Neural Networks in Nonlinear Aircraft Control

Dennis J. Linse
Department of Mechanical and Aerospace Engineering
Princeton University
Princeton, New Jersey

Recent research indicates that *Artificial Neural Networks* offer interesting learning or adaptive capabilities. The current research focuses on the potential for application of neural networks in a nonlinear aircraft control law. The current work has been to determine which networks are suitable for such an application and how they will fit into a nonlinear control law.

Parameter Estimation in Nonlinear Control

The equations of motion of an aircraft can be cast into a set of nonlinear ordinary differential equations that are linear in the control input, **u**. Included in the state and output equations is a parameter vector, **p**. Several methods, including Nonlinear Inverse Dynamics [Lane and Stengel, 1988] and decoupling [Singh and Rugh, 1972], are available for generating nonlinear feedback laws if the system is known. Using these techniques, the feedback laws are determined as functions of the parameter vector, **p**. **p** itself is a function of the current state and possibly external states and controls not included in the system dynamics. The main difficulty is to provide an estimate of the parameters that are possible complex nonlinear functions of the states.

Given a Nonlinear Dynamic System (e.g. an aircraft)

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}, \mathbf{p}) + \mathbf{G}(\mathbf{x}, \mathbf{p})\mathbf{u}$$

$$\mathbf{y} = \mathbf{h}(\mathbf{x}, \mathbf{p})$$

and a Nonlinear Feedback Law

$$\mathbf{u} = \mathbf{a}(\mathbf{x}, \mathbf{p}) + \mathbf{B}(\mathbf{x}, \mathbf{p})\mathbf{v}$$

where

$$\mathbf{p} = \mathbf{p}(\mathbf{x}, \mathbf{x}_{ext}, \mathbf{u}_{ext})$$

how do you estimate the parameters, p ?

Nonlinear Function Approximation

While many techniques exist for nonlinear function approximation, three have been chosen for further investigation. A B-spline (basis spline) technique with coefficients updated using Recursive Least-Squares estimation represents a classic function approximation method. Two neural networks methods are also investigated. The Back-Propagation Feedforward Network [Rumelhart, Hinton, and Williams, 1986] is a popular, widely investigated, model in the neural network community. The Cerebellar Model Articulation Controller (CMAC) [Albus, 1975] is less well known, but very useful and powerful in function approximation implementations.

Three appealing methods

- Classic method
  - Recursive Least Squares Estimation using B-Splines
- Neural methods
  - Back-Propagation Feedforward Network
  - Cerebellar Model Articulation Controller (CMAC) Network

Comments on Operation and Learning

Each of the three techniques has different adaptation (or learning) capabilities. The B-spline method requires a matrix inversion that may be very slow, especially for high input dimension systems. The least-squares estimation scheme effectively extracts information from the input data, making adaptation fast when measured in terms of number of points presented to the estimator. The feedforward network, on the other hand, needs many presentations to accurately approximate the nonlinear function. Once trained, it can be extremely fast in operation, especially if fully implemented on a VLSI chip. The CMAC quickly approximates the desired function in the neighborhood of the training points and provides good operation speed on a traditional computer architecture.

- Recursive Least Squares B-Splines
  - Slow learning
    Matrix inversion
  - Effective learning
- Back-Propagation Feedforward Networks
  - Relatively Inefficient learning
    Many training points needed
  - *FAST* operation
- CMAC
  - Reasonable learning and operation

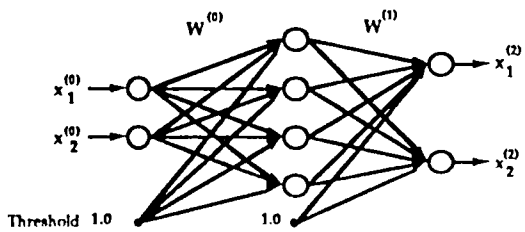
Approximation by Neural Networks

The approximation capabilities of traditional spline techniques are relatively well-known. The capabilities of the neural networks are less well-known. The CMAC approximates by a generalized table look-up. Using overlapping, quantized inputs, the CMAC output is a piece-wise continuous approximation of the input function. The approximation accuracy is limited by the size of the table used in the look-up scheme. The feedforward network provides a continuous approximation of the desired function with the accuracy determined by the number of layers and nodes in the network architecture. There has been much recent interest in exactly determining the approximation capabilities of such networks.

- CMAC's approximate by generalized table look-up
  - Approximation limited by table size
  - Stair-step output
- Feedforward networks
  - Capacity determined by nodes and layers
  - Continuous output
  - Much recent interest in approximation abilities

Back Propagation Neural Networks

A very simple, 2–input / 2–output back-propagation network is given here. The operation of the network can be described by a simple recursive relationship between the outputs of each layer. At each node, a weighted sum of the outputs from all of the nodes of previous layers is acted on by a simple nonlinear function to provide the output for the node. A fixed, unity input is provided to each node to act as a threshold or bias. When the number of layers and nodes in each layer is chosen, the overall nonlinear function calculated by the network is determined by the values of the weights in the interconnections.



$$\mathbf{x}(k) = s[W(k-1)\,\mathbf{x}(k-1)]$$

$$\mathbf{y} = \mathbf{x}(N) = f[\mathbf{x}(0)]$$

Exact Representation using Neural Networks

While investigating the limits of neural networks, Hecht-Nielsen [1987] was able to reinterpret a theorem by Kolmogorov related to the exact representation of a multi-input nonlinear function in terms of simple single-input functions. While back-propagation neural networks can be interpreted in terms of this theorem, the nonlinear function at each node is fixed by a limiting process which depends explicitly on the function to be represented. Most neural networks, on the other hand, have a common, simple function at the network node, and use the weighted interconnections to adjust the output of the network. There are many further difficulties with this theorem that limit its usefulness except as the most basic of existence proofs.

- Hecht-Nielsen's Interpretation of Kolmogorov's Theorem
  (Solution to Hilbert's 13th Problem)
  - Exact Representation

$$f(x) = \sum_{q=0}^{2n} g\left(\sum_{p=1}^{n} \lambda^{p-1} \psi(x_p + \varepsilon q) + q \right)$$

  - ψ can be obtained as a uniform limit of a sequence ψ_r of continuous nondecreasing piecewise linear functions
  - ψ is difficult to compute
  - ψ is different for each $f(x)$

Approximation Representation for Feedforward Networks

In most instances an approximate representation of a nonlinear function is all that is needed. Many researchers have been investigating the approximation capabilities of feedforward networks of the type described. These theorems are usually based on results from functional analysis and give sufficient conditions for the approximation of any continuous function to any desirable degree of accuracy. In general, they are not based on the Kolmogorov's Theorem for exact representation.

- Can feedforward networks approximate nonlinear functions?
- Recent theoretical results by
  Funahashi,
  Cybenko,
  Hecht-Nielsen,
  Stinchcombe and White,
  and many others
- Based on results from functional analysis

Cybenko's Theorem

As an example of these recent theorems, the version due to Cybenko [1989] is outlined here. Defining a sigmoidal function in the most general form, Cybenko's Theorem shows that finite sums of sigmoidal functions, exactly the form of a single hidden layer feedforward neural network, are dense in the continuous functions on the unit hypercube. In simpler terms, any continuous function with inputs between 0 and 1 can be arbitrarily closely approximated by a neural network with one hidden layer. The other researchers have developed similar results. The differences are usually technical details related to how smooth the approximated function is and what type of functions are allowed in the nodes of the hidden layer.

While this is a very promising result, it is only an existence proof. The number of nodes necessary in the hidden layer is only specified as finite, and the weight vector is left unspecified.

Definition: We say that σ is *sigmoidal* if

$$\sigma(t) \rightarrow \begin{cases} 1 \text{ as } t \rightarrow +\infty \\ 0 \text{ as } t \rightarrow -\infty \end{cases}$$

Theorem: Let σ be any continuous sigmoidal function. Then finite sums of the form

$$g(x) = \sum_{j=1}^{N} \alpha_j \sigma(w_j^T x + \theta_j)$$

are dense in $C(I_n)$.

In other words, given $f \in C(I_n)$ and $\varepsilon > 0$, there is a sum, $g(x)$, of the above form, for which

$$|g(x) - f(x)| < \varepsilon$$

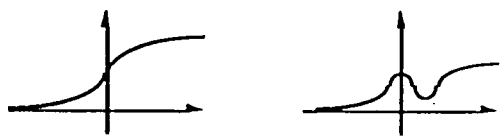
for all $x \in I_n$.

Example Functions

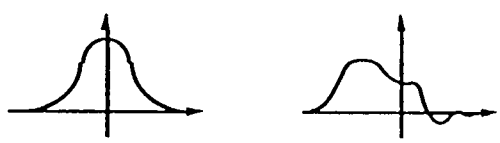
The sigmoidal functions, as defined for Cybenko's Theorem, can have many different shapes, including those shown here. The back-propagation algorithm requires a differentiable nonlinear function at each node. In most implementations the function is similar to the first one shown.

As mentioned previously, the other approximation theorems have different requirements on the nonlinearity allowed in the node. Two are shown here that fail to meet the requirements for Cybenko's Theorem, but are shown to be sufficient for function approximation by others.

- Satisfying Cybenko's Theorem



- Other Theorems



What's Ahead

Future work includes investigating the approximation capabilities of reasonable sized networks for multi-input functions of the type that will be found in a nonlinear control law. This will allow for a complete implementation of a nonlinear control law for the 737 aircraft using neural networks.

- Multidimensional Approximations
- Complete Nonlinear Inverse Dynamics implementation for 737
- Implement Neural Networks in NID control law
- Hope for constructive results regarding Neural Network function approximation