Control of an Active Magnetic Bearing with Multi-Layer Perceptrons using the Torque Method

Roger Achkar, Michel Owayjan
Department of Computer and Communications Engineering
American University of Science & Technology (AUST)
Beirut, Lebanon
{rachkar, mowayjan}@aust.edu.lb

Abstract—The active magnetic bearing (AMB) presents a solution for all the technical problems of the classical bearing since it ensures the total levitation of a body in space eliminating any mechanical contact between the rotor and the stator. The goal of our work is to show the control efficiency of a magnetic sustention, characterized by its nonlinear model, using neural networks (NN). In this paper a study of NN controller for a magnetic bearing under a computed torque control is presented.

Keywords-active magnetic bearing; back-propagation algorithm; multi-layer perceptrons; multiple hidden layers; neural controller; non-linear systems and modelling; optimization in neural network’s training

I. INTRODUCTION

The manufacturers and the users of industrial revolving machines are often confronted with technical problems such as heating, frictions, vibrations, maintenance, pollution, etc. Moreover, many research showed interest in increasing the speeds of machining of materials, which necessitates adapted equipment. The active magnetic bearing (AMB) presents a solution to these problems since it ensures the total levitation of a body in space by electromagnetic forces, thus eliminating any mechanical contact between the rotor and the stator [1]. The development of a new non-linear model considering the unbalance (disturbance due to the non-coincidence of the geometric axis and the axis of inertia of the rotor) presented this disturbance as an intrinsic variable in the system. It can also be considered as an external additive disturbance [2]. A control law based on passivity was developed; in addition to other types of control such as sliding mode, input-output linearization, and fuzzy controller [3], [4], and [5].

A Multi-Layer Perceptrons (MLP) controller is used when the modeling of the system in question is difficult and involves approximations [6], [7]. The MLP has the advantage of learning, i.e. adapting to new situations even if these situations were not learned by the network during the training phase [8].

Because of the complexity of the AMB, an MLP was added to the system [8]. Two methods of control by artificial neural network (NN) were developed to control the system [9], [10]. In the first method the network has an additive role by adding to the system a corrective vector to minimize the error. In the second method, the parameters of the PID controller are changed using MLP. These techniques were tested in real time on the active magnetic bearing of the Heudiasyc laboratory of the Université de Technologie de Compiègne (UTC), France [10]. Both neural methods applied to the system presented better performance than in the case of the control by classical PID in several terms: energy consumption, robustness, and stability. However, they did not succeed in completely eliminating the unbalance [11].

In this paper a study of neural network (NN) controller for a magnetic bearing under a computed torque control is presented. This new method can be applied to the active magnetic bearing in order to completely eliminate the unbalance.

II. COMPUTED TORQUE METHOD

A. AMB: General Presentation

The AMB, presented at the Heudiasyc laboratory of the UTC, is formed by two plane of control: (Y1, Z1) plane and (Y2, Z2) plane, an X axis in the middle and an asynchronous motor (Fig. 1).

To define the equations of the model we must do different calculations [8].

![Figure 1. Representation of the AMB with asynchronous motor.](image-url)
- The mechanical energy represented by (1) is divided into translation kinetic energy and rotation kinetic energy. The calculation of the energy is made by considering a moving body in space in function of the positions measured by the sensors.

\[ T_m = \frac{m(l^2 + \frac{1}{2}l^2 \dot{\phi}^2)}{2} \]  

- The electrical energy and the mechanical potential energy which indicates the effect of the gravity.
- The dissipated electrical energy.
- We can deduce the D matrix from (2):

\[ D(q) \ddot{\theta} + C(q, \dot{\theta}) \dot{\theta} + \frac{\partial R}{\partial \theta} = B.F \]  

- Deduce the C matrix from the D matrix by using (3):

\[ C_k = \sum_{i=1}^{6} \sum_{j=1}^{6} \left( \frac{\partial d_{kj}}{\partial \theta_i} + \frac{\partial d_{ki}}{\partial \theta_j} \right) \dot{\theta}_i \dot{\theta}_j \]  

Equation (3) will be applied for \( C_1, C_2, C_3, C_4, C_5, \) and \( C_6. \) So, the model contains six mechanical equations and ten electrical equations.

The principle is to calculate, by using \( D \) and the \( C \) matrix, the outputs of the model represented by \( X = [x, y_1, z_1, y_2, z_2]' \), to able to integrate it into the buckled system.

**B. Torque Method**

This method is used in training the neural network in order to minimize the error between an estimated model and the real model of the AMB.

The dynamic equation of AMB is given by

\[ D(q) \ddot{\theta} + h(q, \dot{\theta}) \dot{\theta} + T = u \]  

The control law of computed torque method can be written as

\[ u(t) = \hat{D}(q). v(t) + \hat{h}(q, \dot{\theta}) \]  

Where \( \hat{D}(q) \) and \( \hat{h}(q, \dot{\theta}) \) are the estimations of \( D(q) \) and \( h(q, \dot{\theta}) \), respectively, and \( v(t) \) is given by:

\[ v(t) = \dot{\theta}_d + k_p (q_d - q) + k_v (\dot{\theta}_d - \dot{\theta}) \]  

Combining (4), (5), and (6) yields the closed loop error dynamic equation

\[ \ddot{e} + k_p e + k_v \dot{e} = \hat{D}^{-1} [\Delta D \ddot{\theta} + \Delta h + T] \]  

Where \( \Delta D = D - \hat{D}, \Delta h = h - \hat{h}, \) and \( e = q_d - q. \) In the ideal case where \( \Delta D = \Delta h = 0, \) and \( T = 0, \) (7) becomes the following ideal second order error equation, which is denoted as \( l: \)

\[ l = \ddot{e} + k_p e + k_v \dot{e} = 0 \]  

Since there are always uncertainties, the ideal error response (8) cannot be achieved in general. The neural network controller is introduced to compensate these uncertainties.

The new control law can be written as

\[ u(t) = \hat{D}(q). v(t) + \hat{h}(q, \dot{\theta}) + \tau_n \]  

Where \( \tau_n \) is the NN output.

The corresponding closed loop error system is

\[ \ddot{e} + k_p e + k_v \dot{e} = \hat{D}^{-1} [\tau_t - \tau_n] \]  

We note that:

\[ \tau_t = \Delta D \ddot{\theta} + \Delta h + T \]  

Since the control objective is to generate \( \tau_n \) to reduce \( l \) to zero in (8), we therefore use \( \tau_t - \tau_n \) as the signal for training the NN.

The value of \( \tau_t \) is obtained by calculating the value \( \tau_m \) of the estimated model by:

\[ \tau_t = u - \tau_m \]  

Where:

\[ \tau_m = \hat{D}(q). \dot{\theta} + \hat{h}(q, \dot{\theta}) \]  

The NN control scheme is depicted in Fig. 2.

The NN output \( \tau_n \) cancels out the uncertainties caused by the inaccurate model in the computed torque controller.

![Figure 2. Representation of the neural network controller.](image)
C. Neural Network Compensator Design

The one hidden layer MLP shown in Fig. 3 is used as the compensator. It is composed of an input layer, a nonlinear hidden layer and a linear output layer. Backpropagation method is adopted for the learning of the MLP.

The basic backpropagation algorithm is based on minimizing the error represented by:

\[
T_{\text{e}}(t) = T_i(t) - T_r(t)
\]  

For each sample \( n \), a direct and retrograde calculation is completed.

In the direct path, the output of the network is calculated by multiplying the weight vector by the input vector.

\[
T_{nk}(n) = \sum_j W_{kj}(n) \cdot f_j \left( \sum_i W_{ji}(n) \cdot y_i(n) \right)
\]  

In the back propagation path, the calculation is done in a reverse way. After the direct calculation, the local gradient is calculated in order to adjust the weight vector in each period. The idea is to update the value of the neural network weights in order to minimize the global error:

\[
\xi(n) = \frac{1}{2} \sum_k T_{ek}^2(n)
\]  

- Calculation of the output layer:

The back-propagation algorithm applies a correction \( \Delta W_{kj} \) of weights proportional to

\[
\frac{\partial \xi(n)}{\partial W_{kj}(n)} = -\eta \cdot \frac{\delta_{ek}(n)}{\Delta y_{j}(n)}
\]  

Then

\[
\frac{\partial \xi(n)}{\partial W_{kj}(n)} = T_{ek}(n) \cdot (-1) \cdot y_j(n)
\]  

The correction of weights is given by:

\[
\Delta W_{kj}(n) = -\eta \cdot \frac{\delta_{ek}(n)}{\Delta W_{kj}(n)} = \eta \cdot T_{ek}(n) \cdot y_j(n)
\]  

\[
= \eta \cdot \delta_k(n) \cdot y_j(n)
\]  

Where \( \eta \) is the learning parameter.

And

\[
\delta_k(n) = T_{ek}(n) = T_k(n) - T_{nk}(n).
\]  

\( \delta_k \) is the local gradient.

- Calculation of the hidden layer:

The adjustment of the \( W_{ji} \) weights is written as:

\[
\Delta W_{ji}(n) = -\eta \cdot \frac{\partial \xi(n)}{\partial W_{ji}(n)}
\]  

\[
= -\eta \cdot \frac{\delta_{ek}(n)}{\Delta v_j(n)} \cdot \frac{\partial v_j(n)}{\partial W_{ji}(n)}
\]  

And:

\[
\frac{\partial v_j(n)}{\partial W_{ji}(n)} = y_i(n)
\]  

The local gradient \( \delta_j \) can be written as:

\[
\delta_j(n) = -\frac{\partial \xi(n)}{\partial v_j(n)} = -\frac{\partial \xi(n)}{\partial y_j(n)} \cdot \frac{\partial y_j(n)}{\partial v_j(n)}
\]  

\[
= \frac{\partial y_j(n)}{\partial v_j(n)} = f'(v_j(n))
\]  

\[
\frac{\partial \xi(n)}{\partial y_j(n)} = \frac{1}{2} \sum_k \frac{\partial T_{ek}^2}{\partial y_j(n)}
\]  

\[
= \sum_k T_{ek}(n) \cdot \frac{\partial T_{ek}(n)}{\partial y_j(n)}
\]  

\[
= \sum_k T_{ek}(n) \cdot \frac{\partial T_{ek}(n)}{\partial y_j(n)} \cdot \frac{\partial T_{nk}(n)}{\partial y_j(n)}
\]  

\[
= \sum_k T_{ek}(n) \cdot (-1) \cdot W_{kj}(n)
\]  

\[
= -\sum_k \delta_k(n) \cdot W_{kj}(n)
\]
By replacing (21) and (22) in (20) $\delta_j(n)$ becomes:

$$\delta_j(n) = f'(v_j(n)) \ast \sum_k \delta_k(n) \ast W_{kj}(n) \quad (23)$$

And (19) becomes:

$$\Delta W_{kj}(n) = \eta \ast \delta_j(n) \ast y_j(n) \quad (24)$$

Then the updating equations for the weights of the output layer and hidden layer are respectively:

$$\Delta W_{kl}(n) = \eta \ast \delta_k(n) \ast y_l(n) + \alpha \ast \Delta W_{kl}(n-1) \quad (25)$$

$$\Delta W_{jh}(n) = \eta \ast \delta_j(n) \ast y_h(n) + \alpha \ast \Delta W_{jh}(n-1) \quad (26)$$

Where $\alpha$ is the momentum coefficient.

III. SIMULATIONS AND RESULTS

The model used in simulations is the complicated model that considers the unbalance on two planes of control. The simplifications used in this model are: $\cos \theta = 1$ and $\cos \psi = 1$, $\sin \theta = \theta$ and $\sin \psi = \psi$; Center of gravity of the rotor is at equal distance from the two planes of control, which implies $\delta x = 0$; $\delta y$ and $\delta z$ represents the parameters influencing the existence of the unbalance. These components represent the parameters of the center of inertia compared to the center of gravity (Fig.4), i.e., the disturbance parameters representing the unbalance.

NB: $\theta$ represents the swing angle around the y-axis, $\psi$ represents the swing angle around the z-axis. These angles are represented in Fig. 5.

The performance of the proposed NN controller is investigated in this section. A study of NN controller under a computed torque control is presented for: different values of $D(q)$ and $h(q, \dot{q})$ representing the estimated model; and the limiting values of $\delta y$ and $\delta z$ which can be reached representing the unbalance. Simulation studies for temporal response of the five axes are carried out and compared with the classical PID control schemes.

Three choices of estimated model are highlighted:

A- The model represented by $\hat{D}(q) = I$, $\hat{h}(q, \dot{q}) = -\dot{q}_d$

B- The AMB simplified model ($\delta y = \delta z = 0$)

C- The variable estimated model

For each choice, an optimization of the parameters influencing the training of the neural network is done, and a simulation giving the temporal answers function of time for X, Y1, Z1, Y2, Z2 axes was made.

Table I summarizes the optimized neural network parameters obtained with the three different estimated models used.

For these optimizations and for $\delta y = \delta z = 8 \times 10^{-5} m$, a comparison is made between the temporal responses of the system controlled by the classical PID controller and the NN controller.

NB: The value of $\delta y = \delta z = 8 \times 10^{-5} m$ represents the limiting value which we can reach by increasing $\delta y$ and $\delta z$ without exceeding the maximum limit of the displacement of the rotor.

### Table I. Optimized Neural Network Parameters

<table>
<thead>
<tr>
<th>Model Choice</th>
<th>Number of neurons in the hidden layer</th>
<th>Learning parameter $\eta$</th>
<th>Momentum term coefficient $\alpha$</th>
<th>Number of epochs</th>
<th>Number of examples per epoch</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>10</td>
<td>0.035</td>
<td>0.09</td>
<td>15</td>
<td>501</td>
</tr>
<tr>
<td>B</td>
<td>8</td>
<td>0.00018</td>
<td>0.03</td>
<td>15</td>
<td>501</td>
</tr>
<tr>
<td>C</td>
<td>10</td>
<td>0.03</td>
<td>0.09</td>
<td>15</td>
<td>501</td>
</tr>
</tbody>
</table>
Fig. 6 shows the temporal answers for X, Y1, Z1, Y2, Z2 axes of the system controlled by NN controller with the first estimated model (with the table 1.A parameters), compared with the answers obtained by classical controller, and the desired responses.

According to the X axis, it is to be noted that the overshoot, rising time and response time are the same for the two control methods.

Temporal responses on Y and Z axis shows an important improvement with MLP compared to the system controlled by PID alone. According to Y1 and Y2, responses by NN controller converge to the desired responses when the PID responses diverge. These temporal responses are the result of controlling a system with a large nonlinearity due to the disturbance. The learned system with MLP has the capacity to reduce this disturbance in order to stabilize the system around the desired answers.

A simulation with a number of examples per epocs equal to 1001 corresponding to 0.5 sec interval of time shows that the temporal responses of the system controlled by neural network controller under a computed torque control are stabilized around the desired responses without any oscillations.

Other simulations are made with different estimated models. Figures 7 and 8 shows the temporal answers for X, Y1, Z1, Y2, Z2 axis of the system controlled by NN controller with the second and the third estimated model respectively (with the table 1.B and 1.C parameters), compared with the answers obtained by classical controller, and the desired responses.

According to the X, Y and Z axis, a large improvement is measured with MLP for the temporal responses compared with classical controller. These simulations showed that the neural network is able to improve the response and minimize the value of $\tau$ even for a variable estimated model.

The improvement achieved by the NN controller is clearly demonstrated for different values of $D(q)$ and $h(q, \dot{q})$ representing the estimated model; and for the limiting values of $\delta y$ and $\delta z$ which can be reached.

IV. CONCLUSIONS AND PERSPECTIVES

Simulations done with MATLAB® (The MathWorks, Inc., Natick, MA, USA), show very clearly the improvement of answers by neural network controller for a magnetic bearing under a computed torque control, compared with classical controllers. Using NN controller under a computed torque control serves to counteract the uncertainties in the system dynamics and disturbances occurring during application. With these results, one can proceed to the next step: the real time application.
REFERENCES


