

Generalized Networks For Complex Function Modeling

D. G. Ward
Barron Associates, Inc.

Abstract

A generalized neural network architecture and learning algorithm are proposed that are capable of implementing a wide variety of neural and statistical function estimation paradigms, including basis functions, splines, polynomial neural networks, multi-layer perceptrons, recurrent networks, and others. The discussion begins with a description of a generic nodal element that can perform a number of user-defined linear and nonlinear transformations. These nodal elements are combined into networks using an information-theoretic approach that reduces excess network complexity. Finally, an iterative Gauss-Newton training algorithm is developed, and it is shown how this algorithm may be used to optimize the network for a variety of loss functions. The intent is to provide insight into both neural and statistical modeling by exploring the relationships between existing paradigms and by providing a technique that allows the best aspects of existing paradigms to be combined into novel function estimation strategies.

A Generalized Neural Network Architecture

An artificial neural network is a parallel, distributed information processing structure consisting of multiple interconnected nodal elements. Fig. 1 shows a single fully-interconnected layer of a generalized ANN. Note the feedback connections in the generalized structure; this allows the network to be connected in either a feedforward or recurrent configuration.

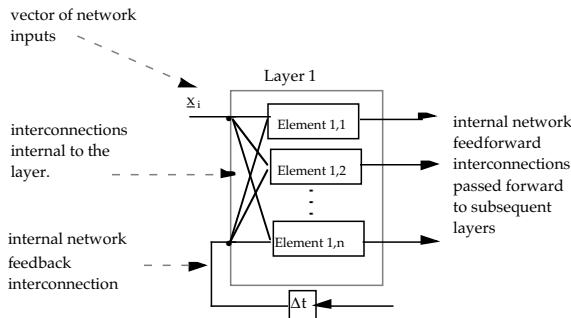


Figure 1: Generalized ANN Structure

Each layer of the generalized ANN is comprised of fundamental building blocks called nodes, elements, or nodal elements; a generalized nodal element is shown in Fig. 2. This generalized nodal element is comprised of three important parts: (1) an algebraic or other series expansion, (2) a fixed linear or non-linear post-transformation function, $h(\cdot)$, and, (3) shift registers or delay banks to allow the series expansion to have access to prior input values. These shift-registers, along with the recurrent interconnections, provide the network with memory. The first two nodal element components are discussed in more detail below:

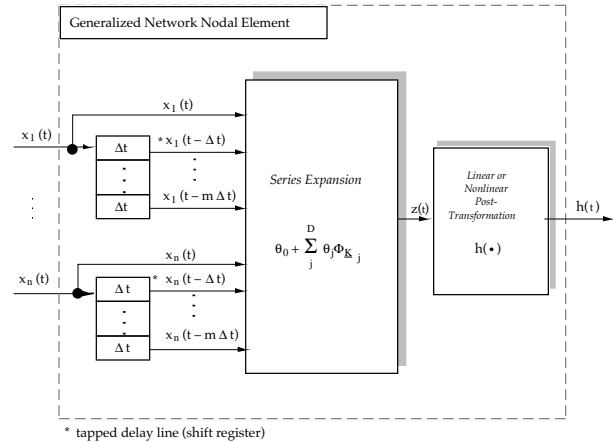


Figure 2: Generalized Network Nodal Element

Basis Functions and Series Expansions

The series expansion of Fig. 2 is of the form

$$z(\underline{x}, \underline{\theta}) = \sum_{j=0}^J \theta_j \Phi(\underline{k}_j, \underline{x}) \quad (1)$$

where $\underline{\theta}$ is the vector of element coefficients, J is the total number of non-constant terms in the expansion, and \underline{k}_j is a vector of integers. A bias term is ensured by requiring that $\Phi(0, \underline{x}) = 1$. The series expansion within a neural network element has the same form as traditional series expansion techniques; however, with network function estimation, it is desirable that the total number of terms in any given element be kept as small as possible. This point will be elaborated on shortly.

The inclusion of \underline{k}_j , sometimes called the set of indices or multi-indices, allows the series expansion to handle both univariate and multivariate cases. For the multivariate case, each $\Phi(\underline{k}_j, \underline{x})$ is a product of functions of scalars. \underline{k}_j is usually taken to be a vector of integers with each element of \underline{k}_j corresponding to one of the variables in the \underline{x} vector. Using this notation, the j th term in the series expansion may be written as:

$$\Phi(\underline{k}_j, \underline{x}) = \Phi(k_{j1}, x_1) \cdot \Phi(k_{j2}, x_2) \cdot \dots \cdot \Phi(k_{jD}, x_D) \quad (2)$$

where D is the total number of inputs to the series expansion (i.e., the total number of outputs of the tapped delay lines).

The notation introduced above (and thus the nodal element) is sufficiently general to implement a variety of basis functions.

Polynomial:

$$\Phi(\underline{k}, \underline{x}) = \underline{x}^{\underline{k}}$$