ADAPTIVE INVERSE CONTROL FOR NONLINEAR SYSTEM

QIAOGE LIU, MENGYIN FU, AND ZHIHONG DENG

Abstract. The research on the application of adaptive inverse control method has received much attention in recent years. But its main problem, when applied to controlling nonlinear systems, is how to adapt the inverse controller. The BPTM (backprop through (plant) model) algorithm and the $\varepsilon$-filtering algorithm can also be used to solve this problem. Based on the $\varepsilon$-filtering algorithm, we proposed another adaptive inverse control system structure according to the sensitivity of nonlinear systems with the input. The adaptive inverse control system based on neural network is presented in this paper. The experimental results showed that the adaptive inverse control system based on either the BPTM algorithm or the $\varepsilon$-filtering algorithm can achieve good performance. And the control structure we proposed can work better.

Key Words. nonlinear system, adaptive inverse control, neural network, BPTM, $\varepsilon$-filtering.

1. Introduction

Strictly speaking, there are no pure linear systems in the real world. Nonlinear systems widely exist in various areas especially in industries. So the research on the control method for nonlinear systems makes great sense. Because the nonlinear systems are complex systems and difficult to control, the majority of conventional control techniques can not work well.

Adaptive inverse control (AIC) is a relatively new approach first introduced by Professor Widrow in 1986. AIC combines the signal processing method with the control theory and is designed to control systems with complex characteristics. The controller has a transfer characteristic inverse to the controlled plant and is in series with the plant. The AIC system is a kind of open-loop control system and the control action is feed-forward. The feedback is only present in the adaptation loop of the controller weights.

The key of AIC is how to seek the inverse controller. If the controlled plant is linear, the inverse model exists and is easy to get. If the controlled plant is nonlinear, the problem is more challenging. Exactly speaking, the nonlinear system has no inverse model and the operation order cannot be exchanged. So the inverse controller is not the inverse model of the plant and cannot be adjusted directly.

Neural network has the ability of nonlinear mappings and can be used in the AIC system. Based on neural network, the inverse controller is commonly updated by the BPTM (backprop through (plant) model) algorithm [2-5]. We specify that the $\varepsilon$-filtering algorithm [1] is also useful for neural network AIC for nonlinear systems, and we also propose another control structure based on the $\varepsilon$-filtering algorithm. AIC for nonlinear systems based on the $\varepsilon$-filtering algorithm are very meaningful.

Received by the editors June 1, 2004 and, in revised form, January 22, 2005.
The AIC method is introduced first in Section 2 and AIC for linear systems is discussed in Section 3. In Section 4, AIC for nonlinear systems which is realized by neural networks is discussed. The BPTM algorithm and two kinds of ε-filtering algorithms are studied respectively. Finally, some simulation results are given and the conclusions are presented.

2. Adaptive Inverse Control (AIC)

The basic structure of AIC is shown in FIGURE 1. The controller $C$ is an inverse controller and can be tuned online according to some rule. The rule is made with a system error $e_S$. $e_S$ is the difference between the command input $r$ and the output $y_S$ of the system. The goal of the control system is to let the output $y_S$ follow the command input $r$. For linear systems, the controller is the inverse model of the controlled plant. For nonlinear systems, the inverse model does not always exist. Though a lot of articles prove nonlinear systems are invertible in certain conditions, the restriction is strong. So we should focus on this problem.

The controller is just a filter that transforms signal. So no matter whether the controlled plant is linear and no matter whether the inverse model exists, the controller is a filter that transforms the command input $r$ to provide a control signal $u$.

3. AIC for linear systems

To adapt controller $C$, the difference between the desired output and the actual output of $C$ should be available.

With AIC for the linear system, this error is easy to get because the linear system is invertible. The regulation can be realized as shown in FIGURE 2. In FIGURE 2, $PM$ is the model of the controlled plant $P$, and controller $C$ is the inverse model of $P$ created by $PM$. Copying $C$ and using it in the main loop can make the response of $P$ follow the command input $r$ exactly.

But the structure shown in Fig.2 is not useful for nonlinear systems and it does not guarantee that $y_S$ tracks $r$ exactly. So in the next section, we’ll study how to get controller $C$ for a nonlinear system.
4. AIC for nonlinear systems

To apply AIC to nonlinear systems, we need a nonlinear filter working as the controller $C$. Neural network has great ability of nonlinear mappings and can make a nonlinear filter. In this paper, we choose multi-layered feed-forward neural networks to realize the controller $C$ and the plant model $PM$.

4.1. BP algorithm for multi-layered feed-forward neural network.

The basic regulation method for multi-layered feed-forward neural networks is the BP algorithm. In this paper, the feed-forward neural network with one hidden layer is selected.

Suppose the input of the neural network is $x(k)$. Let

$$x(k) = [x_1(k), x_2(k), \ldots x_m(k)].$$

Then the output of the hidden layer can be expressed by

$$net_H^j(k) = f_H^j\left(\sum_{i=1}^{m} w_{ij}^H(k)x_i(k) + b_H^j(k)\right),$$

and the output of the network is

$$y(k) = f_O^O\left(\sum_{j=1}^{n} w_{j}^O(k)net_H^j(k) + b_O^O(k)\right),$$

where $n$ is the number of hidden units. $f_H^H$ and $f_O^O$ are the activation functions for the hidden and output layers respectively. In this paper, $f_H^H$ is the hyperbolic tangent activation function and $f_O^O$ is a linear activation function, that is

$$f_H^H(s) = \frac{1 - \exp(-s)}{1 + \exp(-s)}, f_O^O(s) = s.$$

Then we have

$$(f_H^H(s))' = \frac{1}{2}|1 - (f_H^H(s))^2|, (f_O^O(s))' = 1.$$
Suppose the cost function is
\[ J(k) = \frac{1}{2} [y^*(k) - y(k)]^2, \]
where \( y^* \) is the desired output of the network and \( y \) is the actual output of the network. Let \( e(k) = y^*(k) - y(k) \). Then the gradient decent algorithm based updating rule for adjusting the weights in the neural network is given by
\[
\Delta w(k) = \alpha \cdot \frac{\partial J(k)}{\partial w(k)} = \alpha \cdot e(k) \cdot \frac{\partial y(k)}{\partial w(k)}.
\]

At time step \( k \),
\[
\Delta w^O_j(k) = \alpha \cdot e(k) \cdot \frac{\partial y(k)}{\partial w^O_j(k)} = \alpha \cdot e(k) \cdot \text{net}^H_j,
\]
\[
\Delta b^O_j(k) = \alpha \cdot e(k) \cdot \frac{\partial y(k)}{\partial b^O_j(k)} = \alpha \cdot e(k),
\]
\[
\Delta w^H_{ij}(k) = \alpha \cdot e(k) \cdot \frac{\partial y(k)}{\partial w^H_{ij}(k)} = \alpha \cdot e(k) \cdot w^O_j(k) \cdot x_i(k) \cdot (f_H)'(f_H)',
\]
\[
\Delta b^H_j(k) = \alpha \cdot e(k) \cdot \frac{\partial y(k)}{\partial b^H_j(k)} = \alpha \cdot e(k) \cdot w^O_j(k) \cdot (f_H)'(f_H)',
\]
\[
w(k) = w(k-1) + \Delta w(k) + \beta \cdot \Delta w(k-1),
\]
\[
b(k) = b(k-1) + \Delta b(k) + \beta \cdot \Delta b(k-1),
\]
where \( \alpha \) is the learning rate and \( \beta \) is the inertia coefficient.

The above formulas are the basic algorithms for AIC systems based on neural network.

4.2. AIC for nonlinear systems.

For nonlinear system, controller \( C \) is not the inverse model of the controlled plant \( P \) as the linear systems shown in Fig.2. The operation order between \( C \) and \( P \) can not be exchanged and \( C \) can only be ahead of \( P \). So the desired output \( u^* \) of the controller \( C \) isn’t available and \( C \) can’t be adjusted according to the error \( e_C = u^* - u \). The standard BP learning algorithm for adjusting the controller parameters does not work.

In the following part, we will study the algorithms used to adjust the controller in the AIC structure for nonlinear systems.

4.3. BPTM algorithm.

BPTM (backprop through (plant) model) means the error backprop through the plant if the plant function is known or through the plant model if the plant function is unknown. It makes use of the structure of neural networks to update the weights of the controller.

The main idea of BPTM is to regard the plant (model) as additional layers of the neural network \( C \). The additional layers are fixed when adjusting \( C \). So the system error \( e_S \) can be propagated backward to controller \( C \) and then the neural network \( C \) is updated according to BP algorithm. In FIGURE 3, the cost function to adjust \( C \) is \( J = \frac{1}{2}(r - y_S)^2 \). Then
\[ \Delta w_C \propto \frac{\partial J}{\partial w_C} = (r - y_S) \cdot \frac{\partial y_S}{\partial w_C} = (r - y_S) \cdot \frac{\partial y_S}{\partial u} \cdot \frac{\partial y_S}{\partial w_C} = e_S \cdot \frac{\partial y_S}{\partial u} \cdot \frac{\partial u}{\partial w_C}. \]

Let
\[ e_C = e_S \cdot \frac{\partial y_S}{\partial u}. \]

Thus
\[ \Delta w_C \propto e_C \cdot \frac{\partial y_S}{\partial u}. \]

So \( e_C \) can be used to adapt \( C \) with BP algorithm. \( \frac{\partial y_S}{\partial u} \) is the relation of the plant output-input. If the controlled plant \( P \) is unknown, \( \frac{\partial y_S}{\partial u} \) can not be calculated. But if the model \( PM \) is a precise approximation of \( P \), \( y_S \) can be replaced by \( y_{PM} \). \( \frac{\partial y_{PM}}{\partial u} \) can be calculated according to the structure of the neural network \( PM \). Then we can use BP algorithm directly to adjust its parameters.

BPTM has a major drawback. If the NN is realized online, at the beginning of the process, the plant model can’t track the plant well and \( \frac{\partial y_{PM}}{\partial u} \) has great errors which will influence the following process badly.

4.4. \( \varepsilon \)-filtering algorithm.

\( \varepsilon \)-filtering algorithm uses the inverse model \( I_v \) to filter the system error \( e_S \) and get \( e_C \). \( e_C \) is selected as the error updating \( C \) to approach the desired controller \( C^* \), with the goal of minimizing the mean square of the system error \( e_S \).

What is to be emphasized is \( I_v \) is not the inverse model of the nonlinear plant. Produced by the plant model \( PM \), \( I_v \) is only one presentation of the relationship between the output and the input of \( PM \).

4.4.1. \( \varepsilon \)-filtering-A.

The first kind of \( \varepsilon \)-filtering structure, \( \varepsilon \)-filtering-A, is shown in Fig.4. Now we will explain the algorithm of \( \varepsilon \)-filtering with the method of signal flow.

If \( C^* \) is an ideal controller, it can control \( P \) to make
\[ PC^* r = r \]
and then the mean square of \( e_S(k) = r(k) - y_S(k) \) is minimized. Then the goal is to make \( C \) track \( C^* \) as closely as possible to minimize the mean square of \( C^* r - Cr \).

The error to regulate \( C \) is
\[ e_C = I_v r - I_v PC r. \]
It yields

\[ e_C = IvPC^*r - IvPCR. \]

Assuming that PM fits closely P and P can be replaced by PM, then

\[ e_C = IvPMC^*r - IvPMCr. \]

As shown in Fig.4, Iv is trained to be an inverse model of the output signal and input signal of PM, so \( IvPM = 1 \). Then the result \( e_C = C^*r - Cr \) is obtained and is used to regulate C. So minimizing the mean square of \( e_C \), C is as close as possible to \( C^* \), and \( PC^*r = r \) can be realized by \( PCr = r \).

According to this thought, we can use a neural network to act as the controller \( C \). Let the input of the network \( C \) be \( x(k) = [r(k), r(k-1), ... r(k-m+1)] \). Then the regulating rule is

\[ \Delta w(k) \alpha \cdot e_C(k) \cdot \frac{\partial u(k)}{\partial w(k)}, \]

and \( C \) can be adapted by BP algorithm.

4.4.2. \( \varepsilon \)-filtering-B.

The second kind of \( \varepsilon \)-filtering structure is named as \( \varepsilon \)-filtering-B. It is different from \( \varepsilon \)-filtering-A in structure.

As mentioned in 4.4.1, to get \( e_C = IvPMC^*r - IvPMCr = C^*r - Cr \), the key is to make \( IvPM = 1 \). In FIGURE 4, PM is driven by the control signal \( u \) in the course of regulating \( Iv \), while in FIGURE 5, PM is driven from the input signal \( r \).

Though common, such differences make more sense. In FIGURE 4, \( Iv \) is trained by the difference between its output and \( u \), so \( Iv \) is the inverse model of \( PM \) under the control signal. In FIGURE 5, \( Iv \) is trained by the difference between its output and \( r \), so \( Iv \) is the inverse model of \( PM \) under the input signal. It's not possible for the controller to act as the inverse of the nonlinear plant, but the controller is proposed to track the inverse of the plant as closely as possible. Because the action
of a nonlinear plant is influenced not only by its own function but also by the input signal, the input signal is very important for a nonlinear system. So to let \( y_S \) follow \( r \), the controller should act as the inverse of the plant under the signal, but not the signal \( u \).

The following simulation results show that the AIC for nonlinear system based on \( \varepsilon \)-filtering-B works better than \( \varepsilon \)-filtering-A.

5. Simulation results

Select a nonlinear plant to be

\[
y_S(k) = \cos(y_S(k - 1)) + 0.5 \ast u(k - 1),
\]

where \( u(k) \) is the output of the controller at time step \( k \). Choose three neural networks to act as the plant model \( PM \), the inverse model \( Iv \) and the controller \( C \) respectively. The networks for \( PM \) and \( Iv \) can be regulated directly by BP algorithm. The networks for \( C \) can be adapted by BPTM algorithm and \( \varepsilon \)-filtering algorithm with the structure of \( \varepsilon \)-filtering-A and \( \varepsilon \)-filtering-B.

Two kinds of input signals are supposed to drive the simulation system. They are a ladder signal and a random signal. The simulation results are shown.

We can see that the system with BPTM algorithm responds the ladder signal with larger delay. The system with \( \varepsilon \)-filtering-A has larger initial error responding to any kind of input signal. The responses of the system with \( \varepsilon \)-filtering-B have less delays and can track quickly with less oscillation when the input changes suddenly. So the system with \( \varepsilon \)-filtering-B is better.

6. Conclusions

Adaptive inverse control (AIC) is a kind of feedforward control. It combines the signal processing method with the control theory. The key of AIC is how to make and regulate the inverse controller. For linear systems, the inverse controller is the inverse model of the controlled plant, so it is easy for the system to work. For
nonlinear systems, the inverse model of the controlled plant may or may not exist, so the inverse controller is not easy to get. AIC based on BPTM algorithm and $\varepsilon$-filtering algorithm can work for controlling nonlinear systems. This paper studied these two algorithms and proposed another structure named as $\varepsilon$-filtering-B. The key is to set the inverse of the nonlinear plant under certain input, which is more
important for nonlinear system. Simulation results showed that the AIC system based on the $\varepsilon$-filtering worked better.

To apply the adaptive inverse control (AIC) to controlling nonlinear system, the nature of AIC should be emphasized. More useful ideas and better results can be obtained if we refer more to the method in signal processing area.

Figure 7. AIC for Nonlinear System Based on $\varepsilon$-filtering.
Figure 8. AIC for Nonlinear System Based on $\varepsilon$-filtering-B

References


Department of Automatic Control, School of Information Science and Technology, Beijing Institute of Technology, Beijing, 100081, China

E-mail: lqge@bit.edu.cn