

NEAR-OPTIMAL FLIGHT LOAD SYNTHESIS USING NEURAL NETS

Michael T. Manry, Cheng-Hsiung Hsieh, and Hema Chandrasekaran
Department of Electrical Engineering
University of Texas at Arlington
Arlington, Texas 76019
e-mail : manry@uta.edu

Abstract

This paper describes the use of neural networks for near-optimal helicopter flight load synthesis (FLS), which is the process of estimating mechanical loads during helicopter flight, using cockpit measurements. First, modular neural networks are used to develop statistical signal models of the cockpit measurements as a function of the loads. Then Cramer-Rao maximum a-posteriori bounds on the mean-squared error are calculated. Then, multilayer perceptrons for FLS are designed which approximately attain the bounds. It is shown that all of the FLS networks have good generalization.

I. INTRODUCTION

Rotor systems in helicopters suffer extreme vibrations. As a result, system components accumulate amounts of fatigue damage which vary considerably with different types of flight. Since it is necessary to avoid rotor system component failure, a conservative usage spectrum is used which estimates the amount of time spent in each type of flight. Unfortunately, the usage spectrum (1) leads to early retirement of some components and a resulting waste of money, and (2) is difficult to keep track of for individual aircraft.

Flight Load Synthesis (FLS) is one approach to solving these problems. In FLS, cockpit measurements are processed into estimates of the mechanical loads encountered by critical parts. Accumulated loads, rather than a usage spectrum, can then be used to determine component retirement times. The loads estimated from FLS should allow for longer part lifetimes where appropriate. Here, we develop near-optimal neural networks for FLS using data from Bell Helicopter Textron in Fort Worth.

II. DESCRIPTION OF DATA FILES

In FLS, the goal is to process cockpit measurements called features into useful

loads. In this section, we describe the features and loads found in data provided by Bell Helicopter. We then describe the corresponding training data files produced for neural networks.

The first set of FLS experiments uses parameters that are available in the basic health usage monitoring system (HUMS), plus some others. The data was obtained from the M430 flight load level survey conducted in Mirabel Canada in early 1995. The input features include: (1) CG F/A load factor, (2) CG lateral load factor, (3) CG normal load factor, (4) pitch attitude, (5) pitch rate, (6) roll attitude, (7) roll rate, (8) yaw rate, (9) corrected airspeed, (10) rate of climb, (11) longitudinal cyclic stick position, (12) pedal position, (13) collective stick position, (14) lateral cyclic stick position, (15) main rotor mast torque, (16) main rotor mast rpm, (17) density ratio, (18) F/A acceleration, transmission, (19) lateral acceleration, transmission, (20) vertical acceleration, transmission, (21) left hand forward pylon link, (22) right hand forward pylon link, (23) left hand aft pylon link, and (24) right hand aft pylon link.

The desired output loads are as follows: (1) fore/aft cyclic boost tube oscillatory axial load (OAL), (2) lateral cyclic boost tube OAL, (3) collective boost tube OAL, (4) main rotor (MR) pitch link OAL, (5) MR mast oscillatory perpendicular bending st., (6) MR yoke oscillatory beam bending sta., (7) MR blade oscillatory beam bending sta., (8) MR yoke oscillatory chord bending sta., and (9) resultant mast bending, sta.position

Three training data files were produced. File F17.dat used features 1 through 17 and all nine output loads. The features correspond to those used on the basic HUMS. File F20.dat used features 1 through 20 and all nine output loads. File F24.dat used features 1 through 24 and all nine output loads.

III. OBTAINING BOUNDS ON ESTIMATION ERROR VARIANCES

Cramer-Rao maximum a-posteriori (CRMAP) bounds [1] have been shown to provide lower bounds on neural network testing error variances [2,3]. Specifically, let the neural network error E be defined as

$$E \equiv \sum_{i=1}^M MSE_i \quad (1)$$

where MSE_i denotes the error for the i th output parameter (load) and M denotes the number of outputs (number of loads). E and MSE_i denote training error or testing error, depending upon the situation. The CRMAP bound on $\text{var}(\theta_i' - \theta_i)$, the variance of the error between the i th output (load) θ_i and the i th output's estimate θ_i' is denoted by B_i . Therefore,

$$\begin{aligned} MSE_i &\geq B_i \\ E &\geq \sum_{i=1}^M B_i \end{aligned} \quad (2)$$

When we have equality in (2), the estimates are usually optimal. In this section, we show the details of how CRMAP bounds are calculated from training data.

A. Getting Bounds

In this subsection, we describe how to calculate the M by M MAP Fisher information matrix [3] (FIM), given statistics on the training data. Let $\{\mathbf{x}_p, \boldsymbol{\theta}_p\}$ for $1 \leq p \leq N_v$ represent the training set for a neural network. Here, \mathbf{x}_p represents the pth example of the N-dimensional random input vector \mathbf{x} and $\boldsymbol{\theta}_p$ represents the pth example of the M-dimensional random parameter vector $\boldsymbol{\theta}$.

Elements of the M by M MAP FIM, \mathbf{J}^{MAP} are defined as

$$\begin{aligned} J_{ij}^{MAP} &= E_{\theta}[J_{ij}^{MLE}] + E_{\theta}\left[\frac{\partial \Lambda^{AP}}{\partial \theta_i} \frac{\partial \Lambda^{AP}}{\partial \theta_j}\right], \\ J_{ij}^{MLE} &\equiv E_n\left[\frac{\partial \Lambda^{MLE}}{\partial \theta_i} \frac{\partial \Lambda^{MLE}}{\partial \theta_j}\right] \end{aligned} \quad (3)$$

where $E_{\theta}[\cdot]$ denotes expected value over the parameter vector $\boldsymbol{\theta}$ and $E_n[\cdot]$ denotes expected value over the noise, where $\Lambda^{MAP} = \ell n(p_{\theta|\mathbf{x}}(\boldsymbol{\theta}|\mathbf{x}))$, $\Lambda^{MLE} = \ell n(p_{\mathbf{x}|\theta}(\mathbf{x}|\boldsymbol{\theta}))$, and $\Lambda^{AP} = \ell n(p_{\theta}(\boldsymbol{\theta}))$. Assume that the elements $x(k)$ of \mathbf{x} are modeled as

$$x(k) = s(k) + n(k) \quad (4)$$

where the elements $s(n)$ and $n(n)$ are respectively elements of the signal and noise vectors \mathbf{s} and \mathbf{n} . \mathbf{s} is a deterministic function of the parameter vector $\boldsymbol{\theta}$. The N by N covariance matrix of \mathbf{x} or \mathbf{n} is denoted by \mathbf{C}_{nn} . The elements of \mathbf{J}^{MAP} in (3) can now be evaluated as

$$\begin{aligned} E_{\theta}[J_{ij}^{MLE}] &= E_{\theta}\left[\left(\frac{\partial \mathbf{s}}{\partial \theta_i}\right)^T \mathbf{C}_{nn}^{-1} \left(\frac{\partial \mathbf{s}}{\partial \theta_j}\right)\right], \\ E_{\theta}\left[\frac{\partial \Lambda^{AP}}{\partial \theta_i} \frac{\partial \Lambda^{AP}}{\partial \theta_j}\right] &= d_{\theta}(i,j) \end{aligned} \quad (5)$$

where \mathbf{C}_{θ} denotes the M by M covariance matrix of the M-dimensional parameter vector $\boldsymbol{\theta}$ and $d_{\theta}(i,j)$ denotes an element of \mathbf{C}_{θ}^{-1} . Let $(\mathbf{J}^{MAP})^{ij}$ denote an element of $(\mathbf{J}^{MAP})^{-1}$. Then [3],

$$\text{var}(\theta_i' - \theta_i) \geq (\mathbf{J}^{MAP})^{ii} \quad (6)$$

where θ_i' can be any estimate of θ_i . The B_i in (2) are the $(\mathbf{J}^{MAP})^{ii}$ of (6).

In order to calculate the log-likelihood functions and the CRM MAP lower bounds on the variance of the parameter estimates, a statistical model of the input vector \mathbf{x} is required. This model consists of a deterministic expression for the signal vector \mathbf{s} in terms of the parameter vector $\boldsymbol{\theta}$, the joint probability density of the additive noise vector \mathbf{n} . In most applications however, the signal model is unknown and bound calculation is not possible. One approach to this problem is to create an approximate signal model from the given training data.

B. Signal Modeling

Given the training patterns for our estimator, we want to find the signal

$$\mathbf{x}_p = \mathbf{s}_p + \mathbf{n}_p \quad (7)$$

component model and noise probability density function (pdf). We make the following assumptions, which make this problem solvable.

(A1) The unknown, exact signal model of (4) can be written in vector form as where \mathbf{x}_p and $\boldsymbol{\theta}_p$ are defined as before, \mathbf{s}_p denotes the p th example of \mathbf{s} , and \mathbf{n}_p denotes the noise component of \mathbf{x}_p .

(A2) The elements $\theta(k)$ of $\boldsymbol{\theta}$ are statistically independent of \mathbf{n} .

(A3) The noise vector \mathbf{n} has independent elements with a jointly Gaussian pdf.

(A4) The mapping $\mathbf{s}(\boldsymbol{\theta})$ is bijective.

1. Basic Approach

From (5) and the assumptions, we need to find a differentiable deterministic input signal model $\mathbf{s}(\boldsymbol{\theta})$ and the statistics of \mathbf{n} and $\boldsymbol{\theta}$. Our first step is to rewrite the signal model of (7) as

$$\mathbf{x}_p = \mathbf{s}'_p + \mathbf{n}'_p \quad (8)$$

where \mathbf{s}'_p and \mathbf{n}'_p denote approximations to \mathbf{s}_p and \mathbf{n}_p respectively. Noting that \mathbf{s} is a function of the desired output $\boldsymbol{\theta}$, we propose to approximate the n th element of \mathbf{s}_p with an inverse neural net,

$$s'_p(n, \mathbf{w}) = \sum_{k=1}^{N_u} w_o(n, k) f_p(k, \mathbf{w}_i) \quad (9)$$

for $1 \leq n \leq N$ where $w_o(n, k)$ denotes the coefficient of $f_p(k, \mathbf{w}_i)$ in the approximation to $s_p(n)$, $f_p(k, \mathbf{w}_i)$ is the k th input or hidden unit in the network, \mathbf{w}_i a vector of weights connecting the input layer to a single hidden layer, and N_u is the number of units feeding the output layer. The vector \mathbf{w} is the concatenation of the input weight vector \mathbf{w}_i with the output weight vector whose elements are $w_o(n, k)$. Note that $f_p(k, \mathbf{w}_i)$ can represent a multinomial function of parameter vector $\boldsymbol{\theta}$ in a functional link network [4] or a hidden unit output in a MLP or radial basis function network [5]. In practice, because of the capabilities of the MLP for approximating derivatives [6,7], a MLP neural network would be the first choice for generating \mathbf{s}' .

Before we can find the neural net weight vector \mathbf{w} in (9), we must have an error function to minimize during training, which includes desired output vectors. Since we want \mathbf{s}'_p to approximate \mathbf{s}_p , the natural choice for the desired output is \mathbf{s}_p . Because \mathbf{s}_p is unavailable, we can try using \mathbf{x}_p as the desired output vector, which yields the error function for the n th output node,

$$E_x(n) = \frac{1}{N_v} \sum_{p=1}^{N_v} [x_p(n) - s'_p(n, \mathbf{w})]^2 \quad (10)$$

The hope is that by minimizing $E_x(n)$ with respect to \mathbf{w} , we are simultaneously

approximately minimizing

$$E_s(n) = \frac{1}{N_v} \sum_{p=1}^{N_v} [s_p(n) - s_p'(n, \mathbf{w})]^2 \quad (11)$$

The signal model is determined from the noisy data by using a gradient approach, such as backpropagation (BP) or output weight optimization (OWO) [8], to minimize $E_x(n)$ with respect to \mathbf{w} , whose elements are denoted by $w(m)$. The gradient of $E_x(n)$ is

$$\frac{\partial E_x(n)}{\partial w(m)} = \frac{-2}{N_v} \sum_{p=1}^{N_v} [x_p(n) - s_p'(n, \mathbf{w})] \frac{\partial s_p'(n, \mathbf{w})}{\partial w(m)} \quad (12)$$

A MLP network for obtaining the signal model is shown in Fig. 1.

Given a model \mathbf{s}_p' for the deterministic signal component and given the noisy input vectors \mathbf{x}_p , we get \mathbf{n}_p' from (8) as

$$\mathbf{n}_p' = \mathbf{x}_p - \mathbf{s}_p'$$

The mean vector and covariance matrix of the noise component are then estimated as

$$\begin{aligned} \mathbf{m}'_n &= \frac{1}{N_v} \sum_{p=1}^{N_v} \mathbf{n}'_p, \\ \mathbf{C}'_{mn} &= \frac{1}{N_v} \sum_{p=1}^{N_v} (\mathbf{n}'_p - \mathbf{m}'_n)(\mathbf{n}'_p - \mathbf{m}'_n)^T \end{aligned} \quad (13)$$

In addition to the statistics of \mathbf{n} , calculation of the CRMAP bounds in (3) requires knowledge of the statistics of $\boldsymbol{\theta}$, which are

$$\begin{aligned} \mathbf{m}'_{\theta} &= \frac{1}{N_v} \sum_{p=1}^{N_v} \boldsymbol{\theta}_p, \\ \mathbf{C}'_{\theta} &= \frac{1}{N_v} \sum_{p=1}^{N_v} (\boldsymbol{\theta}_p - \mathbf{m}'_{\theta})(\boldsymbol{\theta}_p - \mathbf{m}'_{\theta})^T \end{aligned} \quad (14)$$

When $\boldsymbol{\theta}$ is non-Gaussian it is still possible to calculate good bounds using our approach. Theoretical justification of this can be found in [9,10].

C. Convergence of the Method

Since we are minimizing $E_x(n)$ rather than $E_s(n)$ in the previous subsection, it is not clear that \mathbf{s}_p' is a good model for \mathbf{s}_p . In this subsection, we analyze the convergence of the gradient of $E_x(n)$ to the gradient of $E_s(n)$ with the following theorem.

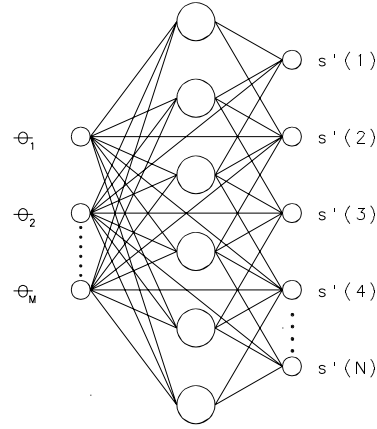


Figure 1. Inverse Network for Signal Modeling

Theorem: Assume that the training of the input weight vector \mathbf{w}_i is stopped after iteration number N_{it} which is fixed and bounded, and that training of the output weights $w_o(n,k)$ is allowed to proceed. Then, in the limit as N_v approaches infinity, $\partial E_x(n)/\partial w(m) = \partial E_s(n)/\partial w(m)$.

Proof: The proof is divided into three parts.

1. Energy of the Gradient Noise

Continuing (12) we get

$$\begin{aligned} \frac{\partial E_x(n)}{\partial w(m)} &= \frac{-2}{N_v} \sum_{p=1}^{N_v} [s_p(n) - s_p'(n, \mathbf{w})] \frac{\partial s_p'(n, \mathbf{w})}{\partial w(m)} + e_{nm} \\ &= \frac{\partial E_s(n)}{\partial w(m)} + e_{nm}, \\ e_{nm} &= \frac{2}{N_v} \sum_{p=1}^{N_v} n_p(n) \frac{\partial s_p'(n, \mathbf{w})}{\partial w(m)} \end{aligned} \quad (15)$$

Using the fact that

$$E[n_p(n)n_q(n)] = \sigma_n^2 \delta(p-q)$$

the mean-square of the noise term e_{nm} is evaluated as

$$\begin{aligned} E[e_{nm}^2] &= \frac{4}{N_v^2} \sum_{p=1}^{N_v} \sum_{q=1}^{N_v} E[n_p(n)n_q(n)] \frac{\partial s_p'(n, \mathbf{w})}{\partial w(m)} \frac{\partial s_q'(n, \mathbf{w})}{\partial w(m)} \\ &= \frac{4\sigma_n^2}{N_v^2} \sum_{p=1}^{N_v} \left(\frac{\partial s_p'(n, \mathbf{w})}{\partial w(m)} \right)^2 \\ &= \frac{4\sigma_n^2 E_m}{N_v} \end{aligned}$$

where E_m is the average energy of the partial derivative of $s_p'(n, \mathbf{w})$ with respect to $w(m)$ and σ_n^2 is the variance of $n(n)$. It remains for us to show that E_m is bounded.

2. E_m for the Output Weight Case

First, assume that $w(m)$ corresponds to an output weight $w_o(n,j)$. Then from (9),

$$\begin{aligned} \frac{\partial s_p'(n, \mathbf{w})}{\partial w(m)} &= f_p'(j, \mathbf{w}_i), \\ E_m &= \frac{1}{N_v} \sum_{p=1}^{N_v} f_p^2(j, \mathbf{w}_i) \end{aligned}$$

Here, the terms in E_m are input or hidden unit activations, or multinomial combinations of inputs for the functional link case. These terms in E_m are bounded if the inverse network inputs (θ_i) are bounded and if the activation functions are bounded. If (9) represents a functional link net, bounding of the inputs produces

bounding of the multinomials $f_p(j, \mathbf{w}_i)$ and E_m .

3. E_m for the Input Weight Case

Assume that $w(m)$ corresponds to a weight $w_i(u)$ which feeds unit number j (a hidden unit). Then

$$\frac{\partial s'_p(n, \mathbf{w})}{\partial w(m)} = w_o(n, j) \frac{\partial f_p(j, \mathbf{w}_i)}{\partial w_i(u)},$$

$$E_m = \frac{w_o^2(n, j)}{N_v} \sum_{p=1}^{N_v} \left(\frac{\partial f_p(j, \mathbf{w}_i)}{\partial w_i(u)} \right)^2 \quad (16)$$

Functional link nets have no input weight vector and have $E_m = 0$ for the input weight case. For MLP and RBF networks, consider the two factors in E_m separately. In the second factor, partials of $f_p(j, \mathbf{w}_i)$ are bounded for bounded activations. Unfortunately, the $w_o^2(n, j)$ term in the first factor in (16) is not bounded in general. Its value depends upon the initial weight values, the training method chosen, and the learning parameters. This can be solved by stopping the training of the input weight vector \mathbf{w}_i after a bounded number of iterations, N_{it} . After this iteration, the vector \mathbf{w} consists only of elements $w_o(n, k)$, and we have only the output weight case.

In the limit, $E_x(n)$ and $E_s(n)$ have equal derivatives and the same local and global minima.

We have developed CRMAP lower bounds on FLS error variance, given the data file described in the previous section, and using the approach described in [11].

IV. Designing Networks for Flight Load Synthesis

After the analyses summarized in section III, we should expect the training errors FLS networks designed for 17, 20, and 24 inputs to be around $.30 \times 10^8$, $.29 \times 10^8$, and $.28 \times 10^8$ respectively. In this section we design MLPs for the FLS data.

A. MLP Training Results

For all data files three-layer MLP structures are employed. The structures 17-13-9, 20-13-9, and 24-11-9 are for data files F17.DAT, F20.DAT, and F24.DAT, respectively. The number of training iterations for each case is 50. Figures 1, 2, and 3 show training MSE versus number of iterations for all data files. Each figure also has CRMAP bounds obtained from an MLP signal model and from a Modular net [12,13] signal model. The average computation time per iteration, on 486DX 50MHz machine, for data files F17.DAT, F20.DAT, and F24.DAT are 15.43 seconds, 17.57 seconds, and 17.28 seconds, respectively.

From figures 1-3, we see that bounds from modular net signal models are smaller than those from MLP signal models. It remains to be seen which bounds are more correct for testing errors.

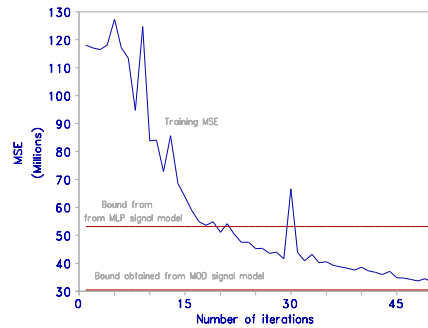


Fig. 1. MLP Training Results and Total Bound for File F17.DAT

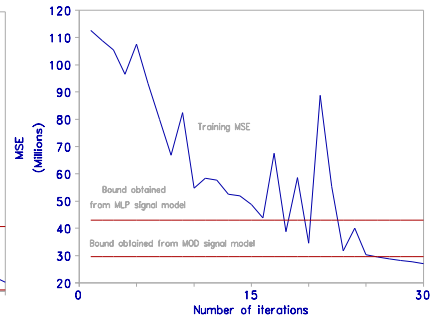


Fig. 2. MLP Training Results and Total Bound for File F20.DAT

B. Testing Results for Designed FLS Networks

Lacking testing data for the FLS problem, we split each original FLS data set into two subsets: one for training and the other for testing. By doing so, we can evaluate the generalization ability of the designed FLS networks. The ratio of the training set to testing set sizes is 3 to 2. The number of training patterns is 2823 and the number of testing patterns is 1922. For the new training data sets, the structures 17-18-9, 20-12-9, and 24-2-9 are used for 17-input, 20-input, and 24-input files, respectively. All networks were trained for 50 iterations. The testing MSEs are obtained when the designed FLS networks are trained after 10, 20, 30, 40, and 50 iterations. The training and testing results are shown in figures 4 to 6, which are respectively for files F17.DAT, F20.DAT, and F24.DAT.

Comparing figures 1 and 4, 2 and 5, and 3 and 6, we see that the modular net signal models provide good lower bounds on testing error. We also see that the FLS networks are close to optimal.

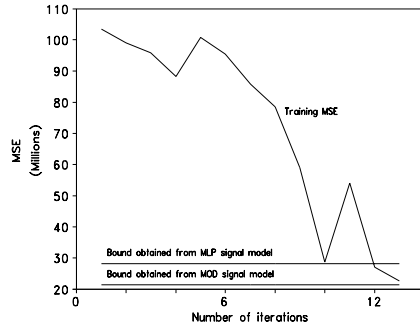


Fig. 3. MLP Training Results and Total Bound for File F24.DAT

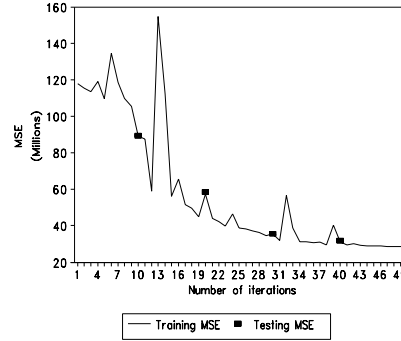


Fig. 4. MLP Training and Testing Results (F17.DAT)

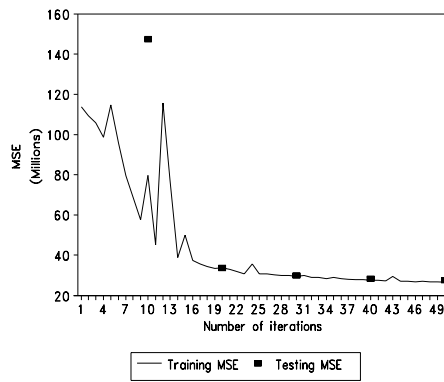


Fig. 5. MLP Training and Testing Results (F20.DAT)

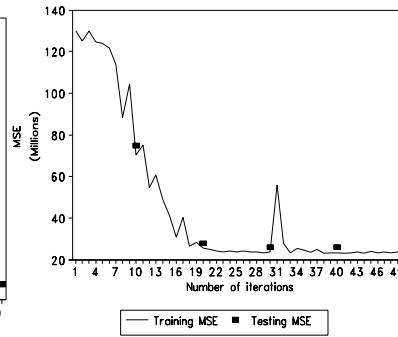


Fig. 6. MLP Training and Testing Results (F24.DAT)

V. CONCLUSIONS

In this paper, a method for calculating CRMAP bounds on neural network testing error variance has been described. The method has been successfully applied to the problem of flight load synthesis in helicopters. MLP neural nets have been trained that approximately attain optimal performance, as indicated by CRMAP bounds. Much work remains to be done. We need to size the inverse networks reliably, in order to produce better bounds. We also need to determine the bijectivity of mappings directly from the training data.

Acknowledgement

This work was funded by the Advanced Technology Program of the state of Texas under grant 003656-063.

VI. REFERENCES

- [1] H. L. Van Trees, *Detection, Estimation, and Modulation Theory -Part I*, New York, NY: John Wiley and Sons, 1968.
- [2] S.J. Apollo, M.T. Manry, L.S. Allen, and W.D. Lyle, "Optimality of Transforms for Parameter Estimation," *Conference Record of the Twenty-Sixth Annual Asilomar Conference on Signals, Systems, and Computers*, Oct. 1992, vol. 1, pp. 294-298.
- [3] M.T. Manry, S.J. Apollo, and Q. Yu, "Minimum Mean Square Estimation and Neural Networks," *Neurocomputing*, vol. 13, September 1996, pp. 59-74.
- [4] Y-H Pao, *Adaptive Pattern Recognition and Neural Networks*, Addison-Wesley Publishing Company, Inc., New York, 1989.
- [5] S. Chen, C.F.N. Cowan, and P.M. Grant, "Orthogonal Least Squares Learning Algorithm for Radial Basis Function Networks," *IEEE Trans. on Neural Networks*, vol. 2, no. 2, March 1991, pp. 302-309.
- [6] K. Hornik, M. Stinchcombe, and H. White, "Universal Approximation of an Unknown Mapping and its Derivatives Using Multilayer Feedforward Networks," *Neural Networks*, Vol. 3, 1990, pp. 551-560.
- [7] Y. Ito, "Approximations of Differentiable Functions and Their Derivatives on Compact Sets by Neural Networks," *Math. Scientist*, Vol. 18, 1993, pp. 11-19.
- [8] H-H.Chen, M.T. Manry and Hema Chandrasekaran , "A Neural Network Training Algorithm Utilizing Multiple sets of Linear Equations", to appear in *Neurocomputing*.
- [9] M.T. Manry, C-H Hsieh, M.S. Dawson, A.K. Fung, and S.J. Apollo, "Cramer Rao Maximum A-Posteriori Bounds on Neural Network training Error for Non-Gaussian Signals and parameters", *International Journal of Intelligent Control and Systems*, Vol. 1, No. 3, 1996, pp. 381-391.
- [10] C-H Hsieh, M. T. Manry, and H-H.Chen., "Cramer Rao Maximum A-Posteriori Bounds for a Finite Number of Non-Gaussian Parameters", *Conference Record of the 30th Asilomar Conference on Signals, Systems, and Computers*, vol. 2, Nov. 1996, pp.1161-1165.
- [11] W. Liang, M.T. Manry, Q. Yu, S.J. Apollo, M.S. Dawson, and A.K. Fung, "Bounding the Performance of Neural Network Estimators, Given Only a Set of Training Data," *Conference Record of the Twenty-Eighth Annual Asilomar Conference on Signals, Systems, and Computers*, vol. 2, Nov. 1994, pp.912-916.
- [12] Hema Chandrasekaran and Michael T. Manry, "Convergent Design of a Piecewise Linear Neural Network", to appear in *International Joint Conference on Neural Networks (IJCNN'99)*, July 1999.
- [13] S.Subbarayan, K.Kim, M.T.Manry, V.Devarajan and H.Chen, "Modular Neural Network Architecture using Piecewise Linear Mapping", *Thirtieth Asilomer Conference on Signals, Systems & Computers*, Vol. 2, pp 1171-1175, Nov 1996.