac{\alpha \cdot \text{sign}(S_j) \cdot |X_{1j}|}{\frac{\partial f(V_j)}{\partial V}} \cdot Y_H
\]
\[
\dot{z}_{th} = \frac{\beta \cdot \text{sign}(S_H) \cdot |X_{1H}|}{\sum_{j=1}^{p} \left( (Y_d - Y_j) \cdot \frac{\partial f(V_j)}{\partial V} \cdot W_j \right)} \cdot \frac{\partial f(R_i)}{\partial R} \cdot T_H
\]  

There are limits for both \( C \) and \( C_H \) (Equations (1) and (3)) that must be guaranteed in order to maintain convergence properties. Although these limits are presented next, further information can be found in Reference [6].

\[
C \geq \max \left\{ -\frac{2}{Y_H} \frac{\partial Y_H}{\partial t} \left| \frac{X_{2j}}{|X_{1j}|} \right| \right\} 
\]  

\[
C_H \geq \max \left\{ -\frac{2}{T_H} \frac{\partial T_H}{\partial t} \left| \frac{X_{2H}}{|X_{1H}|} \right| \right\} 
\]

As we are dealing with a discrete control application, in which there is a sampling interval, the differentiations that appear in Equations (2) and (4) are transformed into a discrete subtraction equation in order to avoid the numerical differentiation.

3. IMPLEMENTATION

The traditional FOC system is depicted in Figure 2. It can be observed that in this system the user must tune five controllers (some systems apply only four). For proportional and integral (PI) and proportional, integral and differential (PID) type controllers, two and three parameters, respectively, need to be tuned for each controller. Moreover, depending on the type of the controller, the machine parameters must be well known, or at least, well bounded. The approach used in this paper is shown in Figure 3.

Note that all controllers in each direction axis were put together into a single ANN controller. The user must now tune only two parameters \((a) \) and \((b) \) for each ANN controller and due to the algorithm employed, these parameters are easier tuned than PI parameters. The algorithm itself is responsible to keep the control, updating network parameters as needed. The induction machine parameters are shown in Table I. The entire system (controllers, machine and measurements) were simulated in a Pentium-II computer. The PWM inverter has a frequency of 8 kHz, data acquisition and control are maintained in a frequency of 4 kHz. The flux observer employed is a Vergheese model [12], but any other could be used, such as the Gopinath observer. The ANN used to control the direct axis variable \((V_{d*})\) has two inputs (flux module and flux...
error), three neurons in the hidden layer and one neuron in the output layer as the direct voltage.
The first ANN that controls the quadrature axis variable \( V_n^q \) has three inputs (flux module, reference speed and measured speed), seven neurons in the hidden layer and one neuron as output, related to the quadrature voltage.

It is well known that in feedback control systems the desired output value for the controller output is not available. So, one could wonder how Equations (2) and (4) can be applied, but, in fact, what we have in Figure 3 is one ANN controlling the machine speed and the other one controlling the machine flux so the errors needed in Equations (2) and (4) are the speed error (for the first ANN) and the flux error (for the second ANN).

4. SIMULATION RESULTS

The results depicted here show that it is possible to use ANN controllers instead of conventional controllers in a FOC system. At the very first time the system is turned on, the initial weights are sampled from a normal distribution with a zero mean. The reference values (speed and flux) are then set and the system starts its normal operation. The results shown in the next section refers to this first startup in which the speed reference is initially set to 150 rad/s and reduced to 50 rad/s at \( t = 7 \) s. A load of 2 Nm is applied at \( t = 3.5 \) s. The reference flux is set to 0.5 Wb as the

Table I. Induction machine parameters.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Power</td>
<td>2 CV</td>
</tr>
<tr>
<td>Poles</td>
<td>2</td>
</tr>
<tr>
<td>Stator resistance</td>
<td>0.995 Ω</td>
</tr>
<tr>
<td>Stator inductance</td>
<td>47.963 mH</td>
</tr>
<tr>
<td>Rotor resistance</td>
<td>0.696 Ω</td>
</tr>
<tr>
<td>Rotor inductance</td>
<td>49.126 mH</td>
</tr>
<tr>
<td>Mutual inductance</td>
<td>45.601 mH</td>
</tr>
<tr>
<td>Inertial momentum</td>
<td>0.018</td>
</tr>
</tbody>
</table>

Copyright © 2003 John Wiley & Sons, Ltd.

nominal value. A long time simulation did not present any disturbance to final results and will not be shown here. Table II shows the training parameters used in this simulation, where ANN 1 and ANN 2 refers to the superior ANN (quadrature axis) and inferior ANN (direct axis) of Figure 3, respectively. The values for $C$ and $C_H$ are due to Equations (7) and (8). Values for $z$ and $\beta$ shown are the medium values of a quite reasonable large range of values within almost 40% of variation, chosen by trial and error.

Figure 4 shows the motor speed and the reference speed. It can be noted that the machine is able to track the reference speed. There is a small spike in $t = 3.5 \text{s}$, when the load is applied, but the controllers show a good dynamic response. The rotor flux is depicted in Figure 5. There was

<table>
<thead>
<tr>
<th>Parameter</th>
<th>ANN1</th>
<th>ANN 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>$z$ (output layer)</td>
<td>0.09</td>
<td>9</td>
</tr>
<tr>
<td>$\beta$ (hidden layer)</td>
<td>0.7</td>
<td>9</td>
</tr>
<tr>
<td>$C$ and $C_H$</td>
<td>8000</td>
<td>10000</td>
</tr>
</tbody>
</table>

Figure 4. Motor speed (reference and simulated).

Figure 5. Observed and reference rotor flux.
not any saturation constraint to the flux in the simulation, in a real case, the flux would not be able to reach the peak value shown in this figure. It is important to note that there is not any change in the flux when the load is applied, but when the reference speed is reduced.

Figure 6 shows the stator flux for fixed axis at 6.5 s < \( t \) < 7.5 s, when the speed reference is reduced. The speed variation when the load is applied is less than 6% and the system promptly responds to this situation. Figure 7 shows the observed electromagnetic torque and the applied load. The figure shows that the ANNs were not able to completely decouple the flux and speed control loop as can be noted in the torque oscillation when the speed reversal occurs (at 7 s) but the overall result is good enough to maintain the control.

The results show that it is possible to use ANN controllers with the employed training algorithm to implement FOC systems. This approach reduces the time needed to tune the conventional controller parameters without losing robustness.

5. DISCUSSIONS AND CONCLUSIONS

A simplified and yet efficient control scheme to replace the classical FOC system has been presented in this paper. The main advantage of the scheme presented is that, when using
classical FOC system, the user has to tune up to five controllers (Figure 2), each one with two or three parameters, depending on whether they are PI or PID. The approach presented makes use of only two ANN controllers, for which the parameters can be obtained easily with on-line learning.

It is important to mention that the control quality can be improved if the training algorithm is employed with adaptive gains as a function of speed and flux references. The training algorithm based on SMC is robust enough to maintain the system controllability using only two ANN controllers. Another important issue is that the results presented refer to the very first system start up, with all ANN weights randomly sampled. If the system is turned off, then the weights are saved and retrieved on the next start up.

Figure 4 shows an overshoot of almost 20% due to the low inertial momentum value of the machine and the prompt controllers’ response to the reference speed input. If a bigger machine was used, the overshoot would be lower but it would not correspond to the real system that is being implemented.

The proposed control scheme needs the flux error to adjust the weights of the ANN used in the flux control loop. Therefore, if the flux observer, which provides the flux module, is not well tuned, the controller may lose track of the system but this is also a common drawback of any classical FOC. The gains of the proposed SMC-ANN controller are the replacement of the PI or PID controllers that are used in classical FOC systems and the reduction of the number parameters to adjust. Tuning PI or PID controllers due to parameter variations or machine replacement is not necessary in the presented approach, since the ANN adapts itself to the new situations.

ACKNOWLEDGEMENTS

The authors are grateful to FAPEMIG, CNPq and CAPES for the financial support for this investigation.

REFERENCES