Sensor Failure Detection/Correction Using Autoassociative Memory

Dukki Chung

Dept. of Computer Engineering and Science
Case Western Reserve University
Cleveland, Ohio 44106
Phone: (216)368-8871
E-mail: dchung@alpha.ces.cwru.edu

Francis L. Merat

Associate Professor
Dept. of Electrical Engineering and Applied Physics
Case Western Reserve University
Cleveland, Ohio 44106
Phone: (216)368-4572
FAX: (216)368-2668
E-mail: flm@po.cwru.edu

Proceedings of the OAI Neural Network Symposium and Workshop
pp. 335-347, 1995

Sensor Failure Detection/Correction using Autoassociative Memory

Dukki Chung[1] and Francis L. Merat[2]

Department of Computer Engineering and Science[1]

Department of Electrical Engineering and Applied Physics[2]

Case Western Reserve University, Cleveland, OH 44106

Abstract

An autoassociative memory using neural networks is proposed for sensor failure detection and correction. Conventional systems for failure analysis and detection rely upon complex models of physical systems; however, neural networks can be used to represent systems through training for which mathematical models can not be formulated. A neural network autoassociative memory can be used to predict sensor outputs. Differences between measured sensor outputs and sensor outputs estimated by the autoassociative memory can be used to identify faulty sensors. The autoassociative memory can also be used to substitute correct sensor output values for faulty values. This technique can be successfully used to process data from dense sensor arrays.

I. Introduction

Sensor failure detection and correction is a very important subject for a variety of applications. If one or more sensors in a system fail or are failing, they should be identified and isolated. Whenever a sensor(s) is not functioning correctly, there should be a way to deal with and potentially correct for this inaccurate sensor data. In conventional approaches dealing with inaccurate sensor outputs, there must be an appropriate mathematical model of the system. However, it may not be possible to develop an accurate model for the system, or, it may not be possible to mathematically solve the model. In either case, conventional systems will either perform very poorly or simply be impossible to implement.

An autoassociative memory using neural networks is proposed for sensor failure detection and correction. Neural networks can learn the characteristics of unknown/unmodeled systems through training samples. By comparing the autoassociative memory output with the output of the sensors, it is easy to find out which sensor(s) is(are) faulty. In addition, the autoassociative memory can substitute correct sensor output values for faulty values filtering out faulty sensor outputs.

II. Autoassociative Memory Using Neural Networks

Neural networks are commonly used for classification and functional approximation. A neural network with one hidden layer with sufficient hidden nodes can be thought of as a universal function approximator [1, 2]. The network learns a mapping from the given inputs to the desired outputs through training samples. The neural network can learn autoassociation from training samples. The mapping is apparent, i.e. the network output should be equal to the network input at all times. The autoassociative memory is useful because it can correct noise, distortion and/or partial input values.

Several researches have used a three layer network (one hidden layer) to implement an autoassociative memory. Baldi and Hornik [3] showed that a three layer network is equivalent to PCA (Principal Component Analysis) projection. Later, Bourlard [4] showed that the three layer network is not superior to PCA. Recently, several researchers have shown that a five layer (three hidden layers) network can improve on Principal Component Analysis, as a non-linear Principal Component Analysis [5, 6, 7, 8].

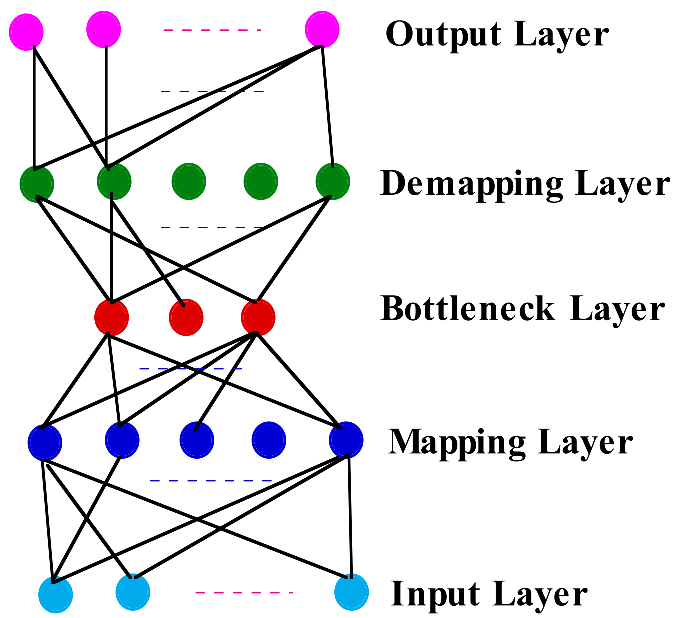

Figure 1. Three hidden layer autoassociative neural network

A three hidden layer autoassociative memory can be viewed as a serial combination of two single-hidden layer networks. The input, mapping, and bottleneck layers represent a nonlinear function or mapping, which projects the input data into a lower dimension feature space. The bottleneck layer, demapping layer, and the output layer represent a second network that remaps an approximation of the input from the feature space of the bottleneck layer output.

Since the bottleneck layer has fewer nodes than in the input and output layers, the net is constrained to develop a lower dimensional representation in the bottleneck layer. This three hidden layer autoassociative neural network works as a nonlinear compressor-decompressor pair[8]. This type of autoassociative memory is quite difficult to train, since lots of nonlinear nodes are involved in the training, and it often fails to converge to an acceptable training error. If large number of inputs are involved in the problem, it may not be a practical solution.

III. Autoassociative Memory Using Random Vector Enhanced Phasor Neural Networks

For implementing an autoassociative memory, a random vector enhanced phasor neural network (RV-PNN) was used [9]. In an RV-PNN autoassociative memory input patterns are enhanced by the multiplication of the input patterns and randomly chosen vectors. These random vector enhancements are further discussed in [10, 11].

Suppose that there are n original attributes (features, or elements) and j augmented attributes in the real domain pattern $\boldsymbol{x}$. For the augmented attribute x_i, the attribute is defined as

$$x_i = e_i \quad \text{for } i = n+1, \ldots, n+j \tag{1}$$

where $e_i = a_{i1}x_1 + \ldots + a_{in}x_n = \boldsymbol{a}_i^T\boldsymbol{x}$

$$\boldsymbol{a}_i = [\, a_{i1} \;\; a_{i2} \;\; \ldots \;\; a_{in} \,]^T$$

$$\boldsymbol{x} = [\, x_1 \;\; x_2 \;\; \ldots \;\; x_n]^T$$

and $\boldsymbol{a}_i$ is the random vector. The elements of the vector $\boldsymbol{a}_i$ are randomly generated on the real interval $[-\Omega, \Omega]$.

These augmented patterns are then transformed into complex vectors. One way to convert the pattern attributes (real numbers) to complex numbers is to assign a phase angle to each attribute and give it a unit magnitude. The resulting attributes are on the unit circle in the complex plane (phasors). A phase angle can be assigned by using the Z-score as

$$\theta = \frac{2\pi}{1 + e^{-\frac{x-\mu}{\sigma}}} \tag{2}$$

where μ and σ are, respectively, the mean and standard deviation of the input key vectors. Equation (2) converts unbounded real values into phase angles from 0 to 2π. For the complex number input patterns, complex number random vectors are generated over a magnitude range which avoids saturation of the transfer function. After the enhancement by equation (1), these patterns are fed into a sigmoidal transfer function like $tanh(real(x)) + i\,tanh(imag(x))$.

These transformed patterns then form the linear associative memory via the Moore-Penrose pseudoinverse. In linear associative memory model, patterns are linearly transformed by a relation of the form

$$\mathbf{y}_i = \mathbf{M}\mathbf{x}_i \qquad \text{for } i=1, \ldots, p \tag{3}$$

where $\mathbf{M}$ is the memory matrix, $(\boldsymbol{x}_i, \boldsymbol{y}_i)$ is the i-th associated pair of patterns, and p is the total number of associated patterns. In terms of the key vector matrix $\mathbf{X}$ and the recollection vector matrix $\mathbf{Y}$, the associative memory must satisfy $\mathbf{Y} = \mathbf{MX}$. Generally $\mathbf{X}$ is not square, and, thus, the Moore-Penrose pseudoinverse is used for the memory matrix $\mathbf{M}$ [12]

$$\mathbf{M} = \mathbf{YX}^{+}. \tag{4}$$

If such a pseudoinverse approach is not possible, gradient-descent learning can be used instead [9], as

$$\mathbf{M}^{new} = \mathbf{M}^{old} + \alpha(\mathbf{Y} - \mathbf{M}^{old}\mathbf{X})\mathbf{X}^{*} \tag{5}$$

where * is the conjugate transpose, and α is the learning rate (a small positive number). By equation (5), it is possible to construct the pseudoinverse based $\mathbf{M}$ matrix iteratively. Since the learning is linear, this approach is guaranteed to find the global minimum.

When dealing with large amounts of input data, the training time can be a critical issue for neural networks. Since training a three hidden layer net, or a single hidden layer net for large input nodes is not trivial, RV-PNNs can be used instead of the backpropagation neural networks to dramatically decrease the necessary training time.

IV. Sensor Failure Detection/Correction

Sensor failure detection and isolation is very important. If one or more sensors fail or are failing, they should be identified and isolated for safe operation. Because the sensors are the least reliable components in the control system and are subjected to harsh conditions, some form of redundancy is necessary to achieve adequate reliability in many control situations.

Hardware redundancy uses multiple sensors to measure variables. Voting schemes compare multiple output values from the sensors and can detect and isolate faulty sensors. Since multiple sensors are deployed, there can be a substantial increase in weight, cost, and space in a physically redundant system.

Analytical redundancy (AR) uses a reference model for the system and redundant information from dissimilar sensors to provide an estimate of measured variables. Analytic formulas must be developed which describe the system. Differences between measured sensor outputs and estimated sensor outputs, called residuals, are used to identify faulty sensors. This residual generation is typically based on knowledge of the system [13, 14, 15, 16]. If such knowledge is not available, analytical redundancy can not be used, or will perform very poorly.

A neural network approach is also a model based approach, but the model is learned through training. Neural networks can learn the characteristics of unknown/unmodeled systems.

Only input/desired output pairs are needed to train the neural networks. In the proposed method of sensor failure detection and correction, outputs from the sensors are initially processed by the RV-PNN autoassociative memory. The residuals are calculated by taking differences between the sensor outputs and the estimated sensor outputs from the autoassociative memory. As a second step, the pattern of the residuals can be processed further using methods like statistical decision theory to identify a particular failure. In addition, the autoassociative memory provides the correct values for the faulty sensor outputs.

V. Computer Simulation

For this simulation, a single hidden layer neural network was trained and tested using real world data from an optical sensor array which consisted of 32 sensors. There was no known analytical relationship between a given input and an estimated output for this sensor array making this sensor array ideal for neural net signal processing. The performance of a single hidden layer neural network function estimator is quite satisfactory (less than 0.4% average estimation error). This result is shown in Figures 2 and 3.

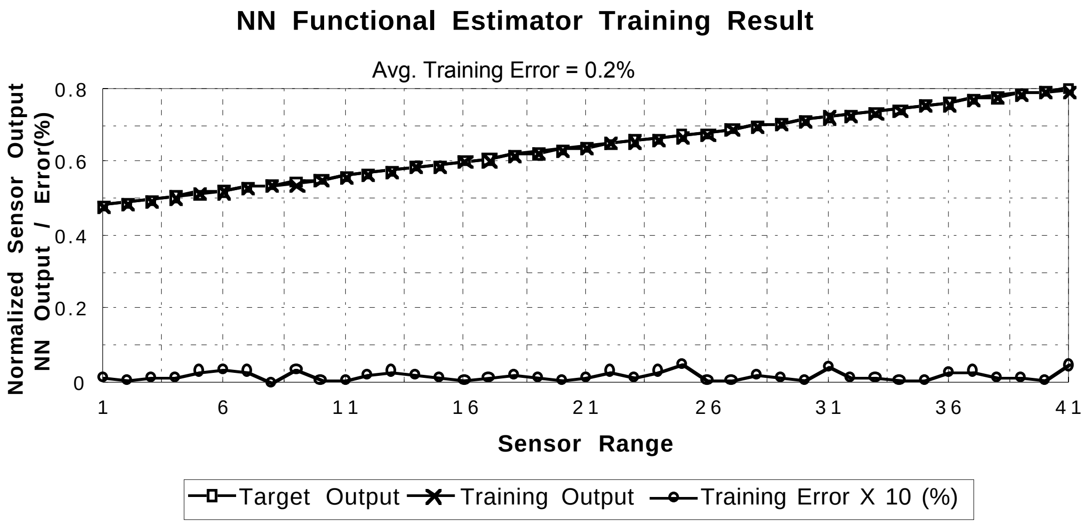

Figure 2. Training result of the neural network functional estimator

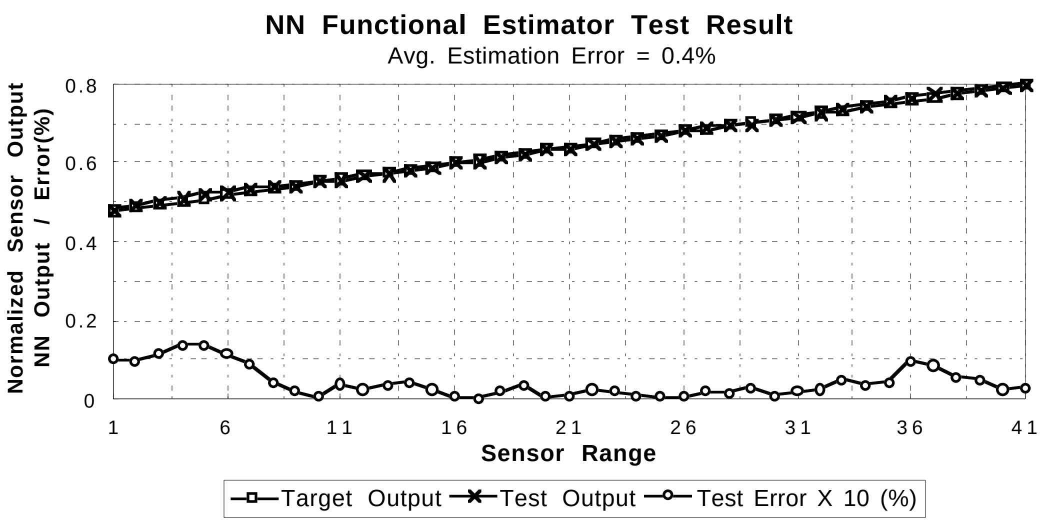

Figure 3. Test result of the neural network functional estimator

As a second step, two autoassociative memories (a three hidden layer autoassociative and an RV-PNN autoassociative memory) were implemented for the same sensor array and tested with the neural network function estimator. Figure 4 shows the method used to test the performance of the autoassociative memories. Gaussian random noise was added to the original data from the sensor array to simulate measurement noise. This noisy data was fed into the three hidden layer autoassociative memory and the RV-PNN autoassociative memory, and fed into the neural net functional estimator for comparison. When the sensor output does not contain any noise, these associative memories do not affect the function estimator as shown in Figure 5.

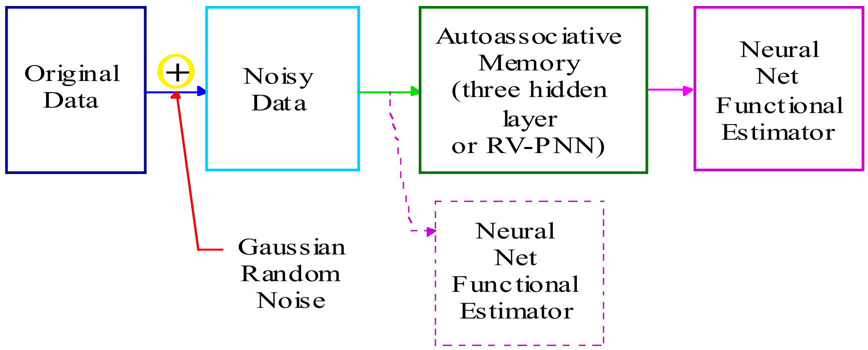

Figure 4. Test method

Table 1 shows the performance of the functional estimator with the RV-PNN autoassociative memory, with the three hidden layer neural autoassociative memory, and without any associative memory signal conditioning. The S/N ratio in Table 1 was calculated as $10log_{10}$(signal variance / noise variance). In this simulation, the functional estimator with the RV-PNN autoassociative memory and with the three hidden layer autoassociative memory performed equally well. They both corrected the faulty sensor values quite well in simulated noisy environments. The difference between the two autoassociative memory models is training time. The training time for the RV-PNN autoassociative memory is orders of magnitude faster than that for the three hidden layer autoassociative memory.

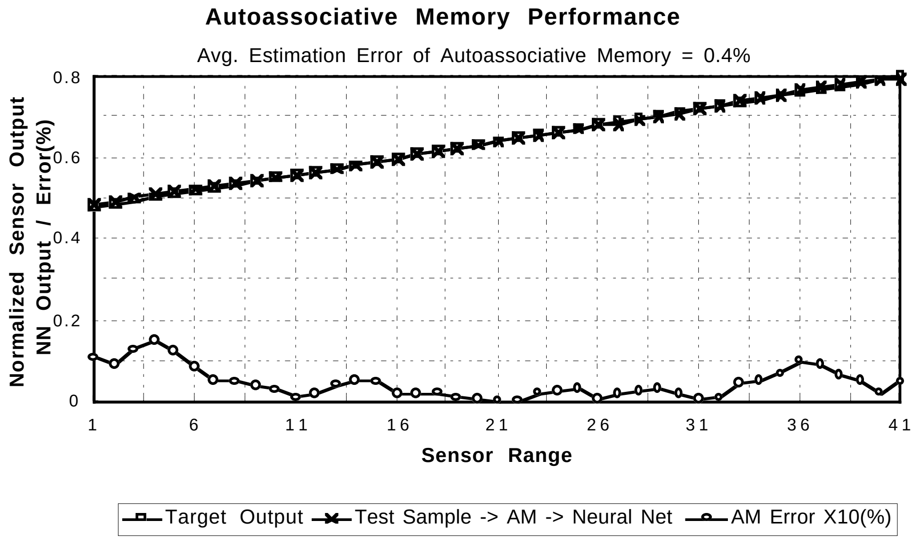

Figure 5. Autoassociative memory with the neural network function estimator

Test No.	S/N Ratio (dB)	RV-PNN AM		3 hidden layer AM		Plain NN Estimator	
		Avg. estimation error (%)	Std. dev of errors	Avg. estimation error (%)	Std. dev. of errors	Avg. estimation error (%)	Std. dev. of errors
1	9.03	1.1	0.99	1.1	0.81	6.6	6.12
2	9.07	1.3	1.51	1.5	2.91	7.0	5.39
3	8.93	1.0	0.81	1.2	0.93	6.6	5.21
4	9.17	1.0	0.66	1.1	1.04	7.5	6.16
5	9.10	1.2	1.04	1.0	0.91	6.7	4.71
6	3.49	2.2	2.40	2.3	2.44	13.4	9.90
7	3.63	1.9	1.52	2.6	2.10	13.2	10.0
8	3.36	2.1	2.32	2.2	3.22	11.0	7.30
9	3.54	2.4	2.45	3.1	4.52	12.8	10.05
10	3.63	2.6	2.82	2.5	2.52	10.6	9.52
Training time on HP 9000 712/60 workstation		2 minutes		120 minutes		5 minutes	

Table 1. Comparison of autoassociative memory models

Figure 6 shows the effect of the autoassociative memory on the individual sensor nodes for noisy outputs from the sensor array. The autoassociative memory treats these noisy sensor outputs as faulty sensor values. Without any autoassociative memory signal processing, the neural net functional estimator by itself has an estimation error of 25.5% for this case of noisy sensor outputs. Using the same data and autoassociative memory signal processing, the estimation error drops to 1.4 %. From Figure 6, it is clear that residuals can be generated by taking the differences between the raw sensor values and the outputs of the autoassociative memory. In addition, the autoassociative memory can provide correct values for the faulty sensor values.

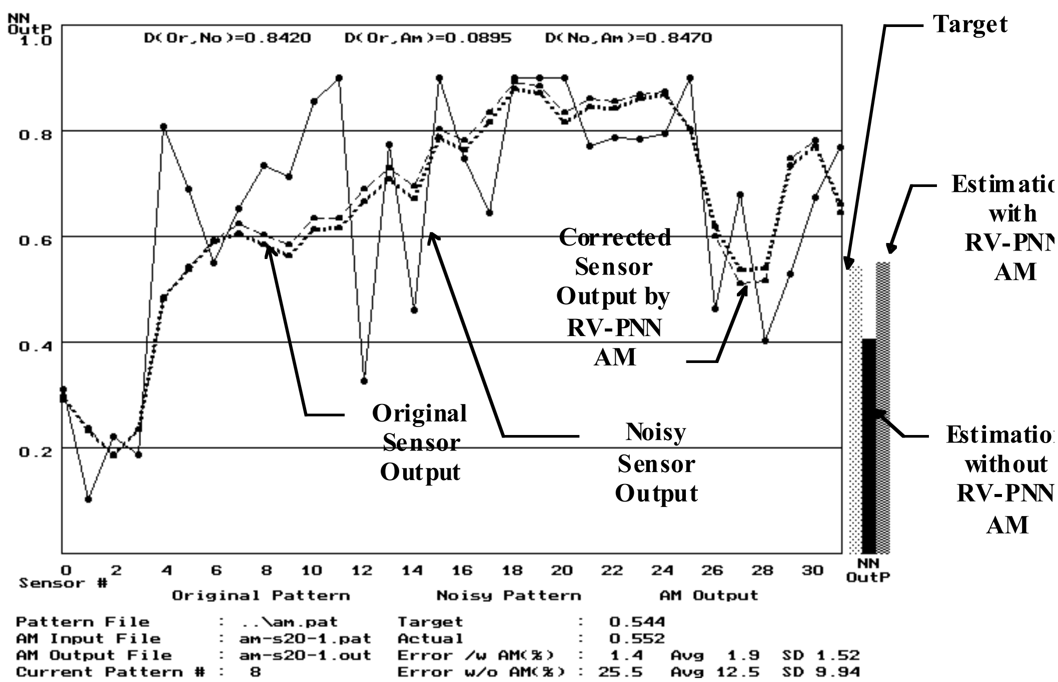

Figure 6. Effect of the autoassociative memory using RV-PNN
(Data S/N = 3.49dB)

VI. Conclusion

An autoassociative memory using neural networks is proposed for sensor failure detection and correction. The proposed autoassociative memory model can learn an unknown system via training. By comparing the autoassociative memory output with the output of the sensors, faulty sensors can be isolated. The autoassociative memory can also provide correct sensor output values for faulty values.

Experiments were conducted using an RV-PNN and a three hidden layer neural network associative memory. Both autoassociative memories resulted in very similar performance in simulated noisy environments. The RV-PNN autoassociative memory is preferable to the three hidden layer autoassociative memory because of its faster training time.

Based upon this work, it is proposed that these techniques can be successfully used to process data from dense sensor arrays for sensor failure detection and correction. Another similar application would be to use an autoassociative memory to filter out sensor-to-sensor variations (e.g. manufacturing tolerances) for mass-produced sensor arrays.

References

1. Hornik, K., M. Stinchombe, and H. White, "Multilayer Feedforward Networks are Universal Approximators," in *Neural Networks,* vol. 2, pp. 359-369, 1989.

2. Funahashi, K., "On the Approximation Realization of Continuous Mappings by Neural Networks," in *Neural Networks*, vol. 2, pp. 183-192, 1989.

3. Baldi, P. and K. Hornik, "Neural Networks and Principal Component Analysis: Learning form Examples without Local Minima,", in *Neural Networks*, vol. 2, pp. 53-58, 1989.

4. Bourlard, H. and Y. Kamp, "Auto-association by Multilayer Perceptrons and Singular Value Decomposition," in *Biol. Cyb.*, vol. 59, pp. 291-294, 1988.

5. Nampol, A., M. Arozullah, and S. Chin, “Higher Order Data Compression with Neural Networks,”, in *Proceedings of International Joint Conference on Neural Networks* (*IJCNN*), pp. I-55 - I59, 1991.

6. Kramer, M. A., “Nonlinear Principal Component Analysis using Autoassociative Neural Networks,” in *AIChE Journal*, vol. 37, pp. 233-243, 1991.

7. Oja, E., “Data Compression, Feature Extraction, and Autoassociation in feedforward neural networks,” in *Artificial Neural Networks*, pp. 737-745, Elsevier Science Publishers B. V., 1991.

8. Kramer, M. A., “Autoassociative Neural Networks,” in *AIChE Journal*, vol. 16, pp. 313-328, 1992.

9. Chung, D. and F. L. Merat, “Associative Memory Using Random Vector Enhanced Phasor Neural Network,” in *proceedings of the OAI Neural Network Symposium and Workshop*, pp. 267-278, 1995.

10. Pao, Y. H., G. H. Park and D. J. Sobajic, “Learning and Generalization Characteristics of the Random Vector Functional-Link Net,”, in *Neuro Computing*, vol. 6, pp. 163-180, 1994.

11. Igelnik, B. and Y. H. Pao, “Stochastic Choice of Basis Functions in Adaptive Function Approximation and the Functional-Link Net,” in *IEEE Transactions on Neural Networks*, 1994 (accepted for publication).

12. Penrose, R., “A Generalized Inverse for Matrices,” in *Proc. Cambridge Philos. Soc.* (England), LI, pp. 406-413, 1955.

13. Chow, E. and A. S. Willsky, “Analytical Redundancy and the Design of Robust Failure Detection Systems”, in *IEEE Transactions on Automatic Control,* vol. AC-29, no. 7, pp. 603-614, Jul 1984.

14. Merril, W. C., J. C. DeLaat, and M. Abdelwahab, “Turbo Engine Demonstration of Sensor Failure Detection,” in *J. Guidance*, vol. 14, no. 2, pp. 337-349, Mar-Apr 1991.

15. Piercy, N. P., “Sensor Failure Estimators for Detection Filters,” in *IEEE Transactions on Automatic Control,* vol. 37, no. 10, pp. 1553-1558, Oct 1992.

16. Zimmerman, D. C. and T.L. Lyde, “Sensor Failure Detection and Isolation in Flexible Structures Using System Realization Redundancy,” in *J. Guidance, Control and Dynamics,* vol. 6, no. 3, pp. 490-497, May-Jun 1993.