On H_∞ Filtering in Feedforward Neural Networks Training and Pruning

He-Sheng Tang[1], Song-Tao Xue[2], and Rong Chen[1]

[1] Research Institute of Structural Engineering and Disaster Reduction,
Tongji University, Shanghai, 200092, China
thstj@mail.tongji.edu.cn
[2] Department of Architecture, School of Science
and Engineering, Kinki University,
Kowakae3-4-1, Higashi Osaka City, 577-0056, Japan

Abstract. An efficient training and pruning method based on H∞ filtering algorithm is proposed for Feedforward neural networks (FNNs). A FNNs' weight importance measure linking up prediction error sensitivity obtained from H_∞ filtering training and a weight salience based pruning technique are derived. The results of extensive experimentation indicate that the proposed method provides better pruning results during the training process of the network without losing its generalization capacity, also provides a robust global optimization training algorithm for given arbitrary network structures.

1 Introduction

For neural networks (NNs) design, there are two crucial problems: one is the choice of a 'fast' and 'robust' training algorithm, and the other is the choice of a suitable or, ideally, minimal NNs topology to be adopted. In neural network training, the most well-known online training method is the backpropagation algorithm (BPA) [1], which is virtually a first-order stochastic gradient descent method and shows slow learning speed [2]. To overcome the slowness, many modified schemes based on the classical nonlinear programming technique have been suggested to speed up the training [3][4]. Recently, a class of second-order descent methods inspired by the theory of system identification and nonlinear filtering [5] has been introduced to estimate the weights of a neural network. Extended Kalman filter (EKF) [6][7][8] and Recursive least square (RLS) method have been applied to multilayer perceptron [9][10][11]. In the above mentioned EKF algorithm, although the learning speed is improved, the method requires the knowledge of the noise statistics. Convergence of this algorithm as well as the final values depends, to great extent, on this initial guess. The authors have presented suboptimal H_∞ filtering to train feedforward multilayer network which is independent on noise statistics [12].

Besides the training algorithms, another concern encountered in the practical application of the NNs is the choice of suitable model architecture. Since an unsuitable topology will increase the training time or even cause non-convergence, it usually decrease the generalization capability of the network. If there are too few weights, the network may not be trained to learn the training data for the system mapping. On the

J. Wang et al. (Eds.): ISNN 2006, LNCS 3971, pp. 515 – 523, 2006.

other hand, if the network size is too large, weights overfitting problems may usually occur and thus lead to worse generalization capacity[13][14]. Thus, in order to eliminate unnecessary weights, the pruning algorithm is applied. There are different pruning or model selection methods, such as Akaike Information Criterion (AIC) and cross-validation techniques [15][16] which require tens of networks to be exhaustively trained before the correct network size is determined, or simple weight decay method [17[18], or error sensitivity-based Optimal Brain Damage (OBD) [19] and Optimal Brain Surgeon (OBS) [20] methods, or OBD-like pruning methods [21][22], or growing methods [4] which may be sensitive to initial conditions and become trapped in local minima[23] .

As the H_∞ filtering was shown to be more efficient and robust than the Kalman filter[24][25], it would be interesting to inquire if there is any possibility of applying H_∞ filtering training method together with network pruning. The objective of the present study is to develop a FNNs training and pruning method based on H_∞ filtering algorithm for identification of nonlinear systems. The presented new method is able to reduce the complexity of the network during the training without diminishing the network's estimation capacity. Also, independent of the statistics of the disturbances of the network's inputs and outputs, the presented method provides a natural global optimization training algorithm for given arbitrary network structures. Examples of nonlinear system identification are given to verify the usefulness and effectiveness of the proposed method.

2 H_∞ Filtering Algorithm in Neural Network Training

Let $\mathbf{y}_k = \mathbf{f}(\mathbf{w}_k, \mathbf{u}_k)$ be the transfer function of a single-layer FNNs where $\mathbf{y}_k$ is the output, $\mathbf{u}_k$ is the input and $\mathbf{w}_k$ is its parameter vector that is combined by the weight matrices $\mathbf{w}^1$, $\mathbf{w}^2$ and $\mathbf{w}^3$. Given a set of training data, the training of a neural network can be formulated as a filtering problem [6][8]. In this case, a discrete-time FNNs' behavior can be described by the following nonlinear state-space model:

$$\mathbf{w}_{k+1} = \mathbf{w}_k + \mathbf{v}_k \tag{1}$$

$$\mathbf{y}_k = \mathbf{f}(\mathbf{w}_k, \mathbf{u}_k) + \mathbf{n}_k \tag{2}$$

Eq. (1) is known as process equation, where $\mathbf{v}_k$ is process noise, the state of system is given by the network's weight parameters values $\mathbf{w}_k$. Eq. (2) is the observation or measurement equation, represents the desired network response vector $\mathbf{y}_k$ as a nonlinear function $\mathbf{f}(\bullet)$ of the input vector $\mathbf{u}_k$ and the weight parameter vector $\mathbf{w}_k$; this equation is augmented by random measurement noise $\mathbf{n}_k$.

To apply the optimal H_∞ filtering algorithm, linear Taylor approximation of the $\mathbf{f}(\mathbf{w}_k, \mathbf{u}_k)$ at $\hat{\mathbf{w}}_{k-1}^-$ (prediction of $\mathbf{w}_k$), $\mathbf{u}_k$ is considered here, that is

$$\mathbf{f}(\mathbf{w}_k, \mathbf{u}_k) \approx \mathbf{f}(\hat{\mathbf{w}}_{k-1}^-, \mathbf{u}_k) + \mathbf{C}_k(\mathbf{w}_k - \hat{\mathbf{w}}_{k-1}^-) \tag{3}$$

where $\mathbf{C}_k = \partial \mathbf{f} / \partial \mathbf{w} \,|_{\mathbf{u}=\mathbf{u}_k, \mathbf{w}=\hat{\mathbf{w}}_{k-1}^-}$. A new quantity is introduced as follows:

$$\boldsymbol{\eta}_k = \mathbf{y}_k - \mathbf{f}(\hat{\mathbf{w}}_{k-1}^-, \mathbf{u}_k) + \mathbf{C}_k \hat{\mathbf{w}}_{k-1}^- \tag{4}$$

The entries in the term $\boldsymbol{\eta}_k$ are all known at time k, and, therefore, $\boldsymbol{\eta}_k$ can be regarded as an observation vector at time k. Hence, the nonlinear model (Eq. (2)) is approximated by the linear model

$$\boldsymbol{\eta}_k = \mathbf{C}_k \mathbf{w}_k + \mathbf{n}_k \tag{5}$$

The problem addressed by the H_∞ filtering is to find an estimate $\hat{\mathbf{w}}_k$ of $\mathbf{w}_k$ given $\mathbf{u}_j, \boldsymbol{\eta}_j (j = 0,1,\cdots k)$. The suboptimal H_∞ estimation [25][26] is interested not necessarily in the estimation of $\mathbf{w}_k$ but in the estimation of some arbitrary linear combination of $\mathbf{w}_k$ using the noise-corrupted observations $\boldsymbol{\eta}_j (j = 0,1,\cdots k)$ i.e.,

$$\mathbf{z}_k = \mathbf{L}_k \mathbf{w}_k \tag{6}$$

where $\mathbf{L}_k \in R^{q \times n}$. Different from that of the modified Wiener/Kalman filter which minimizes the variance of the estimation error, the design criterion of the H_∞ filter is to provide a uniformly small estimation error, $\mathbf{z}_k - \hat{\mathbf{z}}_k$, for any $\mathbf{n}_k \in l_2$ and $\mathbf{w}_0 \in R^n$. The H_∞ filtering will search $\hat{\mathbf{z}}_k$ such that the optimal estimate of $\mathbf{z}_k$ among all possible $\hat{\mathbf{z}}_k$ in the sense that the supremum of the performance measure should be less than a positive pre-chosen noise attenuation factor γ^2, i.e., the worse-case performance measure

$$\sup_{\mathbf{x}_0, \{\mathbf{n}_k, \mathbf{v}_k\}} \frac{\sum_{k=0}^{N-1} \left\| \mathbf{z}_k - \hat{\mathbf{z}}_k \right\|^2}{\left\| \mathbf{w}_0 - \hat{\mathbf{w}}_0 \right\|_{\mathbf{P}_0^{-1}}^2 + \sum_{k=0}^{N-1} \left\{ \left\| \mathbf{n}_k \right\|^2 + \left\| \mathbf{v}_k \right\|^2 \right\}} < \gamma^2 \tag{7}$$

where $\hat{\mathbf{w}}_0$ is an a priori estimate of $\mathbf{w}_0$ and $\mathbf{w}_0 - \hat{\mathbf{w}}_0$ represents unknown initial condition error, $\mathbf{P}_0^{-1} > 0$ is weighting matrix. $\mathbf{P}_0^{-1} > 0$ denotes a positive definite matrix that reflects a priori knowledge on how the initial guess $\hat{\mathbf{w}}_0$ close to $\mathbf{w}_0$ is.

Let $\gamma > 0$ be a prescribed level of noise attenuation. If this is the case, an optimized H_∞ filtering algorithm for neural network training can be derived:

$$\begin{aligned} \hat{\mathbf{w}}_k &= \hat{\mathbf{w}}_{k-1}^- + \mathbf{K}_k (\boldsymbol{\eta}_k - \mathbf{C}_k \hat{\mathbf{w}}_{k-1}^-) \\ &= \hat{\mathbf{w}}_{k-1}^- + \mathbf{K}_k (\mathbf{y}_k - \mathbf{f}(\hat{\mathbf{w}}_{k-1}^-, \mathbf{u}_k)), \ \hat{\mathbf{w}}_k^- = \hat{\mathbf{w}}_k, \ \hat{\mathbf{w}}_{-1}^- = \hat{\mathbf{w}}_0 \end{aligned} \tag{8}$$

where $\hat{\mathbf{w}}_k$ is an a posteriori estimate of the state at step k, the gain $\mathbf{K}_k$ of the filter is given by

$$\mathbf{K}_k = \mathbf{P}_k \mathbf{C}_k^T (\mathbf{I} + \mathbf{C}_k \mathbf{P}_k \mathbf{C}_k^T)^{-1} \tag{9}$$

$$\mathbf{P}_{k+1} = \mathbf{P}_k (\mathbf{I} + \mathbf{C}_k^T \mathbf{C}_k \mathbf{P}_k - \gamma^{-2} \mathbf{L}_k^T \mathbf{L}_k \mathbf{P}_k)^{-1} \tag{10}$$

where the attenuation factor γ must be tuned so as to satisfy the $\mathbf{P}_k$ positive definite.

3 H_∞ Filtering in Neural Network Pruning

In this section, the conjunction of network training and pruning with the H_∞ filtering algorithm will be illustrated. Without loss of generality, the network employed here is considered as a feedforward architecture with n_I input units, n_H hidden sigmoid unites and a single linear output unit. The initial network is fully connected between layers and implements a nonlinear mapping from input space $\mathbf{u}_k$ to target output space $\hat{y}_k = f_k(\mathbf{u}_k, \mathbf{w}_k)$, where $f(\bullet)$ is the actual output mapping function, $\mathbf{w}$ is the network parameters and $\hat{y}_k$ is the prediction of the target output y_k. Then, for a given training set, the cost function can be expressed as:

$$\mathrm{E}(\mathbf{w}) = \frac{1}{2N}\sum_{k=1}^{N}(y_k - f_k)^2 \tag{11}$$

where N is the number of training examples.

Under the assumption that the network is fully trained, that is, the cost function E has adjusted to a local or global minimum on the error surface, the second derivative of E with respect to $\mathbf{w}$ or the Hessian matrix [19] can be approximated as:

$$\mathbf{H}(N) \approx \frac{1}{N}\sum_{k=1}^{N}\left(\frac{\partial f_k}{\partial \mathbf{w}}\right)\left(\frac{\partial f_k}{\partial \mathbf{w}}\right)^T \tag{12}$$

To illustrate the connection between the matrix $\mathbf{P}_k$ and the Hessian matrix $\mathbf{H}$ of the cost function, the Riccati Eq. (10) can be rewritten in an alternative form that will be more convenient for analysis. By employing the following matrix inversion lemma (MIL), the following update for $\mathbf{P}_k^{-1}$ is obtained

$$\mathbf{P}_{k+1}^{-1} = \mathbf{P}_k^{-1} + \mathbf{C}_k^T\mathbf{C}_k - \gamma^{-2}\mathbf{L}_k^T\mathbf{L}_k \tag{13}$$

Suppose that the weight parameter and the 'error covariance matrix' $\mathbf{P}_k$ are both converge. Without loss of generality, it is convenient to select the matrix $\mathbf{L}_k$ equal to $\mathbf{C}_k$. The Riccati recursion Eq. (13) with initial condition $\mathbf{P}_0$ can be rewritten in the form of a recursion as:

$$\mathbf{P}_{k+1}^{-1} = \mathbf{P}_0^{-1} + (1-\gamma^{-2})\sum_{i=0}^{k}\mathbf{C}_i^T\mathbf{C}_i \tag{14}$$

From Eq. (12) and Eq. (14), the inversion of Hessian matrix of the cost function is approximately expressed as:

$$\mathbf{H}^{-1} \approx N(1-\gamma^{-2})\mathbf{P}_{k+1}\left[\mathbf{I} - (\mathbf{P}_{k+1} - \mathbf{P}_0)^{-1}\mathbf{P}_{k+1}\right] \tag{15}$$

The pruning procedure simplifies the computations by making a further assumption of the Hessian matrix $\mathbf{H}$ being a diagonal matrix [19]. Thus the saliencies for each parameter are as follows:

$$S_i = \frac{1}{2}[\mathbf{H}]_{i,i}\mathbf{w}_{[i]}^2 \tag{16}$$

where $[\mathbf{H}]_{i,i}$ is the i-th diagonal element of the $\mathbf{H}$ and $\mathbf{w}_{[i]}$ is the i-th weight.

The pruning strategy is to find the low-saliency or smallest saliency parameters which are then selected for deletion. The weights are pruned according to this pruning order until the change in the estimated training error is greater than the tolerance limit.

4 Illustrative Numerical Examples

A non-linear hysteretic system [27] is selected for example analysis to illustrate the applicability of the proposed method. The non-linear system is described by the following second order difference equation:

$$\frac{d^2u(t)}{dt^2} + z(\frac{du(t)}{dt}, u(t), z(t)) = p(t) \tag{17}$$

$$\frac{dz(t)}{dt} = 24.5\frac{du(t)}{dt} - 2\left|\frac{du(t)}{dt}\right|z(t)|^2 z(t) - 0.5\frac{du(t)}{dt}|z(t)|^3 \tag{18}$$

where $u(t)$ and $p(t)$ are system's output and input, respectively.

The FNNs is taken into account to model the dynamic behaviors of the system, namely, the FNNs is trained to identify $z(t)$. An FNNs with 3 input neurons ($du(t)/dt$, $u(t)$ and $z(t-1)$), 30 hidden neurons (with hyperbolic tangent activation function), and one output neuron ($z(t)$,with linear activation function) is trained to capture the unknown system. In this case, the total number of weights is 120.

In this study, the effectiveness of the noise injection training is also investigated; noises are artificially added to the training data. The noise level is defined as the value of the standard deviation. For instance, if the standard deviation is 0.05, the noise level in the data can be referred as 5% in the root-mean-square (RMS) level.

Training and test samples for the non-linear system identification problem are shown in Fig.1. The training set contains 1000 samples with 3% noise in RMS level and test set contains 500 samples. Numbers of training iterations of some parts of estimated network's weights are shown in Fig.2. Some weights converge to stable values very quickly in first 500 iterations. This fast convergence demonstrated that the H_∞ filtering is a very fast training algorithm. Last step values of the $\mathbf{P}_k$ (120×120 matrix, Fig. 3) of the Riccati recursion show that the matrix $\mathbf{P}_k$ is almost diagonal dominant. Fig.4 shows the estimated training error against the number of weights in the pruned network. It is clear that nearly only 70 weights are enough to capture the unknown system without increasing the training error dramatically. After training, another 500 pairs of testing data are passed to the pruned network. Fig. 5 depicts a segment of the actual and the desired network output for the test set. It shows that output of the pruned network is quite close to the desired output.

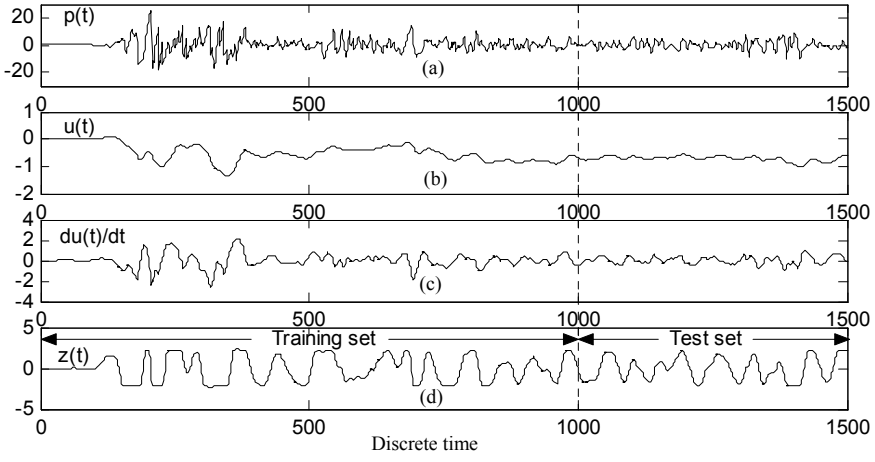

Fig. 1. Training and test samples for the system identification problem

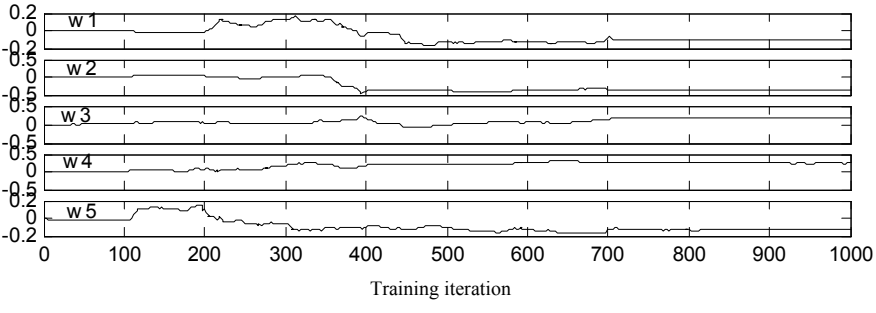

Fig. 2. Convergence of the estimate weights

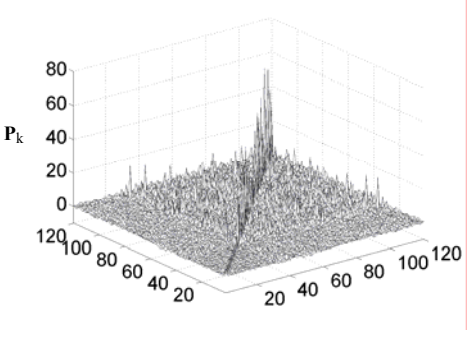

Fig. 3. Values of the $\mathbf{P}_k$ after trained H_∞-network

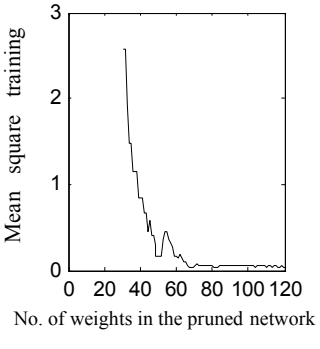

Fig. 4. Number of weights in the pruned network versus training errors

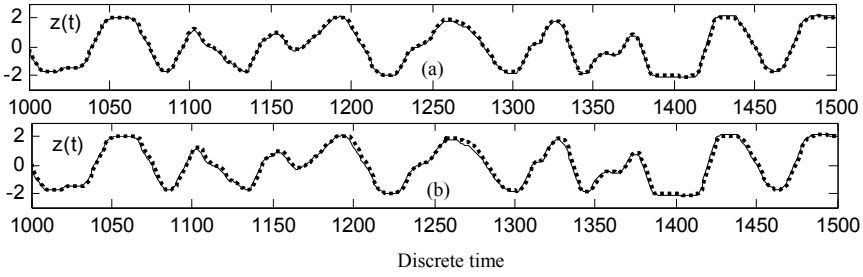

Fig. 5. A segment of the actual and the desired network output for the test set in the system identification example. Solid line represents the desired output; the dotted line corresponds to the actual output of: (a) the unpruned network; (b) the network pruned by our approach

To demonstrate that the generalization capability is not affected much if some weights in the network are pruned, another set of input signal (the test set) is fed for the pruned network. A system input $p(t)$ (Fig.6 (a)) is selected for the validation set. According to this input, corresponding desired output for this validation set are obtained. Fig. 6(b) depicts a segment of the actual and the desired network output corresponding to the test input. The instantaneous error is shown in Fig. 6 (c). The figure shows that the generalization capability of the pruned networks does not change much.

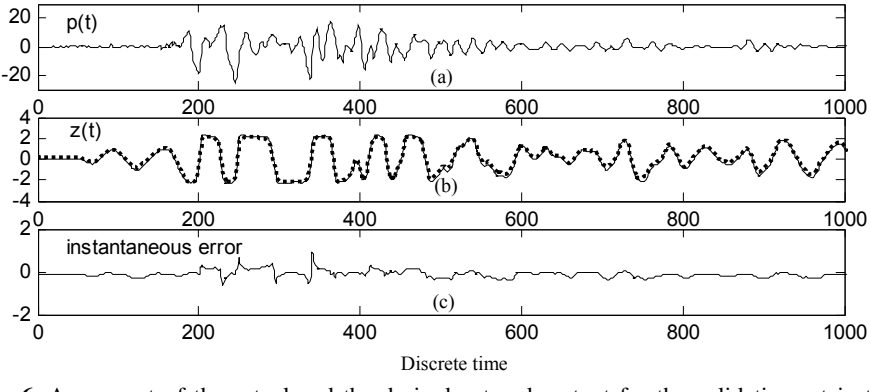

Fig. 6. A segment of the actual and the desired network output for the validation set in the system identification example. (a) Validation input of system; (b) comparison between actual output (solid line) and the pruned H_∞-network output (dashed line); (c) instantaneous error

5 Conclusions

In this paper, we have derived an efficient FNNs training and pruning method using the H_∞ filtering algorithm. FNNs model pruned with H_∞ filtering provides a good architecture design for generalization capacity and requires a smaller number of connection weights than a totally connected net. Examples of nonlinear system identifica-

tion are carried out to evaluate the performance of the network. The results show the effectiveness of the neural network pruning and training by H_∞ filtering algorithm.

References

1. Rumelhart, D., Hinton, G., and Williams, G.: Learning Internal Representations by Error Propagation. Parallel Distributed Processing, Vol.1, Cambridge, MA: MIT Press (1986) 318-362
2. Robbins, H. and Monro, S.: A Stochastic Approximation Method. Ann. Math. Stat. 22 (1951) 400-407
3. Bojarczak, O. S. P., and Stodolski, M.: Fast Second-order Learning Algorithm for Feedforward Multilayer Neural Networks and Its Application. Neural Networks 9(9) (1996) 1583–1596
4. Haykin, S.: Neural Networks: A Comprehensive Foundation, Prentice Hall, Inc. (1999)
5. Anderson, B. D. O., and Moore, J. B.: Optimal Filtering. Englewood Cliffs, NJ: Prentice-Hall (1979)
6. Singhal, S., and Wu, L.: Training Multiplayer Perceptrons with the Extended Kalman Algorithm. In: Touretzky,D.S.(eds.): Advances in Neural Information Processing Systems, Vol.1, Morgan Kaufmann, San Mateo, CA (1989) 133-140
7. Fukuda, W. K. T., and Tzafestas, S. G.: Learning Algorithms of Layered Neural Networks via Extended Kalman Filters. Int. J. Syst. Sci. 22(4) (1991) 753–768
8. Iiguni, Y., Sakai, H., and Tokumaru, H.: A Real-time Learning Algorithm for a Multilayered Neural Network Based on the Extended Kalman Filter. IEEE Trans. Signal Processing 40(4) (1992) 959–966
9. Chen, S., Cowan, C. F. N., Billings, S. A., and Grant, P. M.: Parallel Recursive Prediction Error Algorithm for Training Layered Neural Network. Int. J. Contr. 51(6) (1990) 1215–1228
10. Kollias, S., and Anastassiou, D.: An Adaptive Least Squares Algorithm for the Efficient Training of Artificial Neural Networks. IEEE Trans. Circuits Syst. 36(8) (1989) 1092–1101
11. Leung, C. S., Wong, K. W., Sum, J., and Chan., L. W.: On-line Training and Pruning for RLS Algorithms. Electron. Lett. 32(23) (1996) 2152–2153
12. Tang, H., and Sato, T.: Structural Damage Detection Using the Neural Network and H_∞ Algorithm. In: Kundu, T. (eds.): Health Monitoring and Smart Nondestructive Evaluation of Structural and Biological Systems III, Proceedings of SPIE, Vol. 5394, San Diego, CA (2004) 454-463
13. Abrahart, R.J., See, L., and Kneal, P.E.: Investigating the Role of Saliency Analysis with Neural Network Rainfall-runoff Model. Computer & Geosciences 27(8) (2001) 921-928
14. Makarynskyy, O., Pires-Silva, A.A., Makarynska, D., and Ventura-Soares, C.: Artificial Neural Networks in Wave Predictions at the West Coast of Portual. Computers & Geosciences 31(4) (2005) 415-424
15. Akaike, H.: A New Look at the Statistical Model Identification. IEEE Trans. Automat. Control AC-19 (6) (1974) 716-723
16. Stone, M.: An Asymptotic Equivalence of Choice of Model by Cross-validation and Akaike's criterion. Roy. Stat. Soc. Ser. 39(1) (1977) 44-47
17. Krogh, A., and Hertz, J.: A Simple Weight Decay Can Improve Generalization. In: Touretzky, D.S. (ed.): Advances in Neural Information-Processing Systems, Vol.4, Morgan Kaufmann, San Mateo, CA (1992) 950–957

18. William, P.M.: Bayesian Regularization and Pruning Use a Laplace Prior, Neural Comput. 7(1) (1995) 117– 143
19. LeCun, Y., Denker, J. S., and Solla, S. A.: Optimal Brain Damage. In: Touretzky, D. S. (ed.): Advances in neural information processing, Vol.1, Morgan Kaufman, San Mateo, CA (1990) 396–404
20. Hassibi, B., and Stork, D. G.: Second-order Derivatives for Network Pruning: Optimal Brain Surgeon. In: Hanson, S. J., Cowan, J. D., and Lee Giles, C. (eds.): Advances in neural information processing, Vol.4, Morgan Kaufman, CA (1993) 164-171
21. Sum, J., Leung, C., Young, G. H., and Kan, W.: On the Kalman Filtering Method in Neural-Network Training and Pruning. IEEE Trans. on Neural Networks 10(1) (1999)161-166
22. Leung, C., Wong, K., Sum, P., and Chan, L., A Pruning Method for the Recursive Least Squared Algorithm. Neural Networks 14(2) (2001) 147-174
23. Prechelt, L.: A quantitative Study of Experimental Evaluations of Neural Network Learning Algorithms: Current Research Practice. Neural Networks 9 (3) (1996)457-462
24. Sato, T., and Qi, K.: Adaptive H_∞ fillter: Its Application to Structural Identification. Journal of Engineering Mechanics 124(11) (1998) 1233-1240
25. Hassibi, B., Sayed, A.H., and Kailath, T.: Indefinite-Quadratic Estimation and Control: A Unified Approach to H_2 and H_∞Theories, SIAM (1998).
26. Didinsky, G., Pan, Z., and Basar, T.: Parameter Identification of Uncertain Plants Using H_∞ Methods. Automatica 31(9) (1995) 1227-1250
27. Wen, Yi-K.: Method for Random Vibration of Hysteretic Systems. Journal of Engineering Mechanics 102(EM2) (1976) 249-263