Proceedings of the 36th
Conference on Decision & Control
San Diego, California USA · December 1997

Nonlinear Adaptive Inverse Control

Bernard Widrow and Gregory L. Plett

Department of Electrical Engineering,
Stanford University, Stanford, CA 94305-9510
email: glp@simoon.stanford.edu

I. Abstract

Adaptive control is seen as a two part problem, (a) control of plant dynamics, and (b) control of plant disturbance. Conventionally, one uses feedback control to treat both problems simultaneously. Tradeoffs and compromises are necessary to achieve good solutions, however.

The method proposed here, based on inverse control, treats the two problems separately without compromise. The method applies to SISO and MIMO linear plants, and to nonlinear plants.

An unknown linear plant will track an input command signal if the plant is driven by a controller whose transfer function approximates the inverse of the plant transfer function. An adaptive inverse identification process can be used to obtain a stable controller, even if the plant is nonminimum phase. A model-reference version of this idea allows system dynamics to closely approximate desired reference-model dynamics. No direct feedback is used, except that the plant output is monitored and utilized by an adaptive algorithm to adjust the parameters of the controller. Although nonlinear plants do not have transfer functions, the same idea works well for nonlinear plants.

Control of internal plant disturbance is accomplished with an adaptive disturbance canceler. The canceler does not affect plant dynamics, but feeds back plant disturbance in a way that minimizes plant output disturbance power. This approach is optimal for linear plants, and works surprisingly well with nonlinear plants.

II. Introduction

Many problems in adaptive control can be divided into two parts: (a) control of plant dynamics, and (b) control of plant disturbance. Very often, a single system is utilized to achieve both of these control objectives. The approach of this paper treats each problem separately. Control of plant dynamics can be achieved by preceding the plant with an adaptive controller whose transfer function is the inverse of that of the plant. Control of plant disturbance can be achieved by an adaptive feedback process that minimizes plant output disturbance without altering plant dynamics [1]. The adaptive controller is implemented using adaptive filters.

III. Adaptive Filters

An adaptive digital filter, shown in Fig. 1, has an input, an output, and another special input called the "desired response." The desired response input is sometimes called the "training signal."

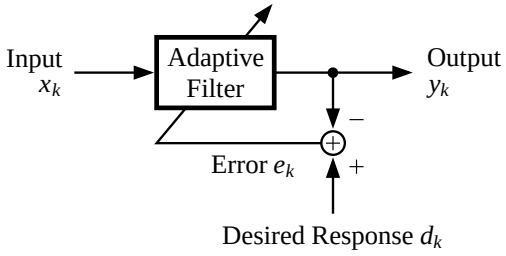


Fig. 1. *Symbolic representation of an adaptive transversal filter adapted by the LMS algorithm.*

The adaptive filter contains adjustable parameters that control its impulse response. These parameters could, for example, be variable weights connected to the taps of a tapped delay line. The filter would thus be FIR, finite impulse response.

The adaptive filter also incorporates an "adaptive algorithm" whose purpose is to automatically adjust the parameters to minimize some function of the error (usually mean square error). The error is defined as the difference between the desired response and the actual filter response. Many such algorithms exist, a number of which are described in the textbooks by Widrow and Stearns [2] and by Haykin [3].

IV. Inverse Plant Modeling

The plant's controller will be an inverse of the plant. Inverse plant modeling of a linear SISO plant is illustrated in Fig. 3. The plant input is its control signal. The plant output, shown in the figure, is the input to an adaptive filter. The desired response for the adaptive filter is the plant input (sometimes delayed by a modeling delay, Δ). Minimizing mean square error causes the adaptive filter $\widehat{P}^{-1}$ to be the best least squares inverse to the plant P for the given input spectrum. The adaptive algorithm attempts to make the cascade of plant and adaptive inverse behave like a unit gain. This process is often called deconvolution. With the delay Δ incorporated as shown, the inverse will be a delayed inverse.

0–7803–3970–8/97 $10.00 © 1997 IEEE

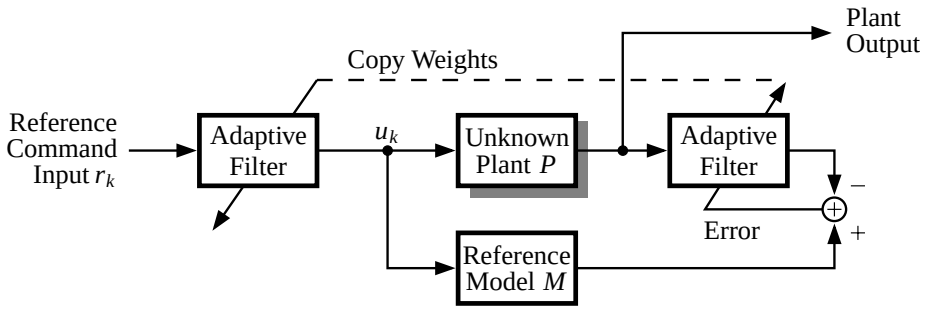


Fig. 2. *Adaptive inverse model control system.*

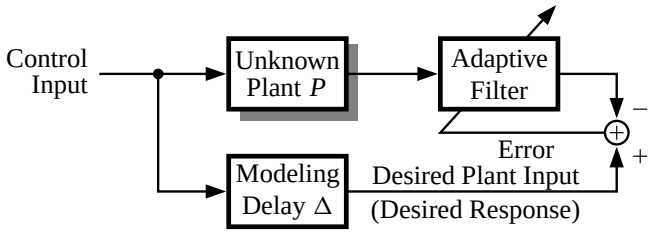


Fig. 3. *Delayed inverse modeling of an unknown plant.*

For sake of argument, the plant can be assumed to have poles and zeros. An inverse, if it also had poles and zeros, would need to have zeros where the plant had poles and poles where the plant had zeros. Making an inverse would be no problem except for the case of a nonminimum phase plant. It would seem that such an inverse would need to have unstable poles, and this would be true if the inverse were causal. If the inverse could be noncausal as well as causal, however, then a two-sided stable inverse would exist for all linear time-invariant plants in accord with the theory of two-sided z-transforms. For useful realization, the two-sided inverse response would need to be delayed by Δ. A causal FIR filter can approximate the delayed version of the two-sided plant inverse. The time span of the adaptive filter (the number of weights multiplied by the sampling period) should be made adequately long, and the delay Δ needs to be chosen appropriately. The choice is generally not critical.

The inverse filter is used as a controller in the present scheme, so that Δ becomes the response delay of the controlled plant. Making Δ small is generally desirable, but the quality of control depends on the accuracy of the inversion process which sometimes requires Δ to be of the order of half the length of the adaptive filter.

A model-reference inversion process is incorporated in the feedforward control system of Fig. 2. A reference model is used in place of the delay of Fig. 3. Minimizing mean square error with the system of Fig. 2 causes the cascade of the plant and its "model-reference inverse" to closely approximate the response of the reference-model M. Much is known about the design of model reference systems [4]. The model is chosen to give a desirable response for the overall system.

Thus far, the plant has been treated as disturbance free. But, if there is disturbance, the scheme of Fig. 4 can be used. A direct plant modeling process, not shown, yields $\widehat{P}$, a close-fitting FIR model of the plant. The difference between the plant output and the output of $\widehat{P}$ is essentially the plant disturbance.

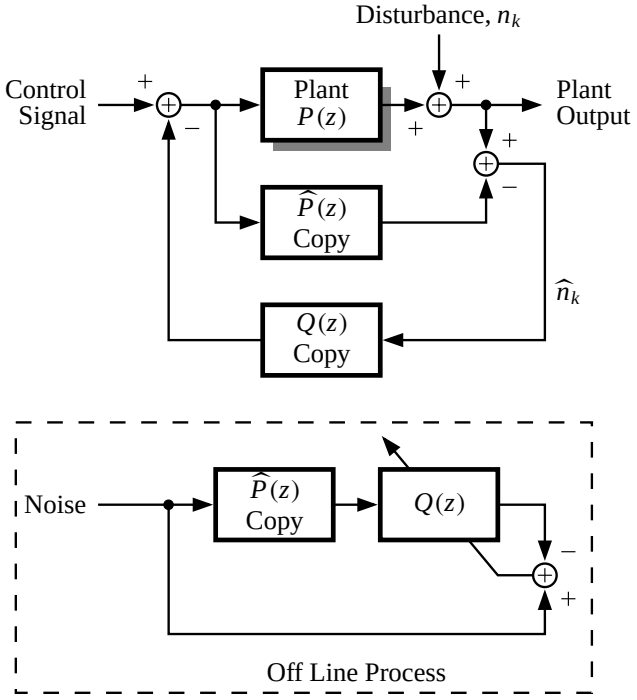


Fig. 4. *Optimal adaptive plant disturbance canceler.*

Now, using a digital copy of $\widehat{P}$ in place of P, an off line process, shown in Fig. 4, calculates the best least-squares plant inverse Q. The off line process can run much faster than real time, so that as $\widehat{P}$ is calculated, the inverse Q is immediately obtained. The disturbance is filtered by a digital copy of Q and subtracted from the plant input. For linear systems, the scheme of Fig. 4 has been shown to be optimal in the least-squares sense [1].

To illustrate the effectiveness of adaptive inverse control, a non-minimum phase plant has been simulated, and its impulse response is shown in Fig. 5(a). The output of this plant and the output of its reference model are plotted in Fig. 5(b), showing dynamic tracking when the command input signal is a random first-order Markov process. Tracking

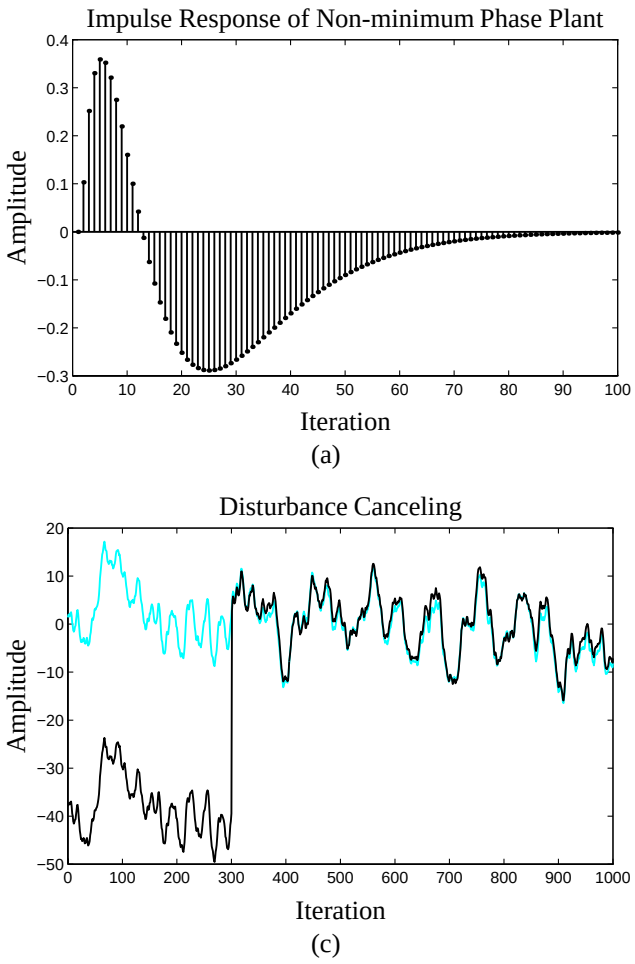


Fig. 5. *(a) Impulse response of the non-minimum phase plant used in simulation; (b) Dynamic tracking of desired output by actual plant output when the plant was not disturbed. The grey line is the desired output and the black line is the actual plant output. The control scheme was model-reference based, with a first-order one-pole model; (c) Cancelation of plant disturbance. The grey line is the desired output and the black line is the actual output. The disturbance canceler was turned on at iteration 300.*

is quite good. With disturbance added to the plant output, Fig. 5(c) shows the effect of disturbance cancelation. Both the desired and actual plant outputs are plotted in the figure, and they become close when the canceler is turned on at 300 sampling periods.

V. Nonlinear Adaptive Inverse Control with Neural Networks

Nonlinear inverse controllers can be used to control nonlinear plants. Although the theory is in its infancy, experiments can be done to demonstrate this. A nonlinear adaptive filter is shown in Fig. 6. It is composed of a neural network whose input is a tapped delay line connected to the exogenous input signal. In addition, the input to the network might include a tapped delay line connected to its own output signal. This type of nonlinear filter is called a *Nonlinear AutoRegressive with eXogenous Input (NARX)* filter, and has recently been shown to be a universal dynamical system [5]. Algorithms such as real-time-recurrent-learning (RTRL) [6] and backpropagation-through-time (BPTT) [7] may be used to adapt the weights of the neural network to minimize the mean squared error. If the feedback connections are omitted, the familiar backpropagation algorithm may be used [8], [9]. In the nonlinear adaptive inverse control scheme of Fig. 7, such filters are used as the plant emulator and controller.

Nonlinear systems do not commute. Therefore, the simple and intuitive block-diagram method of Figs. 2 and 3 for adapting a controller to be the inverse of the plant will not work if the plant is nonlinear. Instead, a lower-level mathematical approach is taken. We use an extension of the RTRL learning algorithm to train the controller. This method can be briefly summarized using the notation of ordered derivatives, invented by Werbos [8]. The goal is to adapt the weights of the controller to minimize the mean squared error at the output of the system. We use the fact that the controller computes a function of the form

$$u_k = g\big(u_{k-1}, u_{k-2}, \ldots, u_{k-m}, r_k, r_{k-1}, \ldots, r_{k-q}, W\big),$$

where W are the weights of the controller's neural network. We also use the fact that the plant model computes a function of the form

$$y_k = f\big(y_{k-1}, y_{k-2}, \ldots, y_{k-n}, u_k, u_{k-1}, \ldots, u_{k-p}\big).$$

The weights of the controller are adapted using steepest descent. The change in the weights at each time step is in the negative direction to the gradient of the system error with respect to the weights of the controller. To find the gradient, we use the chain-rule expansion for ordered derivatives

$$\frac{\partial^+ \|e_k\|^2}{\partial W} = -2e_k \frac{\partial^+ y_k}{\partial W}$$

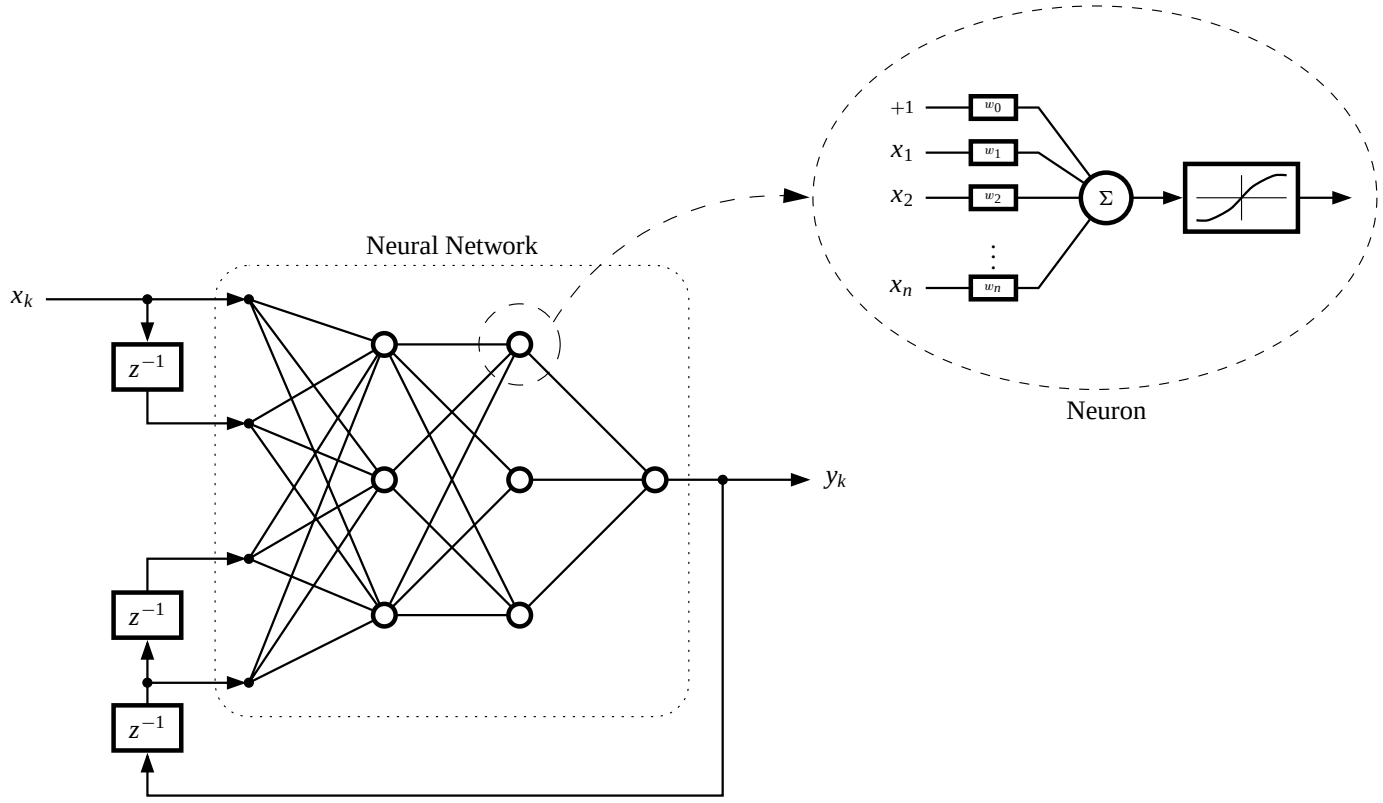


Fig. 6. *An adaptive nonlinear filter composed of a tapped delay line and a three-layer neural network.*

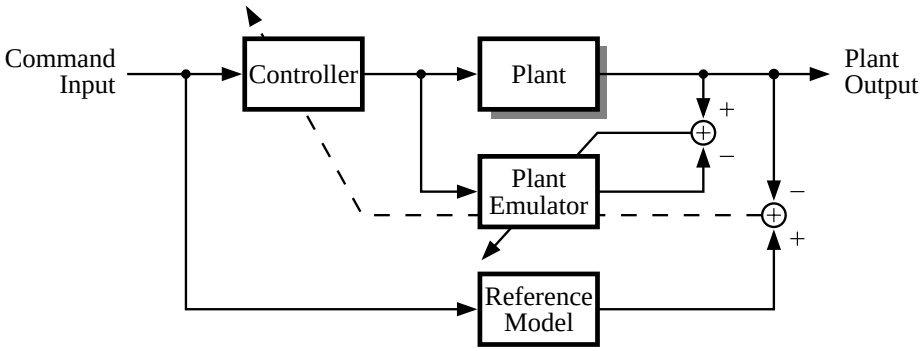


Fig. 7. *A method for adapting a nonlinear controller.*

$$\frac{\partial^+ u_k}{\partial W} = \frac{\partial u_k}{\partial W} + \sum_{j=1}^{m} \left(\frac{\partial u_k}{\partial u_{k-j}} \right) \left(\frac{\partial^+ u_{k-j}}{\partial W} \right) \quad (1)$$

$$\frac{\partial^+ y_k}{\partial W} = \sum_{j=0}^{p} \left(\frac{\partial y_k}{\partial u_{k-j}} \right) \left(\frac{\partial^+ u_{k-j}}{\partial W} \right) + \sum_{j=1}^{n} \left(\frac{\partial y_k}{\partial y_{k-j}} \right) \left(\frac{\partial^+ y_{k-j}}{\partial W} \right). \quad (2)$$

Each of the terms in Eqs. (1) and (2) is either a Jacobian matrix, which may be calculated using the *dual-subroutine* [10] of the backpropagation algorithm, or is a previously calculated value of $\partial^+ u_k / \partial W$ or $\partial^+ y_k / \partial W$.

To be more specific, the first term in Eq. (1) is the partial derivative of the controller's output with respect to its weights. This term is the one normally calculated by the backpropagation algorithm to update the weights of a static neural network.

The second part of Eq. (1) is a summation. The first term of the summation is the partial derivative of the controller's current output with respect to a previous output. However, since the controller is externally recurrent, this previous output is also a current *input*. Therefore the first term of the summation is really just a partial derivative of the output of the controller with respect to one of its inputs. By definition, this is a submatrix of the Jacobian matrix for the network, and may be computed using the dual-subroutine of the backpropagation algorithm.

The second term of the summation in Eq. (1) is the ordered

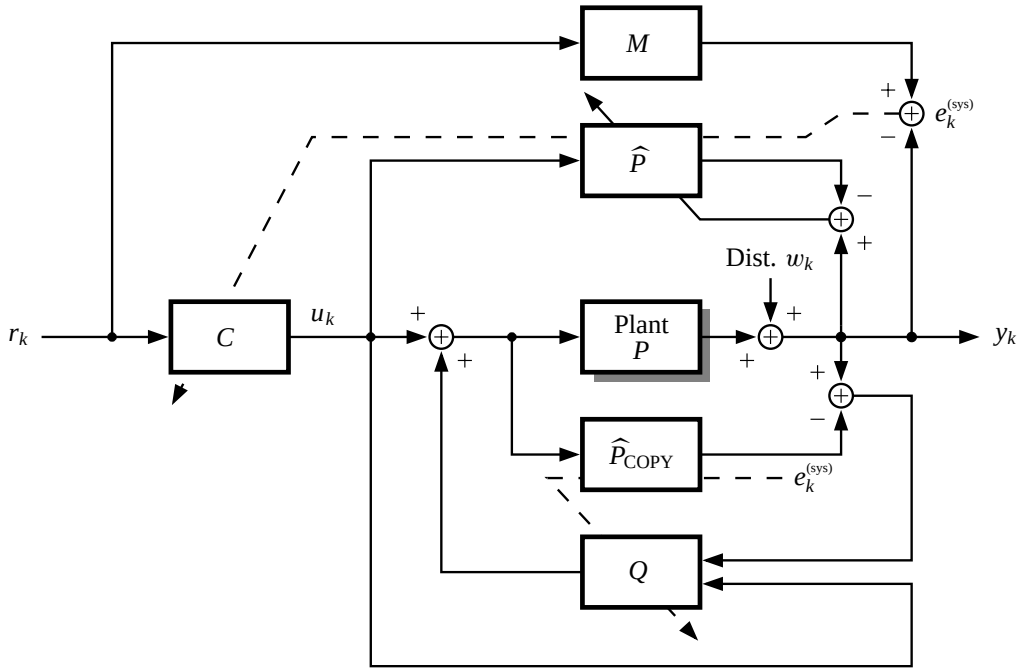
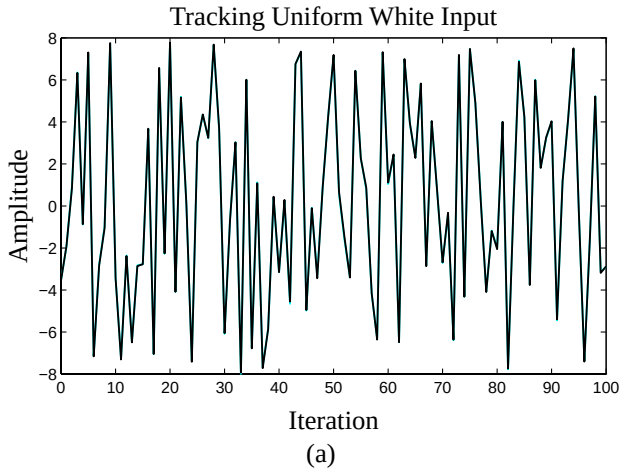
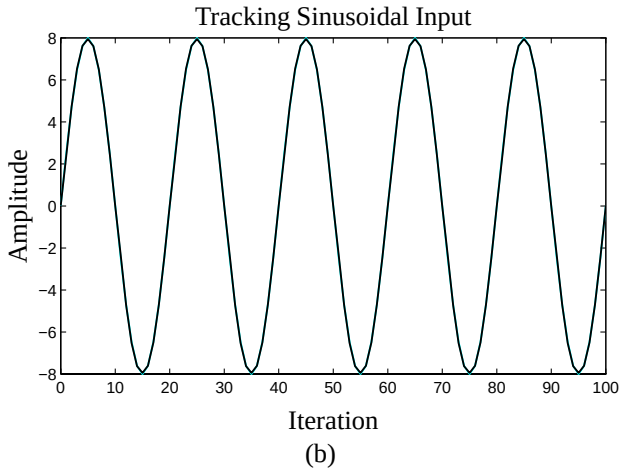


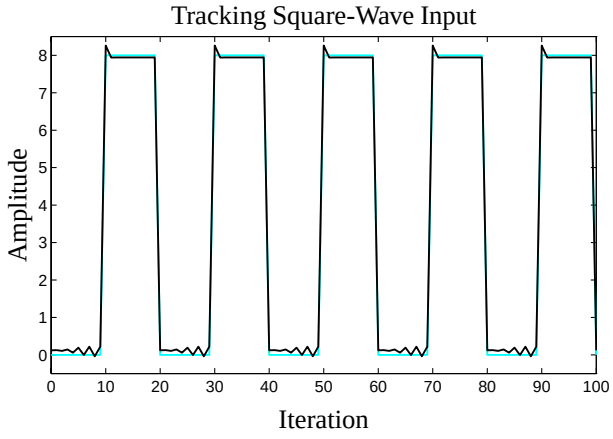
Fig. 8. *A fully integrated nonlinear adaptive inverse control scheme.*



(a)



(b)



(c)

Fig. 9. *Feedforward control of a nonlinear system. The controller was trained with uniformly distributed (white) random input. Plot (a) depicts the desired plant output (gray line) and the true plant output (solid line), at the end of training, when the training signal was used to drive the controller. The gray line is completely covered by the black line, indicating near-perfect control. With the weights fixed at their trained values, the next two plots show the generalization ability of the controller. Plots (b) and (c) show the plant tracking sinusoidal and square waves, having been previously trained with the random input.*

partial derivative of a previous output with respect to the weights of the controller. This term has already been computed in a previous evaluation of Eq. (1), and need not be re-computed.

A similar analysis may be performed to determine all of the terms required to evaluate Eq. (2). After calculating these terms, the weights of the controller may be adapted using the weight-update equation

$$\Delta W_k = 2\mu e_k \frac{\partial^+ y_k}{\partial W}.$$

Continual adaptation will minimize the mean squared error at the system output.

Disturbance canceling for a nonlinear system is performed by filtering an estimate of the disturbance with the nonlinear filter Q and adding the filter's output to the control signal. An additional input to Q is the control signal to the plant, u_k, to allow the disturbance canceler knowledge of the plant state. The same algorithm which was used to adapt the controller can be used to adapt the disturbance canceling filter. The entire control system is shown in Fig. 8.

An interesting discrete-time nonlinear plant has been studied by Narendra and Parthasarathy [11]

$$y_k = \frac{y_{k-1}}{1 + y_{k-1}^2} + u_{k-1}^3.$$

The methods just described for adapting a controller and disturbance canceler were simulated for this plant, and the results are presented here.

With the reference model being a simple unit delay, and the command input being an i.i.d. uniform process, the system adapted and learned to track the model output. The result is shown in Fig. 9(a). After training with the random input, the adaptive process was halted. With no further training, the system was tested with inputs of different character in order to demonstrate the generalization ability of the controller. The first test was a sine-wave command input. Tracking was surprisingly good, as shown in Fig. 9(b). Again, without further training, the system was tested with a square-wave command input, and the results, shown in Fig. 9(c), are excellent.

A disturbance canceler was also trained for this plant, where the disturbance was a first-order Markov signal added to the plant output. Fig. 10 shows the results of disturbance cancelation. The power of the system error is plotted versus time. The disturbance canceler was turned on at iteration 500. Dramatic improvement may be seen.

A great deal of work will be needed to gain greater understanding of this kind of behavior, but the prospects for useful and unusual performance and for the development of new control theory seem very promising.

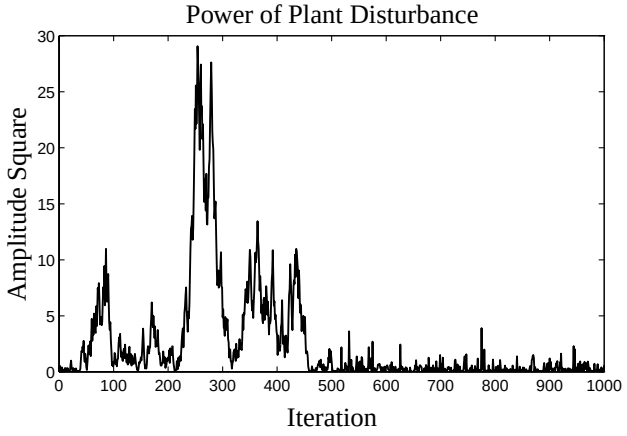


Fig. 10. *Cancelation of plant disturbance for a nonlinear plant. The disturbance canceler was turned on at iteration 500.*

References

[1] B. Widrow and E. Walach. *Adaptive Inverse Control*. Prentice Hall P T R, Upper Saddle River, NJ, 1996.

[2] B. Widrow and S. D. Stearns. *Adaptive Signal Processing*. Prentice-Hall, Englewood Cliffs, NJ, 1985.

[3] S. Haykin. *Adaptive Filter Theory*. Prentice Hall, Upper Saddle River, NJ, third edition, 1996.

[4] I. D. Landau. *Adaptive Control. The Model Reference Approach*, volume VIII of *Control and Systems Theory Series*. Marcel Dekker, New York, 1979.

[5] H. T. Siegelmann, B. B. Horne, and C. L. Giles. Computational capabilities of recurrent NARX neural networks. *IEEE Transactions on Systems, Man and Cybernetics—Part B: Cybernetics*, 27(2):208–215, April 1997.

[6] R. J. Williams and D. Zipser. Experimental analysis of the real-time recurrent learning algorithm. *Connection Science*, 1(1):87–111, 1989.

[7] P. Werbos. Backpropagation through time: What it does and how to do it. *Proceedings of the IEEE*, 78(10):1545–1680, October 1990.

[8] P. Werbos. *Beyond Regression: New Tools for Prediction and Analysis in the Behavioral Sciences*. PhD thesis, Harvard University, Cambridge, MA, August 1974.

[9] D. E. Rumelhart, G. E. Hinton, and R. J. Williams. Learning internal representations by error propagation. In D. E. Rumelhart and J. L. McClelland, editors, *Parallel Distributed Processing*, volume 1, chapter 8. The MIT Press, Cambridge, MA, 1986.

[10] P. Werbos. Neurocontrol and supervised learning: An overview and evaluation. In D. White and D. Sofge, editors, *Handbook of Intelligent Control: Neural, Fuzzy, and Adaptive Approaches*, chapter 3. Van Nostrand Reinhold, New York, 1992.

[11] K. S. Narendra and K. Parthasarathy. Identification and control of dynamical systems using neural networks. *IEEE Transactions on Neural Networks*, 1(1):4–27, March 1990.