TECHNICAL RESEARCH REPORT

Reconstruction of Nonlinear Systems Using Delay Lines and Feedforward Networks

by D.L. Elliott

T.R. 95-17

Sponsored by
the National Science Foundation
Engineering Research Center Program,
the University of Maryland,
Harvard University,
and Industry

ISR Technical Research Report T.R. 95-17

RECONSTRUCTION OF NONLINEAR SYSTEMS USING DELAY LINES AND FEEDFORWARD NETWORKS [1]

David L. Elliott
NeuroDyne, Inc. and
Institute for Systems Research,
University of Maryland
College Park, MD 20742
delliott@src.umd.edu

Abstract

Nonlinear system theory ideas have led to a method for approximating the dynamics of a nonlinear system in a bounded region of its state space, by training a feedforward neural network which is then reconfigured in recursive mode to provide a stand-alone simulator of the system. The input layer of the neural network contains time-delayed samples of one or more system outputs and control inputs. Autonomous systems can be simulated in this way by providing impulse inputs.

Introduction

In the past, methods for identifying nonlinear systems have typically used static polynomial (or sinusoidal) nonlinearities with linear dynamical blocks (Hammerstein or Wiener operators); Volterra functional series (for short times and small inputs); and other polynomial-based methods such as Larimore's CVA [2]. Feedforward neural networks using past samples of system outputs and inputs seem to be a reasonable method for predicting future outputs of time series (a common task in the neural network literature) and for system identification. Work in the control community [3] has used neural network simulators in control applications with some success. The hope is that future inexpensive neural network hardware will provide fast parallel computation.

It is of course necessary to store past information about system inputs and outputs in the network to successfully train the weights and reconstruct dynamics: the network must contain delay elements. In the architecture here presented, all these elements are incorporated in delay-lines at the front end of the network. Others have used delays between the nodes of the network. .

Linear Case

Linear adaptive ARMA filters trained by the Widrow-Hoff LMS algorithm can be used to identify linear systems [1] by finding an empirical transfer function. A sufficiently-exciting input is presented to a linear plant of interest; the past values of the input and output are sampled at clock interval τ and their values u_k and y_k are used as inputs to two delay-lines of length L. The

[1] Portions of this research were supported by the National Science Foundation under Grant ECS 9216530. To appear, Proc. 1995 American Control Conference.

delayline values are related by the plant transfer function: $\hat{y} = H(z)\hat{u}$,
$H(z) = (B_1 + \cdots + B_n z^L)/(1 + A_1 z + \cdots + A_L z^L)$.
The two delay-lines are used to construct an adaptive "transverse filter" as follows. The delay-line taps are provided with adjustable weights $A_k,\ B_k$ and the outputs summed to give
$Y_t = \Sigma_{k=1}^{L}(A_k y_{t-k\tau} + B_k u_{t-k\tau})$.
The most recent sample $y(t)$ of the plant output is compared with the value Y_t predicted by the filter, and the squared error minimized by adjusting the A_k and B_k. online by the LMS rules for $1 \leq k \leq L$:

$$\begin{aligned} \Delta A_k &= \alpha(Y_t - y_t)y_{t-k\tau}, \\ \Delta B_k &= \alpha(Y_t - y_t)u_{t-k\tau} \end{aligned} \qquad (1)$$

or by similar rules (normalized LMS) in which the right-hand sides of Eq. (1) are normalized by the variances of y and u respectively.

Note that these difference equations (1) are linear in the weights. Finding the best L and α to achieve rapid convergence is difficult, and those problems are of course no easier in the nonlinear generalization to be discussed below.

Neural Networks for Nonlinear Systems
The advent of neural network technology has raised the prospect of highly parallel computational approaches to identification, based on theorems showing that several of the function systems used in neural networks can be used to uniformly approximate multivariable functions that are sufficiently smooth. The transition maps for nonlinear systems appear to be an appropriate application.

The task may be attacked through the use of **recurrent** neural networks [3] – in the above case of the linear adaptive filter this means $\Delta A_k = \alpha(Y_t - y_t)Y_{t-k\tau}$, which is quadratic in the weights, so that the dynamics in weight-space may become chaotic. Recurrent networks, like adaptive control systems, rely on slow variations of the trained parameters (weights) to ensure stability. However, the training can be performed by a feedforward network [4, 6, 10]; in this case recursion is used in the **trained** network but **not during training**. That is the approach we adopt here.

Our work is based on a state-space approach, related to control theory notions of state, observability and controllability. The approach of [4] involves input-output relations.

Learning the Dynamics

Even some autonomous systems $\dot{x} = f(x)$ can be treated as input-driven; we initialize at an equilibrium point (we could assume $f(0) = 0$ for instance) and provide a control $bu(t)$ in such a way that for the controlled system $\dot{x} = f(x) + bu(t)$ the region of interest in the state space can be reached from 0. The input $u(t)$ consists of one or more short pulses whose amplitudes vary randomly. As an example consider a van der Pol oscillator equipped with an input which makes it completely reachable with one pulse

$$\begin{aligned} \dot{\xi}_1 &= \xi_2 \\ \dot{\xi}_2 &= -\xi_1 - (a\xi_2(\xi_1^2 - 1) - 1) + u(t) \\ y &= \xi_1 \end{aligned} \qquad (2)$$

The system is simulated digitally by any method which well-approximates the system dynamics. We know the output only through randomly chosen finite sequences of clocked samples $Y = y(t) : t = k\tau,\ k = 0, -1, -2, \cdots, -T$. The choice of τ and the training length T is problem-dependent.

First consider the *autonomous* (no input) case. Suppose initial states $x(0)$ for identification experiments can be randomly chosen, uniformly distributed in a box B of side β. The random output sequences Y are now looked at with a sliding window of length $N \geq n$. The N-vectors of sample values $Y(t) = [y(t - N\tau), \cdots, y(t - \tau)]$ (defined for y a system output) will determine the future of y, and are called **lag coordinates** for the system, if the map $x(t) \to Y_N(t)$ is a regular embedding. That is a stronger requirement than the local rank conditions used in local-observability theory. For observable linear systems $N = n$ suffices for almost all τ, from elementary considerations. From the work of Floris Takens [7] and independently Dirk Aeyels [8] it is known that for smooth "generic" choices of nonlinear $\{f, h\}$ and almost all τ, $N = 2n + 1$ is enough to guarantee regular embedding. For particular systems

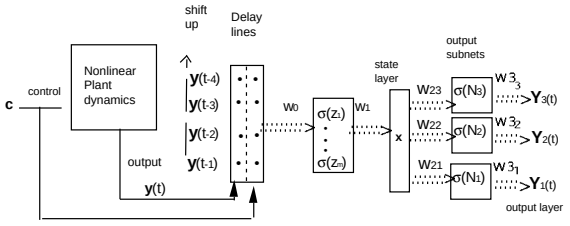

Figure 1: Training the network

the value of N must be determined by experimentation or careful analysis. (Since in this study the inputs are used only to initialize the plant, the same considerations apply to examples like Eq. (2) provided that the input is known.)

A generic smooth nonlinear system is thus observable using a sliding window of $N = 2n + 1$ samples. That is, the state space can be mapped one-to-one into (not onto) Euclidean space of dimension N by the lag coordinates $Z(t) = [y(t), y(t - \tau), ..., y(t - (N - 1)\tau)]$. The Nonlinear Science research community uses this technique in studying the dimension of strange attractors. Recently Sauer [5] used low-pass filtered delay coordinates, and local low-dimensional linear models (as in the work of Farmer and Sidorowich) for prediction. In the Systems community, Moraal and Grizzle [9] have constructed nonlinear state *observers* using n lag coordinates under the hypothesis that the map $x(t) \to Z(t)$ has full rank.

Neural Network Design

In this study $\sigma(s) = \tanh(s)$, or $\sigma(s) = \frac{s}{1+|s|}$. These both have good differential equations to permit fast "backpropagation": $\sigma' = 1 - \sigma^2$ for the first one, $\sigma' = (1 - |\sigma|)^2$ for the second, as I have shown elsewhere.

The first layer of our neural net implements the sliding window by two delay-lines or "FIFOs". The first FIFO at time $0 \le t \le T$ holds a sliding array of samples of $y()$ denoted by $\mathcal{Y}_\sqcup = [y(t - (N + 1)\tau), \cdots, y(t - \tau)]$. This FIFO is left-shifted at each clock time $k\tau$. The second FIFO holds a similar array of input samples, $\mathcal{U}_\sqcup = [u(t - N\tau), \cdots, u(t)]$. The layers of the net are built of linear combinations and sigmoidal functions. $\mathcal{Y}_t$ and $\mathcal{U}_t$ are used as the first layer of our network. They are stacked up in a single vector $\mathbf{r} = [\mathcal{Y}, \mathcal{U}]$. Now we want to project $\mathbf{r}$ down to a vector $\mathbf{x}$ whose dimension P is that of our plant ($P = 2$ for van der Pol's oscillator). This projection may be done either directly by a matrix, or via a sigmoidal hidden layer z of size M (to be chosen in the application) as shown in Figure 1. In that case, we have (using as usual a bias weight a_{i0})
$z_i = \sigma(\sum_{h=1}^{L}(a_{ih}\mathcal{Y}_h + b_{ih}\mathcal{U}_h) + a_{i0})$
for $1 \le i \le M$ (with, as usual, bias weights a_{i0}). We used the notation a_i and b_i in analogy to the linear adaptive filter notation above; they are stacked up in Figure 1, $w0 = [a, b]$. For $j = 1, \cdots, P$,
$x_j = \sum_0^M w2_{ji}z_i$
where $z_0 = 1$ for the bias term; likewise $x_0 = 1$.

In some applications it is possible and convenient to train several network outputs $Y_1, \cdots, Y_R$ corresponding to plant observables (in Figures 1 and 2, $R = 4$). The outputs are now computed by R independent subnetworks, or segments, each with a sigmoidal layer and a linear layer; for $1 \le k \le R$
$N_{kq} = \sigma(\sum_{p=0}^{P} w2_{kqp}x_p)$ and
$Y_k = \sum_{q=1}^{Q} w3_{kq}N_{kq}$.
One of the outputs, denoted $Y_1(t)$ in Figure 1, predicts $y(t)$. The others predict other plant observables (which, for a plant observable with a single output, need not be stored in delay-lines). To allow successful prediction one must train the network by providing sufficiently many inputs u. In Figure 1, $\mathbf{c}$ is used for the class of admissible controls u, which are treated as random functions on [0,T]. Training consists of changing the weights by backpropagation (the multi-layer version of LMS).

The region in the plant's state space explored by these inputs is the interior of the reachable set, allowing for bounds on the inputs. We want to minimize the expected sum of the errors over the ensemble of controls, so the goal (if not always the accomplishment) of backpropagation is to find

$$w^* = arg\min_w \mathrm{E}_{\mathbf{c}}\Sigma_{k=1}^{N}(Y(t - k\tau) - y(t - k\tau))^2$$

This will be our operational definition of identifi-

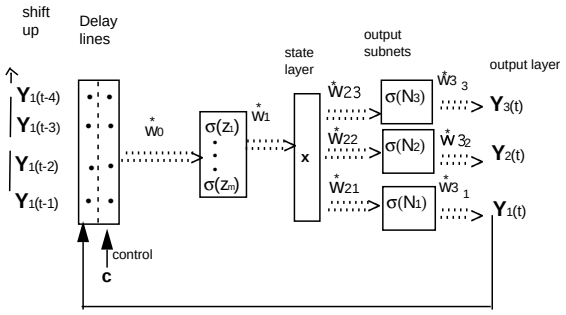

Figure 2: Trained, recurrent network

cation, and it must be understood that ordinarily it cannot be accomplished. The best we can do is, as usual with nonlinear least-squares problems, to find a good local minimum, and use that for w^*. Now, as in Figure 2, we use w^* in a different configuration of the network, in which y from the plant is replaced by the output Y_1 of the network, giving a recursive network which as will be seen in some example figures, can mimic the original plant. The bottleneck layer $\mathbf{x}$ of dimension n may be treated as a state vector in some problems and assigned initial values. The parameters (number of nodes, learning rates) are a matter of art.

A comparison of the impulse response of the controlled van der Pol oscillator of Eq. (2) with our neural network reconstruction is shown in Figure 3; these are lag-coordinate plots of $(y(t), y(t - 0.15))$ and $(Y(t), Y(t-0.15))$. and Figure 4 shows time plots. Here I used the single plant output of Eq. (2), $N = 4$, linear projection to the bottleneck layer
$x_i = \sigma(\sum_{h=1}^{4}(a_{ih}\mathcal{Y}_h + b_{ih}\mathcal{U}_h) + a_{i0})$,
with $P = 2$, one output and $Q = 36$ to obtain the accuracy shown for large transients and good limit-cycle fit.

Conclusion

It has been the intent of this study to show that a simple, canonical architecture with delay-lines and layered feedforward networks is adequate to perform the identification of some interesting and useful nonlinear systems. More complex schemes may well have advantages, but may also require more than off-the-shelf circuits to implement. Single-Instruction Multiple Data parallel circuits can implement the methods presented here.

References

[1] Peter M. Clarkson " Optimal and Adaptive Signal Processing" Boca Raton, Florida, CRC Press, 1993.

[2] W. E. Larimore "Identification and Filtering of Nonlinear Systems Using Canonical Variate Analysis" in **Nonlinear Modeling and Forecasting** M. Casdagli and S. Eubank, editors, New York: Addison-Wesley, 1992.

[3] P. Werbos, T. McAvoy, H. T. Su "Neural Networks, System Identification, and Control in the Chemical Process Industries" in *Handbook of Intelligent Control* D. Sofge and D. White, eds. New York: Van Nostrand Reinhold, 1992.

[4] S.Chen, S. A. Billings "Neural networks for nonlinear dynamic system modeling and identification" *Int.J.Control* **56**, No.2, 319-346, 1992.

[5] Tim Sauer "Time Series Prediction Using Delay Coordinate Embedding," in **Predicting the Future and Understanding the Past: A Comparison of Approaches**, A. Weigend, N. Gershenfeld, editors. Reading, MA : Addison-Wesley Pub. Co., 1994.

[6] A.U. Levin *Neural Networks in Dynamical Systems: a System Theoretic Approach* Ph.D. Thesis, Yale University, November 1992.

[7] F. Takens "Detecting strange attractors in turbulence" in *Dynamic Systems: Warwick, 1980* D. Rand, L.-S. Young, editors. Berlin: Springer Lec. Notes Math. **898**, 366–381 (1981).

[8] D. Aeyels "Generic observability of differentiable systems" *SIAM J. Contr. Opt.* **19** 595-603, 1981.

[9] P. E. Moraal, J. W. Grizzle "Observer design for nonlinear systems with discrete-time measurements", to appear *IEEE Trans. Auto. Con.* March 1995.

[10] A. U. Levin, K. S. Narendra "Identification of nonlinear dynamical systems using neural networks," to appear in **Neural Networks for Control**, D. L. Elliott, editor, Ablex Publishing Corporation, Norwood, NJ, 1995.

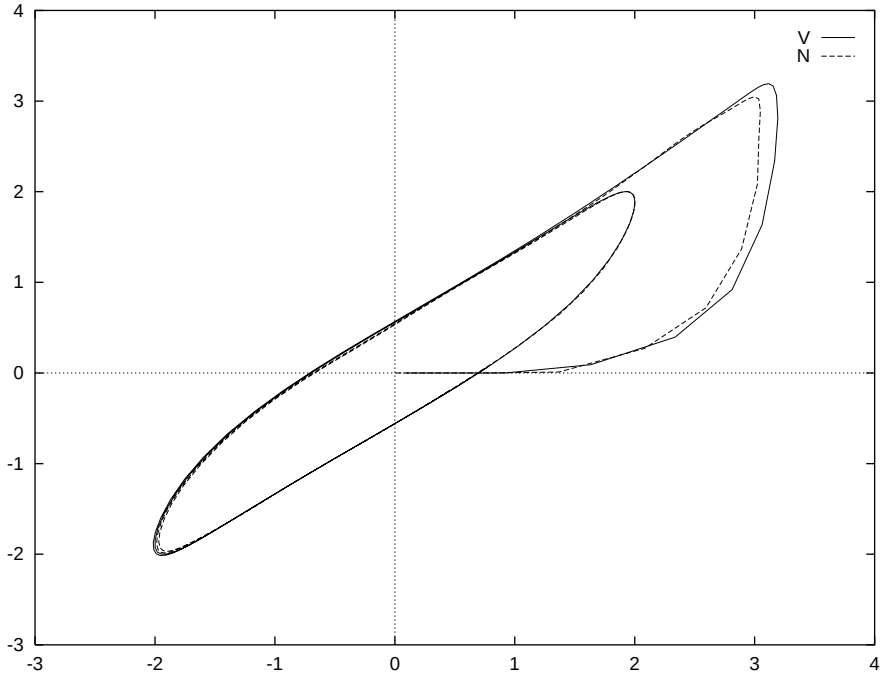

Figure 3: Pulse input excites van der Pol oscillator (V) and Net simulation (N); phase plot

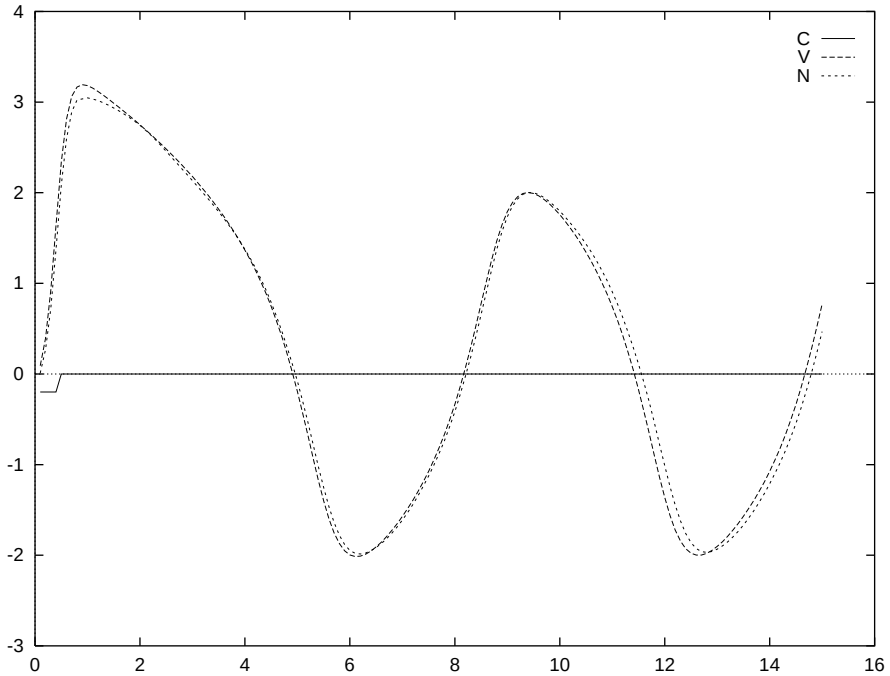

Figure 4: Time plot for van der Pol oscillator