Feedforward Artificial Neural Network Initialization by Mathematical Models

Andrew J. Meade, Jr.[1], Gregory S. Lind and Boris A. Zeldin
Department of Mechanical Engineering
and Materials Science
Mail Stop 321
William Marsh Rice University
6100 South Main Street
Houston, Texas 77005-1892, USA
Phone: (713) 527-8101 ext. 3590
E-mail: meade@rice.edu

Submitted to the *International Journal of Neural Systems* for publication.

---

1. Author to whom correspondence should be addressed. meade@rice.edu
This work was supported under NASA grant NGT 51230 and Office of Naval Research grant N00014-95-1-0741.

Feedforward Artificial Neural Network Initialization by Mathematical Models

Andrew J. Meade, Jr., Gregory S. Lind and Boris A. Zeldin
Department of Mechanical Engineering and Materials Science
Mail Stop 321
William Marsh Rice University
6100 South Main Street
Houston, Texas 77005-1892, USA

Keywords: Neural networks, neural computing, mathematical modelling.

Abstract

An initialization method is developed for feedforward artificial neural networks trained to model physical systems. The construction requires imposing certain constraints on the values of the input, bias, and output weights. The attribution of certain roles to each of these parameters allows for mapping a polynomial approximation into an artificial neural network architecture. Attention is focused on time-independent partial differential equations modelled by the feedforward artificial neural network architecture, in conjunction with the popular hyperbolic tangent transfer function. Moreover, this approach is shown to be capable of incorporating other smooth neuron transfer functions. A numerical example is presented illustrating the accuracy and utility of the method.

Feedforward Artificial Neural Network Initialization by Mathematical Models

Andrew J. Meade, Jr.[1], Gregory S. Lind[2] and Boris A. Zeldin[3]
William Marsh Rice University, Houston, Texas 77005-1892, USA

1. Introduction

At present, an area of increasing interest is the adaptive data-driven emulation and control of engineering systems by the multi-layer perceptron, also known as the feedforward neural network (FFANN). It is known that neural networks can approximate functions and mathematical operators arbitrarily well as the number of neurons in the network tends to infinity [1-6]. In this regard, FFANNs can be considered to be "universal approximators" capable of describing the input-output relationships of mechanical systems. For engineers, the universal approximation capability of ANNs would be especially useful in the emulation of complex mechanical systems.

The supervised training of a FFANN system can be formulated as the procedure of selecting the network parameters so that the response of the neural network is "close" to the available data. This can be formalized by introducing a measure of the distance (ε) between the exact function, represented by available data ($f_{exp}(\bar{x})$), and its FFANN approximation ($f_a(\bar{x}, \bar{\beta})$) usually written as

$$\varepsilon(\bar{\beta}) = \left\| f_{exp}(\bar{x}) - f_a(\bar{x}, \bar{\beta}) \right\|^2 = \sum_j^S \left| f_{exp}(\bar{x}_j) - \sum_l \beta_{k,l} \sigma^{(k-1)}\left(\ldots \beta_{2,r}^{(q)} \sigma^{(1)}\left(\sum_i^{d+1} \beta_{1,i}^{(r)} x_j^{(i)} \right)\right) \right|^2, \quad (1)$$

where the $(d+1)$-dimensional vector $\bar{x} = (x^{(1)}, \ldots x^{(d)}, 1)$ contains the d-dimensional input of

1. Associate Professor. Author to whom correspondence should be addressed. E-mail: meade@rice.edu.
2. Research Assistant.
3. Research Associate.

the FFANN system, S is the number of available data points, $\bar{\beta} = \{\beta^{(r)}{}_{1,i}, \ldots \beta_{k,l}\}$ represents the set of network parameters, and $\sigma^{(k)}$ is a non-linear transfer function associated with the k-th intermediate layer of the network. This paper considers only single-output FFANN systems. An extension to the multi-output case is straightforward.

The network response $f_a(\bar{x}, \bar{\beta})$ depends linearly on only a few parameters, which are shown in Eq. (1) as the coefficients $\beta_{k,l}$ of the expansion. The remaining parameters affect the network response in a non-linear manner. Correspondingly, supervised training of neural networks is based on non-linear optimization procedures. In terms of the connectionist literature, numerical minimization of this error criterion by the steepest descent method constitutes the backpropagation algorithm [7]. The conjugate gradient method, the Levenberg-Marquardt method, and other non-linear optimization methods have also been used for network training. However, it has been found that even the most advanced optimization algorithms can show unreliable convergence in training [8] and may converge to a poor local minimum or may not reach convergence at all; the latter phenomenon is known as overtraining [9]. There are two major approaches to improving the performance of training algorithms: 1) effective selection of the initial values of the network parameters and 2) effective introduction of network parameter constraints through the incorporation of regularizing functionals

The initial values for the network parameters are commonly selected by random sampling [10]. If a satisfactory solution is not obtained during training, the non-linear optimization procedure is restarted with new randomly selected values [11]. Clearly this approach can require prohibitive computational resources for some practical problems. Alternatively, Lehtokangas et al. [12] proposed that an orthogonal least squares algorithm [13] be used for the initiation of the network parameters. In this case, a large network is constructed by first randomly sampling the non-linear parameters and then determining the corresponding optimal values of the linear parameters. Subsequently, he size of the network is reduced by removing neurons which do not significantly contribute to the network approximation. The deficiency in this approach is the necessity in determining a large intermediate network which can adequately emulate the system of interest; this can still require significant computational effort and therefore diminishes the applicability of the initiation method. Moreover, the size of the network is not of primary concern for a significant class of practical problem if an accurate network approximation can be obtained by adjusting only the linear network parameters. In this regard, Burrows and Niranjan [14] proposed to initialize the network parameters by linearizing the neuron transfer functions. Unfortunately, this method is not satisfactory for emulating highly non-linear input-output relationships.

In the second approach, Poggio and Girosi [15-17] adopted the Tikhonov regularization method [18] by incorporating regularization functionals. Commonly, $\varepsilon(\bar{\beta})$ is calculated by the equation

$$\varepsilon(\bar{\beta}) = \left\| f_{exp}(\bar{x}) - f_a(\bar{x}, \bar{\beta}) \right\|^2 + \lambda \Lambda(f_a(\bar{x}, \bar{\beta})), \tag{2}$$

where λ is a parameter that controls the sensitivity of the objective function to the Tikhonov regularization functional $\Lambda(f_a(\bar{x}, \bar{\beta}))$ This approach was further extended to diverse practical applications [19-22]. Unfortunately, regularization does not address all of the issues related to FFANN training. It merely provides requisite constraints so that an approximation exists for the considered class of neural networks.

The objective of this paper is to develop a strategy, using the two major approaches, in which network modelling of physical systems utilize all available engineering information. In this regard, the Tikhonov regularization functional is replaced with a functional $\Lambda(f_a)$ corresponding to the governing equations of the mechanical system. The initial values of the network parameters are selected so that the network accurately approximates the input-output relationship of the engineering systems' mathematical model. This initialized FFANN model can then be further improved by experimental data through the minimization of Eq. (2), with or without the non-linear parameter constraints. The proposed method is based on three assumptions. Firstly, a FFANN system can be treated as a general approximation scheme [16] capable of approximating differential equations. Secondly, polynomial bases, which are a traditional method for approximating functions and analyzing experimental data [23, 24], can be mapped into neural network architectures. Thirdly, mathematical models of engineering systems, linear and non-linear, can be utilized to obtain initial values for the network parameters.

It is shown in this paper that the approximation of mathematical models using polynomial bases can be an efficient tool for initiating a training procedure in network simulations of physical systems. Note that the proposed approach allows the initiation of a connectionist training algorithm without data; this feature can be especially useful for on-line FFANN modelling systems. Also, the method can be useful in obtaining preliminary bounds on the network performance and estimating the accuracy of network approximations. For example, by exploiting the analogy between FFANN systems and polynomial approximations, the effect of augmenting networks with additional processing elements can be evaluated. Finally, initiating training with a mathematical model of sufficient fidelity significantly reduces problems associated with describing a physical system with an inadequately small training set. With this approach the network parameters are pre-determined by a mathematical model leading to a reliable approximation of a physical system in parameter regions lacking data.

For the sake of brevity attention is focused on modelling an elliptic differential equation by the FFANN architecture with the hyperbolic tangent transfer function. However, the approach should be able to incorporate other smooth transfer functions.

2. FFANN Parameter Initiation Using Mathematical Models.

2.1 Feedforward Artificial Neural Networks

A simple FFANN architecture with two layers of linear processing elements and one layer of non-linear processing elements, as illustrated in Fig. (1), is quite common in network applications. Moreover, the proposed approach shows that a single hidden-layer FFANN system (3-layer FFANN), which is utilized in this study, can be readily associated with some conventional approximation procedures.

Several types of transfer functions have been used in the processing elements of FFANNs, but the more popular training procedures, such as backpropagation, require that the transfer functions σ be continuously differentiable and bounded. In this regard, sigmoidal neurons are often selected for network applications. The corresponding functions are continuous and are characterized by the following limits

$$\lim_{\xi \to \infty} \sigma(\xi) = 1 \text{ and } \lim_{\xi \to -\infty} \sigma(\xi) = -1 . \tag{3}$$

The function

$$\sigma(\xi) = \tanh(\xi) \tag{4}$$

is a paradigmatic example of such sigmoidal functions.

2.2 Training of Feedforward Artificial Neural Networks

The training of the FFANN system is performed by adjusting the network parameters in the matrix $\bar{\beta}$. Specifically,

$$\bar{\beta}^{\text{new}} = \bar{\beta}^{\text{old}} - \alpha \nabla_{\tilde{\beta}} \varepsilon(\bar{\beta}) , \tag{5}$$

where α is the learning rate constant selected by emulating a corresponding non-linear optimization procedure and $\varepsilon(\bar{\beta})$ is the training objective function of Eq. (2). The objective function taken in the form of Eq. (2) readily yields an analytical expression for the gradient

$\nabla_{\tilde{\beta}}\varepsilon(\bar{\beta})$ needed in Eq. (5) [9, 19].

As it has been noted previously, the success in training depends to a great extent on an appropriate selection of the initial values and/or constraints for the network parameters; in this regard, the new procedure is presented in the insuing sections.

2.3 Linear versus Non-Linear Training

The response of a 3-layer FFANN illustrated in Fig. (1) can be written as

$$f_a(\bar{x}, \bar{\beta}) = \sum_r^M \beta_{2,r}\sigma^{(1)}\left(\sum_i^d \beta_{1,i}^{(r)} x^{(i)} + \theta_r\right) = \sum_r^M \beta_{2,r}\sigma_r(\bar{x}), \tag{6}$$

where $\theta_r = \beta_{1,d+1}^{(r)}$ are biases associated with the M neurons of the intermediate layer and where $\sigma_r(\bar{x}) = \sigma^{(1)}\left(\sum_i^d \beta_{1,i}^{(r)} x^{(i)} + \theta_r\right)$. Note that the network response depends linearly on the parameters $\beta_{2,r}$ only, other parameters ($\beta_{1,i}^{(r)}$) affect the network response in a highly non-linear manner. In this regard, initial values of the network parameters adequate for successful network training can be determined efficiently by establishing a set of constraints on the non-linear parameters. The response of these types of neural networks can then be seen as a linear combination of N basis functions $\phi_i(\bar{x})$. Specifically,

$$f_a(\bar{x}, \bar{\beta}) = \sum_i^N c_i\phi_i(\bar{x}). \tag{7}$$

For engineering applications, bases with the following properties could be especially useful:

- The bases should be assembled in a straightforward manner from ANN architectures with parameters that can be evaluated in a numerically stable manner.

- The bases should accurately approximate the desired functions.
- To evaluate the linear coefficients, the bases should allow for utilizing a variety of computational techniques familiar to the engineering community such as finite difference, finite elements and spectral methods.
- The bases should allow the user to control the accuracy of the network approximation.
- The developed bases should have the property of approximating desired functions using a small number of hidden layer neurons and of requiring only a modest increase in the number of neurons to improve modelling accuracy (i.e., good rate of convergence).

In this paper global polynomial bases, with the useful properties described previously, will be formed from linear combinations of the network transfer functions. The linear coefficients (output weights) associated with the bases will be evaluated by linear optimization techniques. This approach readily yields reliable estimates of the initial values of network parameters for subsequent training by non-linear optimization algorithms.

2.4 Sigmoidal Networks and Polynomial Bases Approximation

The idea of mapping a conventional approximation scheme, with well established approximation properties, into neural network architecture has been explored in the literature [3-5, 25, 26] for theoretical studies regarding the density and approximation properties of ANNs and the dependence of the network approximation error on the number of neurons. In this paper, the mapping approach is used with a new and practical intent. Specifically, the well known method of polynomial approximation, commonly used in data analysis and computational mechanics applications, will be mapped into a network architecture for the emulation of mechanical systems.

Proceeding along these lines, attention is drawn to the following formula

$$\sigma_{(\boldsymbol{p})}(\bar{w}, \bar{x}) = \frac{\partial^{|\boldsymbol{p}|}}{\partial w_1^{p_1} \ldots \partial w_s^{p_s}} \sigma(\bar{w}^T \bar{x} + \theta) = \bar{x}^{\boldsymbol{p}} \sigma^{(|\boldsymbol{p}|)}(\bar{w}^T \bar{x} + \theta), \tag{8}$$

where $(\circ)^T$ denotes a vector transpose, $\bar{x}^{\boldsymbol{p}} = x_1^{p_1} \cdot \ldots x_s^{p_s}$, $\boldsymbol{p}$ is a multi-index such that $|\boldsymbol{p}| = p_1 + \ldots p_s$, and σ is a non-linear non-polynomial function. Equation (8) shows that a polynomial $\bar{x}^{\boldsymbol{p}}$ can be readily expressed as

$$\bar{x}^{\boldsymbol{p}} = (\sigma^{(|\boldsymbol{p}|)}(\theta))^{-1} \sigma_{(\boldsymbol{p})}(0, \bar{x}), \tag{9}$$

where the bias θ is selected so that $\sigma^{(|\boldsymbol{p}|)}(\theta)$ is large. Therefore, by replacing the derivative $\sigma_{(\boldsymbol{p})}(0, \bar{x})$ by an adequate finite difference approximation, one can accurately approximate the polynomial $\bar{x}^{\boldsymbol{p}}$ by a finite linear combination of functions in the form $\sigma(\bar{w}_i^T \bar{x} + \theta)$. It has been determined that the following finite difference scheme is adequate for practical applications

$$\sigma_{(\boldsymbol{p})}(0, \bar{x}) \cong \phi(\bar{x}) = (2\Delta w)^{-|\boldsymbol{p}|} \sum_{0 \le r \le p} (-1)^{|\boldsymbol{r}|} \binom{\boldsymbol{p}}{\boldsymbol{r}} \sigma((2\boldsymbol{r} - \boldsymbol{p})^T \Delta w \cdot \bar{x} + \theta), \tag{10}$$

where Δw is a small positive constant and the multi-integer binomial is defined as

$$\binom{\boldsymbol{p}}{\boldsymbol{r}} = \prod_{j=1}^{s} \binom{p_j}{r_j}. \tag{11}$$

Note that the accuracy of the finite difference scheme of Eq. (10), within a bounded domain of $\bar{x}$, is given by the following error bound

$$max\left|\phi(\bar{x}) - \sigma_{(\boldsymbol{p})}(0, \bar{x})\right| \le C(\Delta w)^2. \tag{12}$$

Finally, combining Eqs. (9) and (10) and replacing the function $\sigma(x)$ by the intermediate neuron transfer function one can show that the polynomial $\bar{x}^{\boldsymbol{p}}$ can be approximated by a network with $\prod_{j=1}^{s}(p_j+1)$ neurons in the single hidden layer. The error of this approximation can be made arbitrarily small by the appropriate selection of Δw. For example, the linear function x can be approximated as

$$x \cong \frac{\tanh(\Delta w \cdot x + \theta) - \tanh(-\Delta w \cdot x + \theta)}{2\Delta w \cdot \tanh'(\theta)}. \tag{13}$$

where the prime denotes first-order ordinary differentiation.

Note that the developments of this subsection have been previously utilized in references [25] and [26] for establishing the approximation properties of some types of neural network architectures. However, in the next subsection this approach will be combined with mathematical models of physical systems to obtain a reliable approximation scheme for initiating FFANN training.

2.5 Developing an Initial FFANN Using A-Priori Mathematical Models

Conventional smoothness based Tikhonov regularization of ANN systems, as used in reference [17], may not be consistent with the smoothness required in emulating a physical system. Instead the network regularization can be based on an a-priori mathematical model of some level of fidelity.

A large class of mechanics problems can be described by the equations

$$Lf_0(\bar{x}, \bar{\beta}) = g(\bar{x}), \quad \bar{x} \in \Omega, \tag{14}$$

$$Bf_0(\bar{x}, \bar{\beta}) = 0, \quad \bar{x} \in \partial\Omega, \tag{15}$$

where $f_0(\bar{x}, \bar{\beta})$ is the exact solution to the equations, L is a linear self-adjoined differential operator, $g(\bar{x})$ is the forcing function and B is the boundary operator.

Since the response of the FFANN given by Eq. (7) cannot automatically satisfy Eq. (14), nor satisfy the boundary conditions of the mathematical model, any regularization functional to be developed must incorporate the boundary conditions through the method of Lagrange multipliers. Specifically,

$$\Lambda(f_a) = \frac{1}{2}l\langle f_a, f_a\rangle - \langle f_a, g\rangle + b\langle f_a, q\rangle, \tag{16}$$

where $l\langle \circ, \circ\rangle$, defined as

$$l\langle f, f\rangle = \int_{\Omega}(Lf(\tau))f(\tau)d\tau, \tag{17}$$

is a bilinear continuous form corresponding to the differential operator L and $b\langle \circ, \circ\rangle$ is a bilinear form corresponding to the boundary operator B of Eq. (15) and defined as

$$b\langle f, q\rangle = \int_{\partial\Omega}(Bf(\tau))q(\tau)d\tau. \tag{18}$$

A numerical algorithm for the network modelling can be obtained by substituting Eq. (6) and Eq. (16) into the objective function defined by Eq. (2). After some algebraic manipulation it yields

$$\begin{aligned}\varepsilon(\bar{\beta}) = {} & \left(\sum_{j}^{S} f_{exp}(\bar{x}_j)^2 - 2\sum_{k}^{M}\beta_{2,k}\sum_{i}^{S} f_{exp}(\bar{x}_j)\sigma_k(\bar{x}_j) + \sum_{k,i}^{M}\beta_{2,k}\beta_{2,i}\sum_{i}^{S}\sigma_k(\bar{x}_j)\sigma_i(\bar{x}_j)\right) + \\ & \lambda\left(\frac{1}{2}\sum_{k,i}^{M}\beta_{2,k}\beta_{2,i}l\langle\sigma_k(\bar{x}), \sigma_i(\bar{x})\rangle - \sum_{k}^{M}\beta_{2,k}\langle\sigma_k(\bar{x}), g\rangle + \sum_{k}^{M}\beta_{2,k}b\langle\sigma_k(\bar{x}), q\rangle\right).\end{aligned} \tag{19}$$

where $\bar{x}_j$ represents the *j*-th *d*-dimensional data point. With Eq. (19) the network modelling problem can be formulated as the evaluation of the network parameter set $\bar{\beta}$ and the Lagrange multiplier function q that minimizes $\varepsilon(\bar{\beta})$. A general back-propagation-type algorithm can be developed in this regard. However, in this study a more straightforward option is taken. Assuming the values of $\beta_{1,i}^{(r)}$ in Eq. (6) can be found to produce a set of bases $\phi_i(\bar{x})$, with the properties described in subsection 2.3, the minimum of $\varepsilon(\bar{\beta})$ is found by differentiating Eq. (19) with respect to the linear parameters c_k associated with the basis functions $\phi_k(\bar{x})$. This yields the following system of equations

$$\frac{2}{\lambda}\left(\sum_{i}^{N}\sum_{j=1}^{S}\phi_k(\bar{x}_j)\phi_i(\bar{x}_j)c_i - \sum_{j=1}^{S} f_{exp}(\bar{x}_j)\phi_k(\bar{x}_j)\right) + \left(\sum_{i}^{N} l\langle\phi_k(\bar{x}),\phi_i(\bar{x})\rangle c_i - \langle\phi_k(\bar{x}), g\rangle + b\langle\phi_k(\bar{x}), q\rangle\right) = 0 \quad , \quad \text{for } k = 1,\dots,N, \tag{20}$$

$$\sum_{i}^{N} c_i b\langle\zeta,\phi_i(\bar{x})\rangle = 0, \quad \text{for any } \zeta \in (H^{s-\nu}(\partial\Omega))', \tag{21}$$

where ζ and q belong to an appropriate finite dimensional subspace of $(H^{s-\nu}(\partial\Omega))'$.

To initialize the FFANN with an a-priori mathematical model, the available data $f_{exp}(\bar{x}_j)$ are neglected by allowing $\lambda \to \infty$ so that

$$\sum_{i}^{N} l\langle\phi_k(\bar{x}),\phi_i(\bar{x})\rangle c_i - \langle\phi_k(\bar{x}), g\rangle + b\langle\phi_k(\bar{x}), q\rangle = 0, \quad \text{for } k = 1,\dots,N. \tag{22}$$

The values of c_i that satisfy Eq. (22) and Eq. (21) are then determined. That is, $f_0(\bar{x},\bar{\beta})$ is approximated by the network to an acceptable accuracy.

Note that in the computational mechanics literature Eqs. (22) and (21) are known as the Galerkin method [27]. This observation provides a useful link between the initialization of FFANNs and methods in computational mechanics. Extending this link with computational mechanics, Eq. (22) is rewritten as

$$\sum_i^N [\langle \psi_k(\bar{x}), L(\phi_i(\bar{x}))\rangle] c_i + b\langle \psi_k(\bar{x}), q\rangle = \langle \psi_k(\bar{x}), g\rangle, \quad \text{for } k = 1,\ldots, N, \tag{23}$$

using an arbitrary weighting function $\psi_k(\bar{x})$. This facilitates the utilization of the method of weighted residuals [28], which encompasses many computational mechanics methods. Equation (23) is presented in the matrix form as

$$\sum_i^N A_{ki} c_i + \sum_i^\Gamma B_{ki} q_i = s_k. \tag{24}$$

$$\sum_i^\Gamma B_{ik} c_i = 0 \tag{25}$$

where $A_{ki} = \langle \psi_k(\bar{x}), L(\phi_i(\bar{x}))\rangle$ and $s_k = \langle \psi_k(\bar{x}), g\rangle$. Further, q_i are boundary Lagrange multipliers, the coefficients B_{ki} result from the discretization of the boundary bilinear form in Eq. (21) and Γ denotes the number of boundary basis functions. With the solution of the coefficients c_i, and the preselected values of $\beta_{1,i}^{(r)}$, the approximation of $f_0(\bar{x}, \bar{\beta})$ can be realized as a 3-layer FFANN. The details of the developed procedure will be clarified by means of a specific example.

3. The Poisson Equation as a Mathematical Model

A numerical example is presented in this section to illustrate the usefulness and applicability of the proposed training initiation method for mechanical applications. This example involves modelling the fully developed flow through a square duct.

3.1 Problem Formulation

Assume the user wishes to use a FFANN to emulate and/or control steady incompressible viscous fluid flow in a duct of square cross section as shown in Fig. (2). Using a Cartesian coordinate system the fluid flow is governed by the x-momentum equation ([27], pp. 13-15),

$$u\frac{\partial u}{\partial x} + v\frac{\partial}{\partial y}(u) + w\frac{\partial}{\partial z}(u) - \frac{1}{\rho}\frac{\partial p}{\partial x} = \frac{\mu}{\rho}\left[\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} + \frac{\partial^2 u}{\partial z^2}\right], \tag{26}$$

where u, v, w, ρ, p and μ represent the x, y, z components of fluid velocity, fluid density, pressure and viscosity, respectively. Far from inlet or exit of the channel, the flow does not vary in the x direction and the velocity components v and w are zero, which reduces Eq. (26) to

$$-\frac{1}{\mu}\frac{\partial p}{\partial x} = \frac{\partial^2 u}{\partial y^2} + \frac{\partial^2 u}{\partial z^2}. \tag{27}$$

For this type of flow the pressure variation $\frac{1}{\mu}\frac{\partial p}{\partial x}$ is equal to a constant Υ. Therefore, with the appropriate non-dimensionalization, the fluid flow can be described by the Poisson equation for the dimensionless velocity $\tilde{u} = u/\Upsilon$ and the dimensionless normalized length,

$$\frac{\partial^2 u}{\partial y^2} + \frac{\partial^2 u}{\partial z^2} = -1, \tag{28}$$

where the superscripts have been dropped for notational convenience. Further, Eq. (28) is augmented by the following boundary conditions

$$u = 0 \text{, on } y = \pm 1 \text{ and } z = \pm 1 , \tag{29}$$

The dimensionless dependent variable u is approximated in this problem by the following equation

$$u_a(y, z) \cong u_{poly}(y, z) = g(y, z) \sum_{j=1}^{N2} \sum_{i=1}^{N1} \phi_{ij}(y, z) c_{(j-1) \cdot N1 + i} \tag{30}$$

where $\phi_{ij}(y, z)$ is a general polynomial expansion and $g(y, z)$ is a function that satisfies the boundary conditions identically. Note that since the FFANN can only approximate various polynomials, we must distinguish between the FFANN approximation u_a and the approximation by polynomial expansion u_{poly}. For this example,

$$\phi_{ij}(y, z) = y^{(i-1)} z^{(j-1)} \text{ and } g(y, z) = (1 - y^2)(1 - z^2) . \tag{31}$$

Substitution of Eq. (30) into Eq. (28) results in

$$\frac{\partial^2 u_{poly}}{\partial y^2} + \frac{\partial^2 u_{poly}}{\partial z^2} + 1 = R(y, z) , \tag{32}$$

where *R(y,z)* is the equation residual. Also, we define $y^{(i-2)} = 0$ and $z^{(j-2)} = 0$ for $i = 1$ and $j = 1$, respectively.

3.2 FFANN Formulation

Using the method of weighted residuals Eq. (32) can be rewritten as a series of algebraic equations.

$$\langle \psi_{lk}, R_{ij} \rangle = \langle \psi_{lk}, -1 \rangle \text{, for } l = 1, \ldots, N1 \text{ and } k = 1, \ldots, N2 \tag{33}$$

Using the orthogonal collocation method (p.28, [28]), we set $\psi_{lk} = \delta(y - y_l, z - z_k)$, where δ is the Dirac delta function evaluated at $N1 \cdot N2$ collocation points. The matrices can be rearranged so that Eq. (33) can be written as

$$\sum_{r=1}^{N1 \cdot N2} A_{hr} c_r = s_h \quad \text{for} \quad h = 1, \ldots, N1 \cdot N2 \tag{34}$$

Since equal resolution is desired in both the y and z directions, *N1* and *N2* are set to the same value. After evaluating the coefficients c_r, Eq. (9) is utilized to replace the polynomials of Eq. (31) with weighted linear combinations of hyperbolic tangents and generate the initialized 3-layer FFANN.

We restrict our numerical example to *N1* = *N2* = 1 with $y_l = z_k = \sqrt{0.2}$ ([28], p. 102). Upon writing

$$\sigma(y, z) = \sigma(w_1(y+1) + w_2(y-1) + w_3(z+1) + w_4(z-1) + w_5 y + w_6 z + \theta_1), \tag{35}$$

Eq. (30) becomes

$$u_{poly} = \frac{c_1}{\sigma^{(|4|)}(\theta_1)} \cdot \frac{\partial^4 \sigma}{\partial w_1 \ldots \partial w_4} \cong u_a = \sum_{l=1}^{M} \beta_{2,l} \sigma_l \tag{36}$$

evaluated at $w_1 = \ldots = w_6 = 0$. The initialized 3-layer FFANN described by u_a of Eq. (36) can then be substituted into Eq. (19), with a finite value for λ, where the FFANN approximation can be improved with experimental data through backpropagation with or without constraints on the non-linear network parameters.

3.3 Numerical Results

A 3-layer FFANN that implements the relationship of Eq. (36) was constructed with $\Delta w_1 = \ldots = \Delta w_4 = 0.04$ and $\theta_1 = -0.6585$ using two neurons with linear transfer functions in the initial layer, 16 neurons using hyperbolic transfer functions in the intermediate layer, and one neuron with a linear transfer function in the final layer. The response of the FFANN system is plotted in Fig. (3) and the distribution of the absolute value of the error from the network approximation, using the exact analytical solution given in reference [27], is shown in Fig. (4). Note that while the FFANN model does not satisfy the boundary conditions exactly, the root mean square of the error (RMS = 7.28×10^{-3}) is only slightly greater than that for the polynomial approximation (RMS = 7.25×10^{-3}). This single hidden-layer FFANN emulates the mechanical system governed by the Poisson equation with acceptable engineering accuracy.

4. Conclusion

The adaptive data-driven emulation and control of mechanical systems are popular applications of artificial neural networks in engineering. However, because ANN training is a non-linear optimization problem it is difficult to apply this attractive modelling tool to engineering applications. As a remedy, a new initiation method for connectionist training algorithms has been presented. The method is formulated on building global polynomial bases from linear combinations of smooth transfer functions and then mapping the polynomial bases into a FFANN architecture. A-priori mathematical models of engineering systems can then be used to obtain initial values for the network parameters. It is noted that the proposed method allows for the initialization of a connectionist on-line training algorithm without data. Also, since polynomial bases are well known in conventional computational mechanics, the method can be useful in obtaining preliminary bounds on the network performance and estimating the accuracy of network approximations. With this approach the network parameters are pre-determined by a mathematical model leading to a reliable approximation of a physical system in parameter regions lacking data.

For the sake of brevity attention has been focused on modelling an elliptic partial differential equation by the feedforward artificial neural network architecture, using the hyperbolic tangent transfer function. A numerical example has illustrated the accuracy and utility of the initialization method. Since the approach needs only smooth transfer functions to form conventional polynomial bases, it should also be applicable to linear and non-linear spatially dependent equations [23] with other smooth transfer functions.

Acknowledgments

This work was supported under NASA grant NGT 51230 and Office of Naval Research grant N00014-95-1-0741.

References

1. Cybenko, G., "Approximation by Superposition of a Sigmoidal Function," *Mathematics of Control, Signals, and Systems*, Vol. 2, 1989, pp. 303-314.

2. Girosi, F., and Poggio, T., "Networks and the Best Approximation Property," *Biological Cybernetics*, Vol. 63, 1990, pp. 169-176.

3. Hornik, K., Stinchcombe, M., and White, H., "Multilayer Feedforward Networks are Universal Approximators," *Neural Networks*, Vol. 2, 1989, pp. 359-366.

4. Hornik, K., Stinchcombe, M., and White, H., "Universal Approximation of an Unknown Mapping and its Derivatives Using Multilayer Feedforward Networks," *Neural Networks*, Vol. 3, 1990, pp. 551-560.

5. Ito, Y., "Representation of Functions by Superpositions of a Step or Sigmoidal Function and Their Applications to Neural Network Theory," *Neural Networks*, Vol. 4, 1991, pp. 385-

394.

6. Park J., and Sanberg, I.W., “Universal Approximation Using Radial-Basis-Function Networks,” *Neural Computation* 3, 1991, pp. 246-257.

7. Rumelhart, D., Hinton, G.E., and Williams, R.J., “Learning Internal Representations by Error Propagation,” in *Parallel Distributed Processing: Exploration in the Microstructure of Cognition*, D. Rumelhart, J.L. McClelland, and the PDP Research Group, eds., Ch. 8, 1986, MIT Press, Cambridge, MA, pp. 318-362.

8. Saarinen, S., Bramley, R., and Cybenko, G., “Ill-Conditioning in Neural Network Training Problems,” *SIAM J. Sci. Comput*. Vol. 14, No. 3, 1993, pp. 693-714.

9. Hecht-Nielsen, R., *Neurocomputing*, Addison-Wesley, Inc., New York, 1990.

10. Igelnik, B., and Pao, Y.-H., “Stochastic Choice of Basis Function in Adaptive Function Approximation and the Function-Link Net”, *IEEE Transactions on Neural Networks*, Vol. 6, No. 6, 1995, pp. 1320-1329.

11. Schidt, W.F., Raudys, S., Kraaijveld, M.A., Skurikhina, M., and Duin, R.P.W., “Initializations, Back-Propagation and Generalization of Feed-Forward Classifiers”, *Proceedings of IEEE International Conference on Neural Networks, ICNN-93*, Vol 1, 1993, pp. 598-604.

12. Lehtokangas, M., Saarinen, J., Kaski, K., and Huuntanen, P., (1995). “Initializing Weights of a Multilayer Perceptron Network by Using the Orthogonal Least Squares Algorithm”, *Neural Computation*, Vol. 7, 1995, pp. 982-999.

13. Chen, S., Cowan, C.F.N., and Grant, P.M., “Orthogonal Least Squares Learning Algorithm for Radial Basis Function Networks”, *IEEE Transactions on Neural Networks*, Vol. 2, No. 2, 1991, pp. 302-309.

14. Burrows, T.L., and Niranjan, M., *The Use of Feed-Forward and Recurrent Neural Networks for System Identification*, Technical Report CUED/F-INFENG/TR158, Cambridge

University Engineering Department, Cambridge, England, 1993.

15. Poggio, T., and Girosi, F., "Regularization Algorithms for Learning that are Equivalent to Multilayer Networks," *Science*, Vol. 247, 1990, pp. 978-982.

16. Poggio, T., and Girosi, F. "Networks for Approximation and Learning," *Proceedings of the IEEE*, Vol. 78, No. 9, 1990, pp. 1481-1497.

17. Girosi, F., Jones, M., and Poggio, T., "Regularization Theory and Neural Network Architectures," *Neural Computation*, Vol. 7, 1995, pp. 219-269.

18. Tikhonov, A. N., and Arsenin, V.Y., *Solution of Ill-Posed Problems,* W.H. Winston, Washington, D.C., 1977.

19. Bishop, C.M., "Curvature-Driven Smoothing: A Learning Algorithm for Feedforward Networks," *IEEE Transactions on Neural Networks*, Vol. 4, No. 5, 1993, pp. 882-884.

20. Bishop, C.M., "Training with Noise is Equivalent to Tikhonov Regularization," *Neural Computation*, Vol. 7, 1995, pp. 108-116.

21. Webb, A.R., "Functional Approximation by Feed-Forward Networks: A Least Square Approach to Generalization," *IEEE Transactions on Neural Networks*, Vol. 5, No. 3, 1994, pp. 363-371.

22. Reed, R., Marks, R.J., and Oh, S., "Similarity of Error Regularization, Sigmoid Gain Scaling, Target Smoothing, and Training with Jitter," *IEEE Transactions on Neural Networks*, Vol. 6, No. 3, 1995, pp. 529-538.

23. Canto, C., Hussaini, M.Y., Quarteroni, A., Zang, T.A., *Spectral Methods in Fluid Dynamics*, Springer-Verlag, Berlin Heidelberg, 1988.

24. Box, G.E.P., and Draper, N.R. *Empirical Model-Building and Response Surface*, Wiley, New York, 1987.

25. Leshno, M., Lin, V.Y., Pinkus, A., and Schocken, S., "Multilayer Feedforward Networks

with a Nonpolynomial Activation Function Can Approximate Any Function", *Neural Networks*, Vol. 6, 1993, pp. 861-867.

26. Mhaskar, H.N., "Neural Networks for Optimal Approximation of Smooth and Analytic Functions", *Neural Computation*, Vol. 8, 1995, pp. 164-177.

27. Fletcher, C.A.J., *Computational Galerkin Methods*, Springer, New York, 1984.

28. Finlayson, B.A., *The Method of Weighted Residuals and Variational Principles*, Academic Press, New York, 1972.

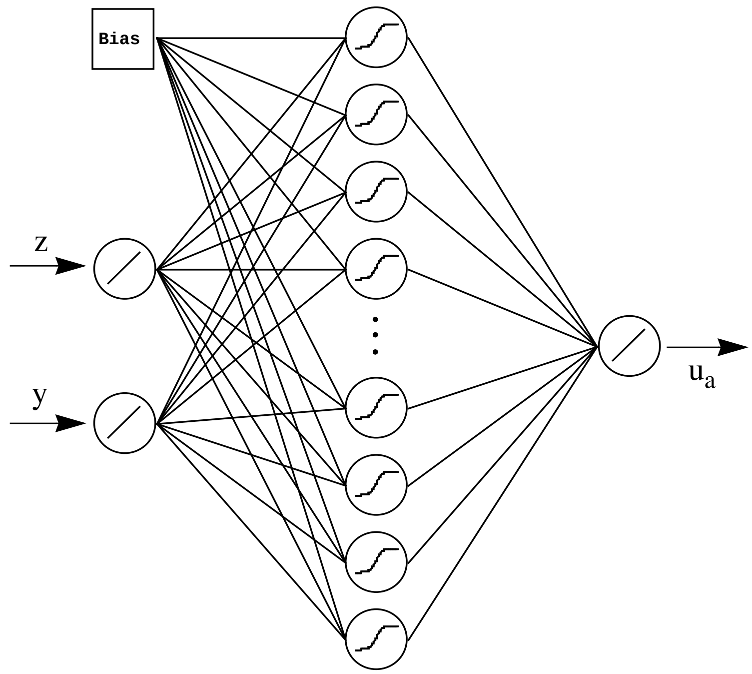

Figure 1. FFANN architecture.

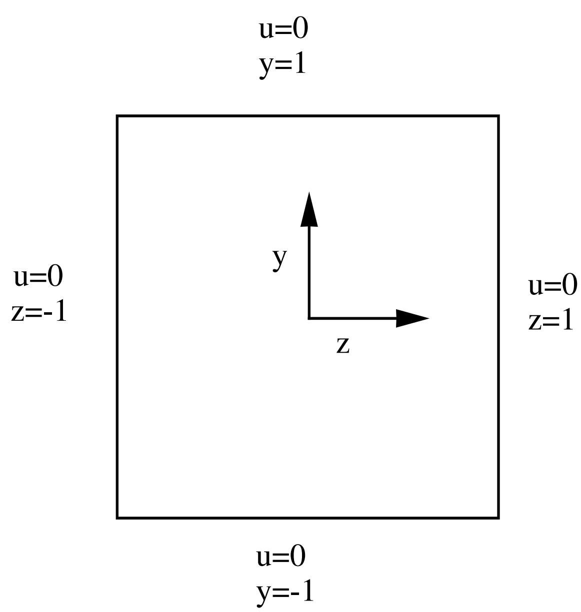

Figure 2. Problem domain and boundary conditions

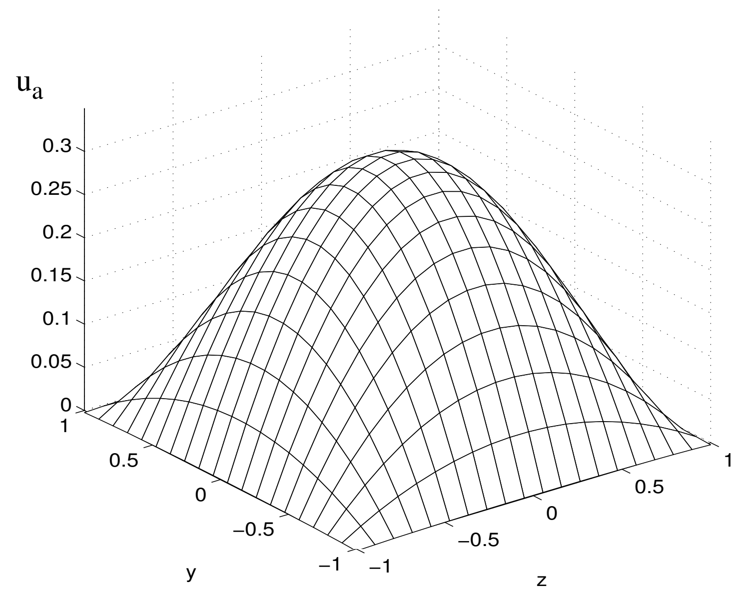

Figure 3. The FFANN model of the Poisson equation

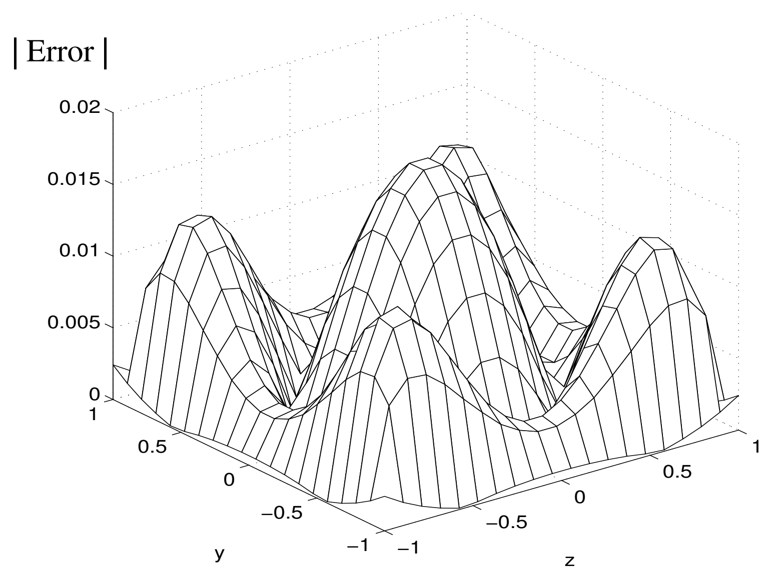

Figure 4. Distribution of absolute value of error for the FFANN model