H_∞ Filtering in Neural Network Training and Pruning with Application to System Identification

He-Sheng Tang[1]; Songtao Xue[2]; and Tadanobu Sato[3]

Abstract: An efficient training and pruning methodology based on the H_∞ filtering algorithm is proposed for artificial neural networks (ANNs). ANNs are first trained by the H_∞ filtering algorithm and then some unimportant weights are removed based on the training. The results presented in the paper show that the proposed method provides better pruning results of the network without losing its generalization capacity. It also provides a robust training algorithm for given arbitrary network structures. The usefulness and effectiveness of the proposed methodology are demonstrated in developing an ANN model of a hysteretic structural system.

DOI: 10.1061/(ASCE)0887-3801(2007)21:1(47)

CE Database subject headings: Training; Neural networks; Algorithms.

Downloaded from ascelibrary.org by Tongji University on 12/08/18. Copyright ASCE. For personal use only; all rights reserved.

Introduction

It has been recognized that artificial neural networks (ANNs) offer a number of potential benefits for application in the field of nonparametric system identification, particularly for modeling nonlinear systems (Masri et al. 1996; Nakamura et al. 1998; Chen et al. 1995; Masri et al. 2000; Wu et al. 2002). One concern in neural networks is the choice of a "fast" and "robust" training algorithm for a given problem. In neural network training, the most used online training method is the back-propagation algorithm (BPA) (Rumelhart et al. 1986; Abid et al. 2001). However, it is virtually a first-order stochastic gradient descent method (Jaakkola et al. 1994), and hence, its learning speed could be very slow. Many modified schemes based on the classical nonlinear programming technique have been proposed to speed up the training (Bojarczak and Stodolski 1996; Haykin 1999). Adopting from the idea in identification theory (Haykin 1996), a class of second-order descent methods such as the recursive least-squared (RLS) (Chen et al. 1990; Kollias and Anastassiou 1989; Leung et al. 1996) and extended Kalman filtering (EKF) (Singhal and Wu 1989; Fukuda and Tzafestas 1991; Shah et al. 1992; Iiguni et al. 1992) algorithms, has been introduced to estimate the weights of a neural network. The RLS algorithm is an effective online training method for neural networks. However, one main drawback of the existing RLS algorithm is its stability. RLS may not be able to run for a large number of samples. Although the EKF algorithm is somewhat efficient for the network training, it should be noted that the EKF method requires knowledge of the noise source statistics. The convergence of this algorithm as well as the final values depends, to a great extent, on this initial guess, which is unrealistic in modeling identification. In an earlier work, the writers have presented suboptimal H_∞ filtering to train feedforward multilayer networks, which is independent of noise statistics (Tang and Sato 2004). Despite so many techniques, a further improvement is highly desirable as regards learning accuracy, computational complexity, numerical stability, and generalization capability.

Apart from the training algorithm, another concern in neural networks is the size of a neural network for a given problem. If the size is too small, the network may not be trained to solve the given problem. On the other hand, if the size is too large, overfitting occurs and also the resource is wasted (Abrahart et al. 2001; Makarynskyy et al. 2005). Thus, in order to eliminate unnecessary weights, the pruning algorithm has been applied. There are different pruning or model selection methods, such as the Akaike information criterion (AIC) and cross-validation techniques (Akaike 1974; Stone 1977), which require tens of networks to be exhaustively trained before the correct network size is determined, or the simple weight decay method (Krogh and Hertz 1992; William 1995; Bebis et al. 1997), or error sensitivity based optimal brain damage (OBD) (LeCun et al. 1990) and optimal brain surgeon (OBS) (Hassibi and Stork 1993) methods, or OBD-like pruning methods (Sum et al. 1999; Leung et al. 2001), or growing methods (Haykin 1999) which may be sensitive to initial conditions and become trapped in local minima (Prechelt 1996). In the OBD and OBS approaches, the estimation is based on the Hessian matrix of the training error. However, the Hessian matrix is usually unavailable in online mode operation since the training patterns are not held after training.

As the H_∞ filtering was shown to be more efficient and robust than the Kalman filter (Sato and Qi 1998; Hassibi et al. 1998), it is useful to see the possibility of applying the H_∞ filtering training method together with network pruning. The objective of the present study is to develop an ANN training and pruning methodology based on the H_∞ filtering algorithm for identification of nonlinear systems. It will be shown that the proposed methodology can reduce the complexity of the network during the training

[1]Lecturer, Research Institute of Structural Engineering and Disaster Reduction, Tongji Univ., Shanghai 200092, China. E-mail: thstj@mail.tongji.edu.cn

[2]Associate Professor, Dept. of Architecture, School of Science and Engineering, Kinki Univ., Kowakae 3-4-1, Higashi Osaka City 577-0056, Japan.

[3]Professor, Disaster Prevention Research Institute, Kyoto Univ, Uji, Kyoto 611-0011, Japan.

Note. Discussion open until June 1, 2007. Separate discussions must be submitted for individual papers. To extend the closing date by one month, a written request must be filed with the ASCE Managing Editor. The manuscript for this paper was submitted for review and possible publication on December 27, 2004; approved on May 23, 2006. This paper is part of the ***Journal of Computing in Civil Engineering***, Vol. 21, No. 1, January 1, 2007. ©ASCE, ISSN 0887-3801/2007/1-47–58/$25.00.

without diminishing the network's estimation capacity. Moreover, independent of the statistics of the disturbances of the network's inputs and outputs, the proposed method provides a robust 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 methodology.

H_∞ Filtering Algorithm in Neural Network Training

H_∞ Filtering Algorithm

H_∞ filtering (Didinsky et al. 1995; Hassibi et al. 1998; Sato and Qi 1998) is a state estimation problem of minimizing the maximum energy in the estimation error over all the disturbance trajectories and without any assumptions on the statistics or distributions of the disturbance signals. The worst-case performance is a useful formulation when the unknown signals contain deterministic components, or when they are random but with unknown statistics.

The design of discrete H_∞ filtering is discussed by Hassibi et al. (1998), and is explained as the following linear discrete system:

$$\mathbf{x}_{k+1}=\mathbf{A}_k\mathbf{x}_k+\mathbf{B}_k\mathbf{v}_k \tag{1}$$

$$\mathbf{y}_k=\mathbf{C}_k\mathbf{x}_k+\mathbf{n}_k \tag{2}$$

where $\mathbf{x}_k$=state vector of the system at the time step k; $\mathbf{y}_k$=vector of measurement at time step k; $\mathbf{v}_k$=process noise; and $\mathbf{n}_k$=measurement noise. There is no assumption on the nature of the unknown quantities $\mathbf{n}_k$ and $\mathbf{v}_k$, and $(\mathbf{A}_k,\mathbf{B}_k,\mathbf{C}_k)$ are known system matrices.

The suboptimal H_∞ estimation problem is interesting not necessarily in the estimation of $\mathbf{x}_k$ but in the estimation of some arbitrary linear combination of $\mathbf{x}_k$ using the noise-corrupted observations $\{\mathbf{y}_k, k=0,1,2,N-1\}$, i.e.

$$\mathbf{z}_k=\mathbf{L}_k\mathbf{x}_k \tag{3}$$

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 $\mathbf{n}_k$, $\mathbf{v}_k\in l_2$ and $\mathbf{x}_0\in R^n$. Let the estimation performance measure be

$$J=\frac{\sum_{k=0}^{N-1}\|\mathbf{z}_k-\hat{\mathbf{z}}_k\|^2_{\mathbf{Q}_k}}{\|\mathbf{x}_0-\hat{\mathbf{x}}_0\|^2_{\mathbf{P}_0^{-1}}+\sum_{k=0}^{N-1}\left\{\left\|\mathbf{v}_k\right\|^2_{\mathbf{W}_k^{-1}}+\left\|\mathbf{n}_k\right\|^2_{\mathbf{V}_k^{-1}}\right\}} \tag{4}$$

where $[(\mathbf{x}_0-\hat{\mathbf{x}}_0),\mathbf{w}_k,\mathbf{v}_k]\neq 0$; $\hat{\mathbf{x}}_0$=a priori estimate of $\mathbf{x}_0$; $\mathbf{x}_0-\hat{\mathbf{x}}_0$ represents the unknown initial condition error; $\mathbf{P}_0^{-1}>0$, $\mathbf{Q}_k\geq 0$, $\mathbf{W}_k>0$, and $\mathbf{V}_k>0$=weighting matrices. $\mathbf{P}_0^{-1}>0$ denotes a positive definite matrix that reflects a priori knowledge on how the initial guess $\hat{\mathbf{x}}_0$ close to $\mathbf{x}_0$ is. The notation $\|\mathbf{z}_k\|^2_{\mathbf{Q}}$ is defined as the square of the weighted (by $\mathbf{Q}$) l_2 norm of $\mathbf{z}_k$, i.e., $\|\mathbf{z}_k\|^2_{\mathbf{Q}}=\mathbf{z}_k^T\mathbf{Q}\mathbf{z}_k$. The H_∞ filter 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 prechosen noise attenuation factor γ^2, i.e., the worse case performance measure

$$\sup_{\mathbf{x}_0,\{\mathbf{n}_k\},\{\mathbf{v}_k\}} J<\gamma^2 \tag{5}$$

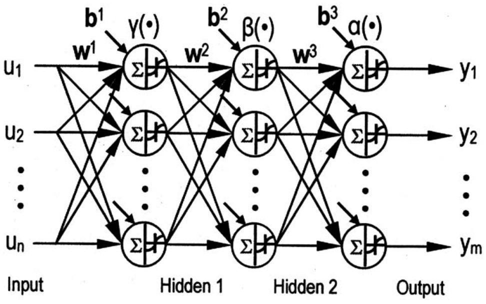

Fig. 1. Feedforward neural network model

Theorem (Hassibi et al. 1998)
Let $\gamma>0$ be a prescribed level of noise attenuation. Then, there exists an H_∞ filtering for $\mathbf{x}_k$ if and only if there exists a stabilizing symmetric solution $\mathbf{P}_k>0$ to the following discrete-time Riccati-type equation:

$$\mathbf{P}_{k+1}=\mathbf{A}_k\mathbf{P}_k(\mathbf{I}_n+\mathbf{C}_k^T\mathbf{V}_k^{-1}\mathbf{C}_k\mathbf{P}_k-\gamma^{-2}\bar{\mathbf{Q}}_k\mathbf{P}_k)^{-1}\mathbf{A}_k^T+\mathbf{B}_k\mathbf{W}_k\mathbf{B}_k^T,\ \mathbf{P}_0=\mathbf{p}_0 \tag{6}$$

where $\bar{\mathbf{Q}}_k=\mathbf{L}_k^T\mathbf{Q}_k\mathbf{L}_k$. If this is the case, then an H_∞ filtering can be given by

$$\hat{\mathbf{z}}_k=\mathbf{L}_k\hat{\mathbf{x}}_k\quad \hat{\mathbf{x}}_k^-=\mathbf{A}_k\hat{\mathbf{x}}_k \tag{7}$$

$$\hat{\mathbf{x}}_k=\hat{\mathbf{x}}_{k-1}^-+\mathbf{K}_k(\mathbf{y}_k-\mathbf{C}_k\hat{\mathbf{x}}_{k-1}^-),\quad \hat{\mathbf{x}}_{-1}^-=\hat{\mathbf{x}}_0=\text{initial guess} \tag{8}$$

where $\hat{\mathbf{x}}_k$=a posteriori estimate of the state at the step k; $\hat{\mathbf{x}}_k^-$=prediction of $\mathbf{x}_k$; and $\mathbf{K}_k$=gain of the filter given by

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

The above formulation shows that H_∞-optimal estimators guarantee the smallest estimation error energy over all possible disturbances with finite energy. Consequently, they are overly conservative and will result in better robust behavior to disturbance variations.

H_∞ Filtering in Network Training

In this study, a feedforward neural network (FNN) is adapted (Fig. 1). Fig. 1 shows a typical three-layer FNN: the input layer u_i $(i=1,\ldots,n)$ with n nodes, the two hidden layers with p and q nodes, and the output layer y_i $(i=1,\ldots,m)$ with m nodes. Between layers, there are weights, w_{ij}^1, w_{ij}^2, and w_{ij}^3, representing the strength of connections of the nodes in the network. In Fig. 1 we can assign a different activation function of $\gamma(\bullet)$, $\beta(\bullet)$, and $\alpha(\bullet)$ with corresponding bias terms, $\mathbf{b}^1$, $\mathbf{b}^2$, and $\mathbf{b}^3$, for each layer. In this paper, the hyperbolic tangent function

$$f(x)=\frac{1-e^{-\alpha x}}{1+e^{-\alpha x}}\quad(\alpha>0) \tag{10}$$

is chosen as the activation function.

The output vector $\mathbf{y}$ is calculated by feeding the input vector $\mathbf{u}$ through the hidden layer of the FNN, which is given as

$$y=\alpha\{\mathbf{w}^1\beta[\mathbf{w}^2\gamma(\mathbf{w}^3\mathbf{u}+\mathbf{b}^3)+\mathbf{b}^2]+\mathbf{b}^1\} \tag{11}$$

with $\mathbf{w}^1=[w_{ij}^1]$; $\mathbf{w}^2=[w_{ij}^2]$; $\mathbf{w}^3=[w_{ij}^3]$; $\mathbf{b}^1=\{b_1^1,\ldots,b_p^1\}^T$; $\mathbf{b}^2=\{b_1^2,\ldots,b_q^2\}^T$; $\mathbf{b}^3=\{b_1^3,\ldots,b_m^3\}^T$; and $\mathbf{u}=\{u_1,\ldots,u_n\}^T$. Without loss of generality, the bias can be omitted in this network in order to reduce network complexity (Haddadnia et al. 2002).

Downloaded from ascelibrary.org by Tongji University on 12/08/18. Copyright ASCE. For personal use only; all rights reserved.

Downloaded from ascelibrary.org by Tongji University on 12/08/18. Copyright ASCE. For personal use only; all rights reserved.

Let $\mathbf{y}_k=\mathbf{f}(\mathbf{w}_k,\mathbf{u}_k)$ be the transfer function of a single-layer FNN, where $\mathbf{y}_k$=output; $\mathbf{u}_k$=input; and $\mathbf{w}_k$=parameter vector. Given a set of training data, the training of a neural network can be formulated as a filtering problem (Singhal and Wu 1989; Iiguni et al. 1992). In this case, a discrete-time FNN's behavior can be described by the following nonlinear state-space model without process noise (here, the prescribed $\mathbf{x}_k$ will be replaced by $\mathbf{w}_k$):

$$\mathbf{w}_{k+1}=\mathbf{w}_k \tag{12}$$

$$\mathbf{y}_k=\mathbf{f}(\mathbf{w}_k,\mathbf{u}_k)+\mathbf{n}_k \tag{13}$$

Eq. (12) is known as the process equation, where the state of the system is given by the network's weight parameter values $\mathbf{w}_k$. Eq. (13) is the observation or measurement equation; it 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 the random measurement noise $\mathbf{n}_k$.

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

$$\mathbf{f}(\mathbf{w}_k,\mathbf{u}_k)\approx\mathbf{f}(\hat{\mathbf{w}}_{k-1}^-,\mathbf{u}_k)+\mathbf{C}_k(\mathbf{w}_k-\hat{\mathbf{w}}_{k-1}^-) \tag{14}$$

where

$$\mathbf{C}_k=\left.\frac{\partial\mathbf{f}}{\partial\mathbf{w}}\right|_{\mathbf{u}=\mathbf{u}_k,\mathbf{w}=\hat{\mathbf{w}}_{k-1}^-} \tag{15}$$

A new quantity is introduced as follows:

$$\boldsymbol{\eta}_k=\mathbf{y}_k-\mathbf{f}(\hat{\mathbf{w}}_{k-1}^-,\mathbf{u}_k)+\mathbf{C}_k\hat{\mathbf{w}}_{k-1}^- \tag{16}$$

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. (13)] is approximated by the linear model

$$\boldsymbol{\eta}_k=\mathbf{C}_k\mathbf{w}_k+\mathbf{n}_k \tag{17}$$

The problem addressed by H_∞ filtering is to find an estimate $\hat{\mathbf{w}}_k$ of $\mathbf{w}_k$ with given values of $\mathbf{u}_j,\boldsymbol{\eta}_j$ $(j=0,1,\ldots,k)$. The full algorithm is then obtained by substituting the linearized mapping functions into the H_∞ filtering recursion Eqs. (6)–(9). The process equation, Eq. (12), is linear; thus matrix $\mathbf{A}_k$ is an identity matrix. Suppose that weighting matrices $\mathbf{Q}_k$, $\mathbf{W}_k$, and $\mathbf{V}_k$ are all identity matrices. Then an optimized H_∞ filtering algorithm for neural network training can be derived

$$\begin{aligned}\hat{\mathbf{w}}_k&=\hat{\mathbf{w}}_{k-1}^-+\mathbf{K}_k(\boldsymbol{\eta}_k-\mathbf{C}_k\hat{\mathbf{w}}_{k-1}^-)\\&=\hat{\mathbf{w}}_{k-1}^-+\mathbf{K}_k[\mathbf{y}_k-\mathbf{f}(\hat{\mathbf{w}}_{k-1}^-,\mathbf{u}_k)+\mathbf{C}_k\hat{\mathbf{w}}_{k-1}^--\mathbf{C}_k\hat{\mathbf{w}}_{k-1}^-]\\&=\hat{\mathbf{w}}_{k-1}^-+\mathbf{K}_k[\mathbf{y}_k-\mathbf{f}(\hat{\mathbf{w}}_{k-1}^-,\mathbf{u}_k)],\quad\hat{\mathbf{w}}_{k-1}^-=\mathbf{A}_k\hat{\mathbf{w}}_k\end{aligned} \tag{18}$$

$$\mathbf{K}_k=\mathbf{P}_k(\mathbf{I}+\mathbf{C}_k^T\mathbf{C}_k\mathbf{P}_k)^{-1}\mathbf{C}_k^T \tag{19}$$

$$\mathbf{P}_{k+1}=\mathbf{P}_k(\mathbf{I}+\mathbf{C}_k^T\mathbf{C}_k\mathbf{P}_k-\gamma^{-2}\mathbf{L}_k^T\mathbf{L}_k\mathbf{P}_k)^{-1} \tag{20}$$

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

H_∞ Filtering in Neural Network Pruning

In this section, the conjunction of network training and pruning with the H_∞ filtering algorithm will be illustrated. To present the new methodology, the background of the Hessian-based network pruning approaches (Haykin 1999; LeCun et al. 1990) is first introduced. The basic idea of this approach to network pruning is to use information on second-order derivatives of the error surface in order to make a trade-off between network complexity and training-error performance.

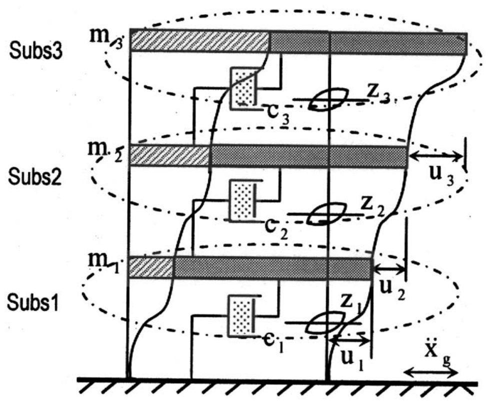

Fig. 2. Model of three degrees of freedom structural system

Without loss of generality, the network employed here is considered as a feedforward architecture with n_I input units, n_H hidden sigmoid units, 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 $f(\bullet)$=actual output mapping function; $\mathbf{w}$=network parameters; and $\hat{y}_k$=prediction of the target output y_k. Then, for a given training set, the cost function can be expressed as

$$E(\mathbf{w})=\frac{1}{2N}\sum_{k=1}^{N}(y_k-f_k)^2 \tag{21}$$

where N=number of training examples.

Under the assumption that the network is fully trained, i.e., 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 can be approximated as

$$\mathbf{H}(N)\approx\frac{1}{N}\sum_{k=1}^{N}\left(\frac{\partial f_k}{\partial\mathbf{w}}\right)\left(\frac{\partial f_k}{\partial\mathbf{w}}\right)^T \tag{22}$$

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

$$S_i=\frac{1}{2}[\mathbf{H}]_{i,i}\mathbf{w}_{[i]}^2 \tag{23}$$

where $[\mathbf{H}]_{i,i}$=ith diagonal element of the $\mathbf{H}$; and $\mathbf{w}_{[i]}$= ith weight.

However, the assumption of diagonal dominant of matrix $\mathbf{H}$ is not made in the OBS procedure so that the OBS recalculates the magnitude of all the weights in the network, and saliencies for each parameter are given by

$$S_i=\frac{\mathbf{w}_{[i]}^2}{2[\mathbf{H}^{-1}]_{i,i}} \tag{24}$$

where H^{-1}=inverse of the Hessian matrix $\mathbf{H}$; and $[\mathbf{H}^{-1}]_{i,i}$=ith diagonal element of the inverse matrix.

The pruning strategy is to find the low-saliency or smallest saliency parameters, which are then selected for deletion. After training, two kinds of information are obtained: (1) the parametric vector $\hat{\mathbf{w}}_k$, and (2) the recursion matrix $\mathbf{P}_k$. For simplicity, the

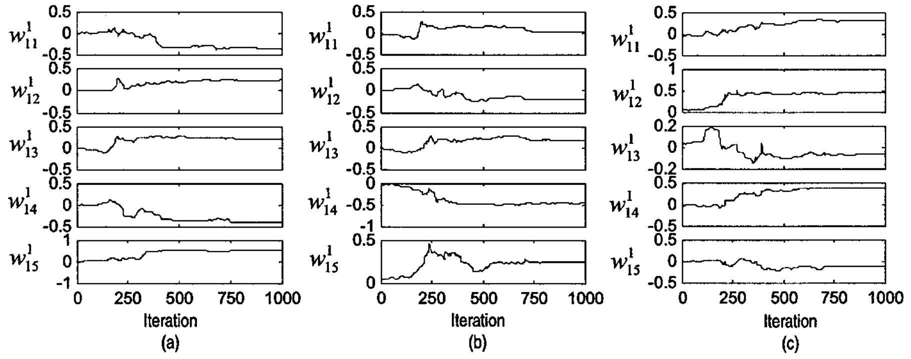

Fig. 3. Time history of network weights: (a) some parts of Subs 1; (b) some parts of Subs 2; and (c) some parts of Subs 3

Riccati Eq. (20) can be rewritten in an alternative form that is more convenient for analysis. By employing the following matrix inversion lemma (MIL):

$$\mathbf{A} - \mathbf{AB}(\mathbf{C} + \mathbf{B}^T\mathbf{AB})^{-1}\mathbf{B}^T\mathbf{A} = (\mathbf{A}^{-1} + \mathbf{BC}^{-1}\mathbf{B}^T)^{-1} \tag{25}$$

the following update for $\mathbf{P}_k^{-1}$ is obtained

$$\mathbf{P}_{k+1}^{-1} = \mathbf{P}_k^{-1} + \mathbf{C}_k^T\mathbf{C}_k - \gamma^{-2}\mathbf{L}_k^T\mathbf{L}_k \tag{26}$$

Suppose that the weight parameter and the "error covariance matrix" $\mathbf{P}_k$ are both convergent. Without loss of generality, it is convenient to select matrix $\mathbf{L}_k$ equal to $\mathbf{C}_k$ and then to readily establish the asymptotic behavior for the matrix

$$\mathbf{P}_{k+1}^{-1} = \mathbf{P}_k^{-1} + (1-\gamma^{-2})\mathbf{C}_k^T\mathbf{C}_k \tag{27}$$

To illustrate the connection between the matrix $\mathbf{P}_k$ and the Hessian matrix H of the cost function, the Riccati recursion Eq. (27) with initial condition $\mathbf{P}_0$ can be rewritten in the form of a recursion as

$$\mathbf{P}_{k+1}^{-1} = \mathbf{P}_0^{-1} + (1-\gamma^{-2})\sum_{i=0}^{k}\mathbf{C}_i^T\mathbf{C}_i \tag{28}$$

From Eqs. (15), (22), and (28), the Hessian matrix of the cost function is approximately expressed as

$$\mathbf{H} \approx \frac{\mathbf{P}_{k+1}^{-1} - \mathbf{P}_0^{-1}}{N(1-\gamma^{-2})} \tag{29}$$

and the inversion is

$$\mathbf{H}^{-1} \approx N(1-\gamma^{-2})\mathbf{P}_{k+1}[\mathbf{I} - (\mathbf{P}_{k+1} - \mathbf{P}_0)^{-1}\mathbf{P}_{k+1}] \tag{30}$$

Hence, the by-product $\mathbf{P}_k$ can be used to measure the increase in the training error E due to the changes in the weight vector $\mathbf{w}$.

In the rest of this section, we will first discuss how to use the matrix H to measure the saliencies of weights and then present the pruning approach.

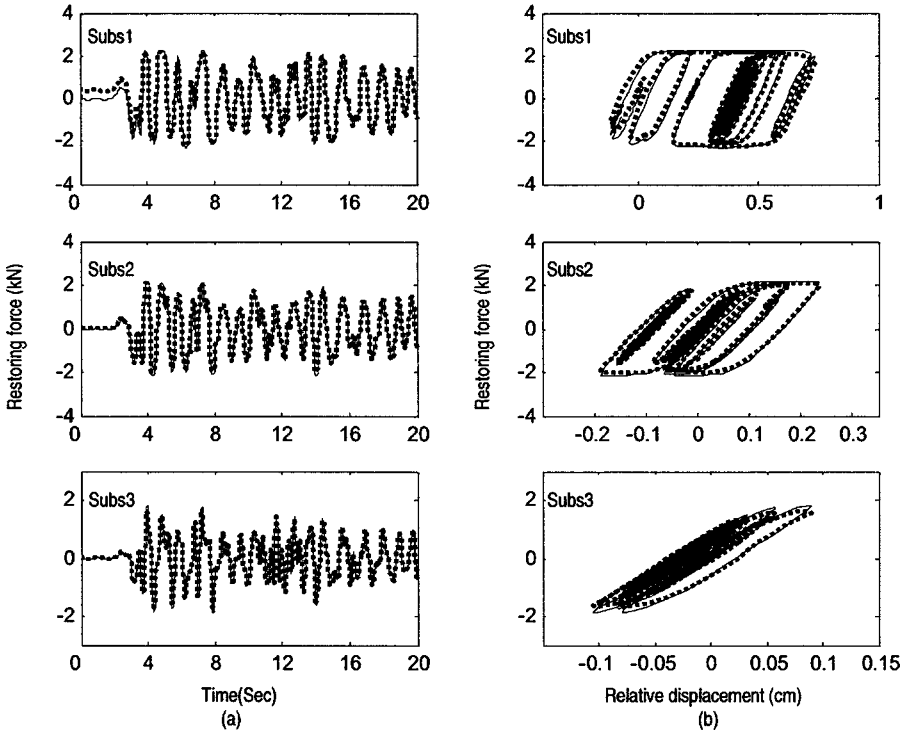

Fig. 4. Comparison between actual output (solid line) and H_∞-network output (dashed line) of the online training (with 0% noise injection in training data): (a) time history of restoring forces; (b) restoring forces versus relative displacements

Downloaded from ascelibrary.org by Tongji University on 12/08/18. Copyright ASCE. For personal use only; all rights reserved.

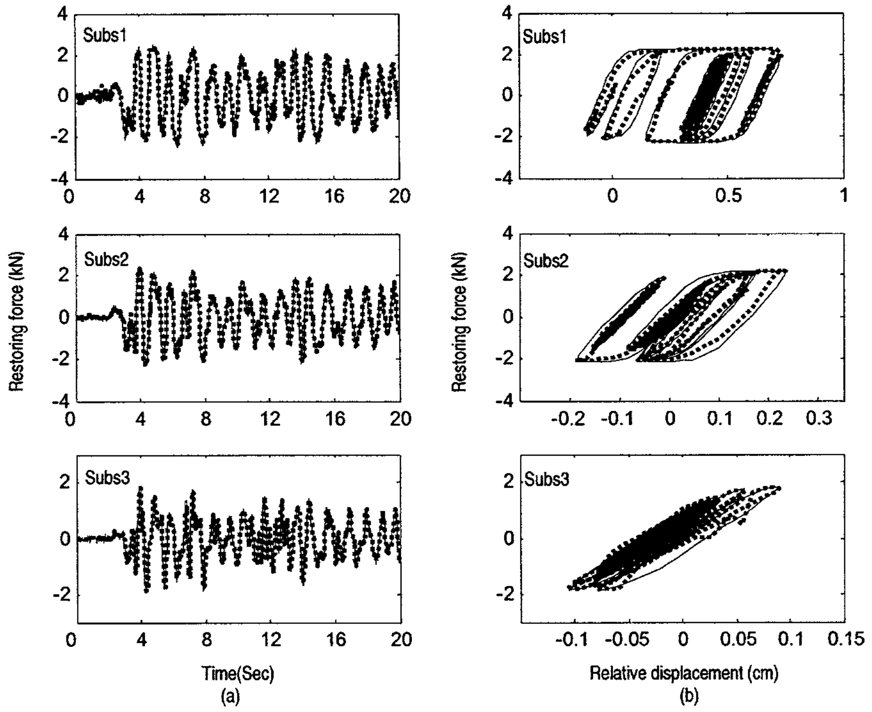

Fig. 5. Comparison between actual output (solid line) and H_∞-network output (dashed line) of the online training (with 5% noise injection in training data): (a) time history of restoring forces; (b) restoring forces versus relative displacements

From the classical OBD and OBS pruning approaches, removing the weight $\hat{\mathbf{w}}_{[i]}$ (the ith element of the weight vector) means to set $\hat{\mathbf{w}}_{[i]}$ to zero. Thus, a fast heuristic pruning algorithm (OBD-like) for the FNN based on the by-product $\mathbf{P}_k$ of the weight estimation of the proposed H_∞-training algorithm is presented. It is described as follows:

1. Set the initial values $\mathbf{P}_0$ and $\hat{\mathbf{w}}_0$;
2. Train the given network with the proposed H_∞-training approach to acceptable minimum error;
3. Calculate the estimated weight vector $\hat{\mathbf{w}}$, the estimated training error E and the by-product $\mathbf{P}_k$ after training;
4. Evaluate the ith saliency of weight

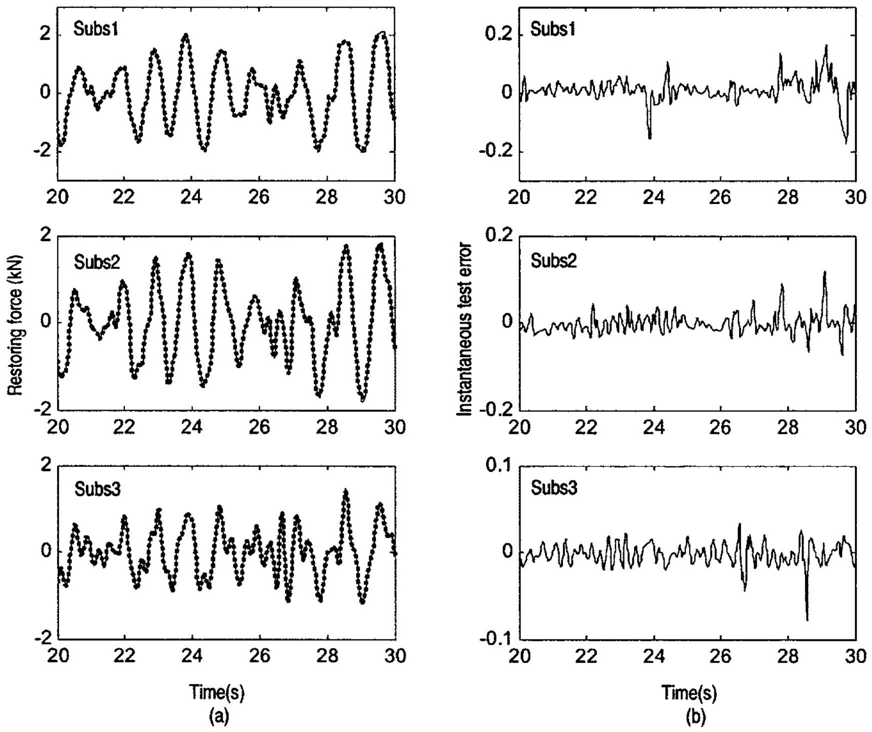

Fig. 6. Comparison between a segment of the actual output (solid line) and trained H_∞-network (with 0% noise injection in training data) output (dashed line) for the test set: (a) time history of restoring forces; (b) instantaneous test errors

Downloaded from ascelibrary.org by Tongji University on 12/08/18. Copyright ASCE. For personal use only; all rights reserved.

Downloaded from ascelibrary.org by Tongji University on 12/08/18. Copyright ASCE. For personal use only; all rights reserved.

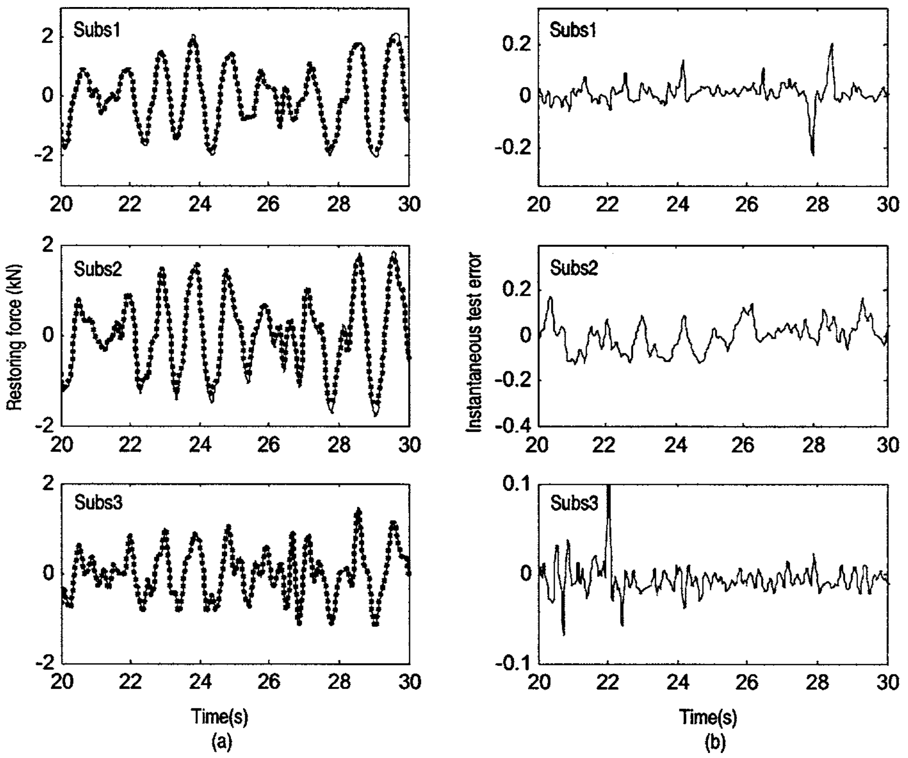

Fig. 7. Comparison between a segment of the actual output (solid line) and trained H_∞-network (with 5% noise injection in training data) output (dashed line) for the test set: (a) time history of restoring forces; (b) instantaneous test errors

$$S_i = \frac{1}{2}[\mathbf{H}]_{i,i}\hat{\mathbf{w}}_{[i]}^2 \tag{31}$$

5. Define the tolerance level in the estimated training error as λE $(0<\lambda<1)$. If the saliency $S_i \leq \lambda E$, then delete the synaptic weight $\hat{\mathbf{w}}_{[i]}$, and return to Step 4; otherwise, go to Step 6; and
6. Stop the computation when no more weights can be deleted from the network.

It should be noticed that the above pruning approach is suitable for the online situation, where the training set is not available after training. This is because the pruning approach does not use the training set. However, the traditional Hessian-based pruning approach uses the training set to calculate the Hessian matrix. Hence, the Hessian-based pruning approach is not suitable for the online situation. The example simulation in the next section shows that $\mathbf{P}_k$ of the Riccati recursion is diagonally dominant. Therefore, the pruning procedure can be further simplified by neglecting the effect of the off-diagonal elements of $\mathbf{P}_k$ in the calculation of S_i.

Illustrative Numerical Examples

Problem Statement

A hysteretic structural system is selected for the example analysis to illustrate the applicability of the proposed methodology. One of the most widely used models, the Bouc–Wen model (Wen 1976), is studied because it can capture many commonly observed types of hysteretic behavior. The reduced-order motion equation for the ith $(i=1,\ldots,n)$ active degree of freedom (DOF) for an n DOF shear-type structure (Fig. 2) subjected to earthquake-induced ground excitations $(\ddot{x}_g)$ can be written as

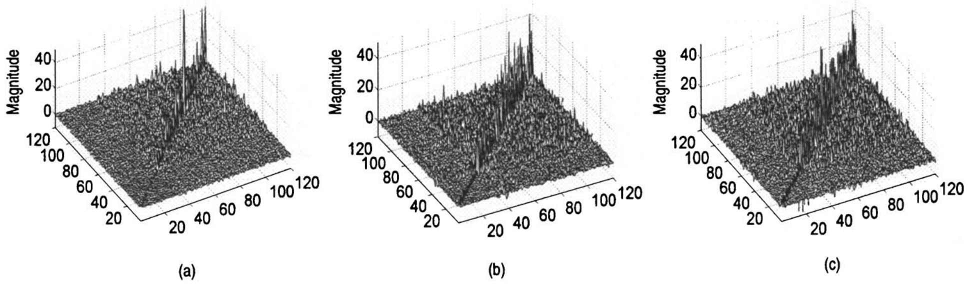

Fig. 8. Values of the $\mathbf{P}_k$ after training H_∞-network: (a) network for Subs 1; (b) network for Subs 2; and (c) network for Subs 3

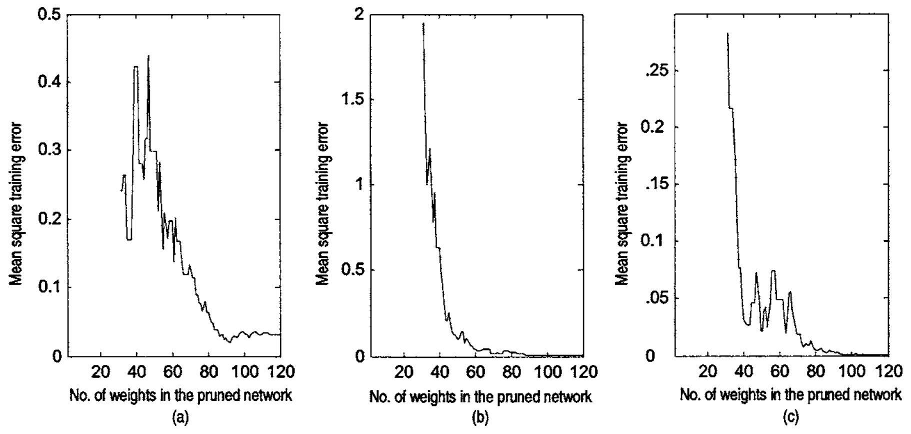

Fig. 9. Number of weights in the pruned network versus the training error (using El Centro responses with 5% noise injection in training data): (a) network for Subs 1 identification; (b) network for Subs 2 identification; and (c) network for Subs 3 identification

$$m_i\ddot{x}_i + r_i - (1-\delta_{in})r_{i+1} = -m_i\ddot{x}_g \tag{32}$$

where $\delta_{in}=0$ $(i \neq n)$ or $\delta_{in}=1$ $(i=n)$; and m_i, $\ddot{x}_i$, and r_i=ith mass, relative acceleration, and restoring force, respectively.

The ith component of the interstory restoring force vector is expressed by

$$r_i(\dot{u}_i, u_i, r_i) = c_i\dot{u}_i + z_i \tag{33}$$

and z_i is satisfied by

$$\dot{z}_i = k_i\dot{u}_i - \alpha_i|\dot{u}_i||z_i|^{n_i-1}z_i - \beta_i\dot{u}_i|z_i|^{n_i} \tag{34}$$

where $\dot{u}_i$=relative velocity between the $(i-1)$th and ith mass point; c_i=damping; k_i=stiffness; and α_i, β_i, and n_i=nonlinear parameters, respectively.

Assume that the mass is known and the experimental measurements for $\ddot{x}_i$ $(i=1,\ldots,n)$ and $\ddot{x}_g$ are available and that the corresponding $\dot{x}_i$ and x_i can be found by direct measurements or through integration of $\ddot{x}_i$. Hence, the restoring up force values r_i

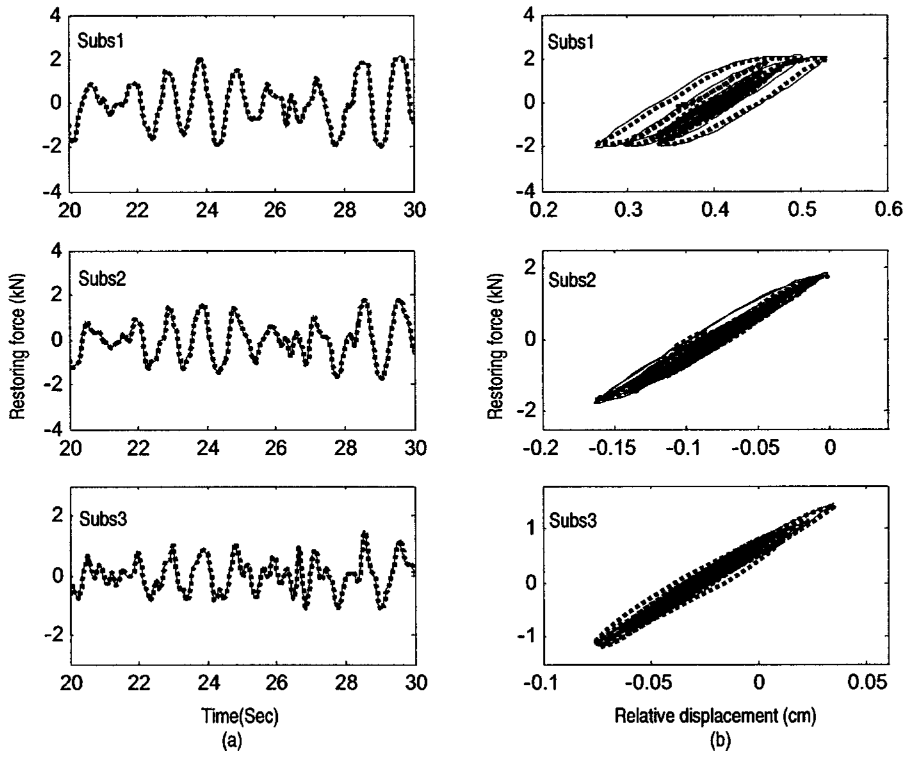

Fig. 10. Comparison between a segment of the actual output (solid line) and pruned H_∞-network (using El Centro responses with 5% noise injection in training data) output (dashed line) for the test set (using El Centro responses): (a) time history of restoring forces; (b) restoring forces versus relative displacements

Downloaded from ascelibrary.org by Tongji University on 12/08/18. Copyright ASCE. For personal use only; all rights reserved.

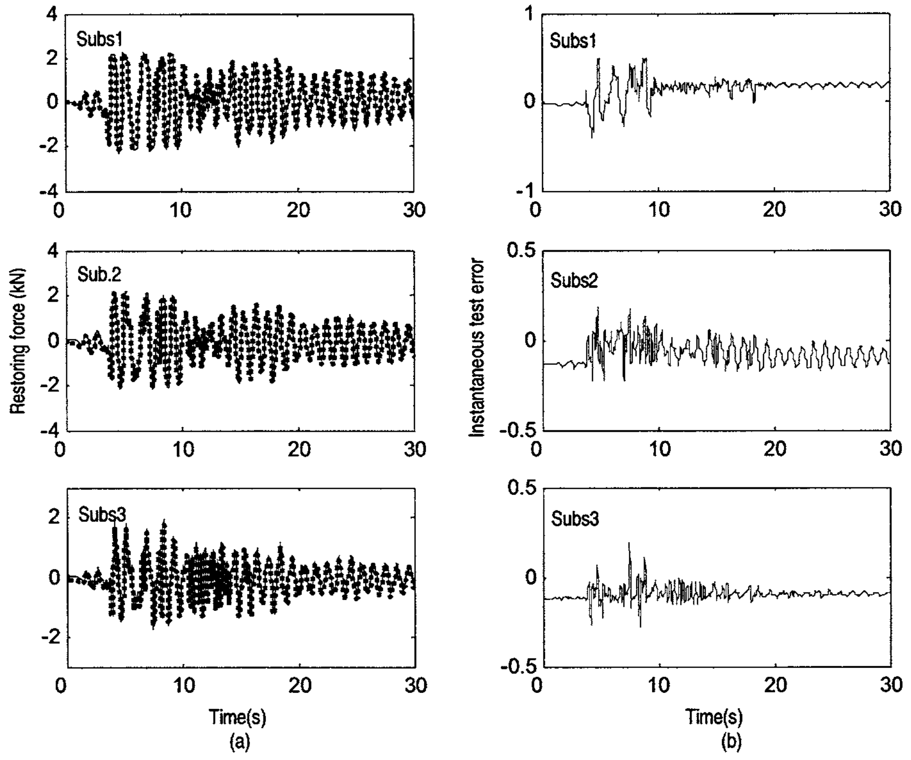

Fig. 11. Comparison between a segment of the actual output (solid line) and pruned H_∞-network (using El Centro responses with 5% noise injection in training data) output (dashed line) for the validation test set (using Kobe responses): (a) time history of restoring forces; (b) instantaneous test errors

can be calculated using Eq. (33). The parameters for this model are chosen to be $m_i=125.53$ kg, $c_i=70$ N s/m, $k_i=24{,}500$ N/m, $n_i=3$, $\alpha_i=2$, and $\beta_i=0.5$ ($i=1,2,3$).

Using substructure identification (Nakamura et al. 1998; Wu et al. 2002), the structure will be divided into several substructures. Here, one active degree of freedom is simplified as a substructure system (Subs). The FNN is taken into account to model the dynamic behaviors of the substructures, namely, the FNN is trained to identify a substructure model of $r_i(\dot{u}_i,u_i,r_i)$. In this paper, the hysteretic nonlinearity system is considered as a memory-type model, and the restoring force with one step time delay will be selected as the input (Smyth et al. 2002; Masri et al. 2004; Tang and Sato 2004). A three-layer FNN is applied in which the input signals are $\dot{u}_i$ and u_i at time step k, and r_i at time step $k-1$, and the outputs are the calculated values of r_i at time step k. Hence, the network has three input nodes, one output node, and one hidden layer with 30 nodes. The total number of the network weights is 120.

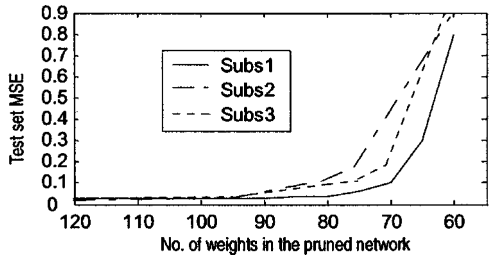

Fig. 12. Number of weights in the pruned network versus the test set MSE of the system identification problem

In this study, the effectiveness of the noise injection training is also investigated; noises are artificially added to structural responses in generating 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 to as 5% in the root-mean-square (RMS) level.

Case 1: Nonlinear System Identification

El Centro (May 18, 1940) with modified maximum amplitude of 25 cm/s^2 and 30 s time history are used in this case. The sampling interval of the structural responses to be used for identification is $\Delta t=0.02$ s. The training data sets are prepared for cases with 0 and 5% noise in the RMS level.

The time history of some parts of the estimated network's weights is shown in Figs. 3(a–c). Some weights converge to stable values very quickly in the first 500 iterations. This fast convergence demonstrates that the H_∞ filtering is a very fast training algorithm.

Excellent online restoring force estimation results using the proposed approach are shown in Fig. 4(a) (0% noise). The actual and identified force-displacement loops are shown in Fig. 4(b) (0% noise), where a quite complete agreement is clear between the actual and identified loops. Moreover, the tendency is the same while varying the noise level, and one example of 5% noise level is shown in Fig. 5.

To verify the generalization ability of the H_∞-training FNN, the trained networks mentioned above are applied to predict responses to another 500 test samples. It is noted that these test samples have not been used during the training phase. The time history of the predicted restoring forces is shown in Fig. 6(a) (0% noise), which shows very good agreement with the true results,

Downloaded from ascelibrary.org by Tongji University on 12/08/18. Copyright ASCE. For personal use only; all rights reserved.

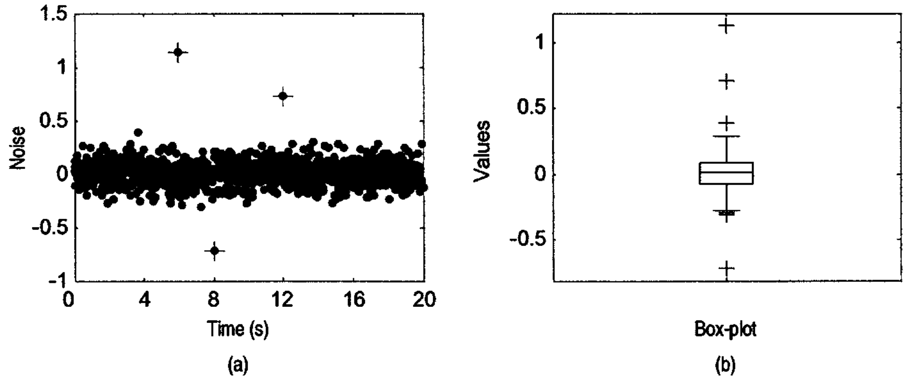

Fig. 13. Zero mean Gaussian noise and three outliers (denoted by "+")

and Fig. 6(b) (0% noise) shows the corresponding instantaneous test errors. Moreover, Fig. 7 shows the estimation results with 5% noise level. It is seen that the network is performing extremely well in matching the system's response.

Case 2: H_∞ Filtering in Network Pruning

To illustrate the conjunction of the H_∞ filtering algorithm in network training and pruning, the example of the above-mentioned nonlinear substructure identification is studied. Different noise cases and two different types of structural responses are selected for example analysis to illustrate the applicability of the proposed approach. There are two structural response data, one is by El Centro, and the other is by the Kobe earthquake (January 17, 1995, modified maximum amplitude of 25 cm/s^2, sampling interval $\Delta t=0.02$ s).

The total number of the network weights is 120, and the network is trained using the H_∞ filtering algorithm. The training set contains 1,000 samples using El Centro responses with 5% noise in the RMS level, and the test set contains 500 samples using Kobe earthquake responses.

Figs. 8(a–c) show the last step values of $\mathbf{P}_k$ (120×120 matrix) of the Riccati recursion for the three substructure identifications using El Centro response data. Figs. 8(a–c) show that the matrix $\mathbf{P}_k$ is almost diagonally dominant. Fig. 9 shows the estimated training error against the number of weights in the pruned network. It is clear that nearly only 95 weights are enough to capture the unknown Substructure System 1 without increasing the training error dramatically, and 90 weights suffice for Substructure Systems 2 and 3. Another 500 pairs of test data are used to validate the pruned network. Fig. 10(a) shows a comparison of the pruned H_∞-neural network output and the actual system output. Fig. 10(b) depicts corresponding loops of restoring forces versus relative displacements. Fig. 10 shows that the generalization capability of the pruned network does not change much. To demonstrate further that the generation capability is not affected much if some weights in the networks are pruned, we feed in other test data, generated by the Kobe earthquake, into the pruned network. Fig. 11(a) depicts a segment of the actual and the desired pruned network output corresponding to the test input, and Fig. 11(b) shows corresponding instantaneous test errors. Fig. 11 clearly shows that the generalization capability of the pruned network is not affected much. It is seen that in both cases the pruned network performs well in modeling the unknown system.

We use the mean-squared error (MSE) to measure the generalization ability of the pruned network. It is calculated by feeding a test set, generated by the Kobe earthquake, into the pruned network with different pruning steps. The gathered data are plotted in Fig. 12, which shows that the average generalization error will increase as the number of pruned weights increases. It shows that the test MSE of the pruned network is quite close to the original network when the number of pruned weights less than 95

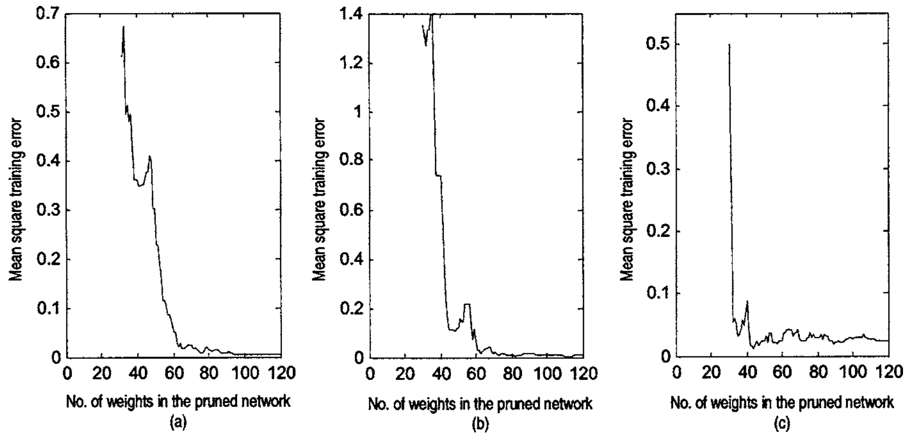

Fig. 14. Number of weights in the pruned network versus the training error (using Kobe responses with 5% noise injection in training data): (a) network for Subs 1 identification; (b) network for Subs 2 identification; and (c) network for Subs 3 identification

Downloaded from ascelibrary.org by Tongji University on 12/08/18. Copyright ASCE. For personal use only; all rights reserved.

Downloaded from ascelibrary.org by Tongji University on 12/08/18. Copyright ASCE. For personal use only; all rights reserved.

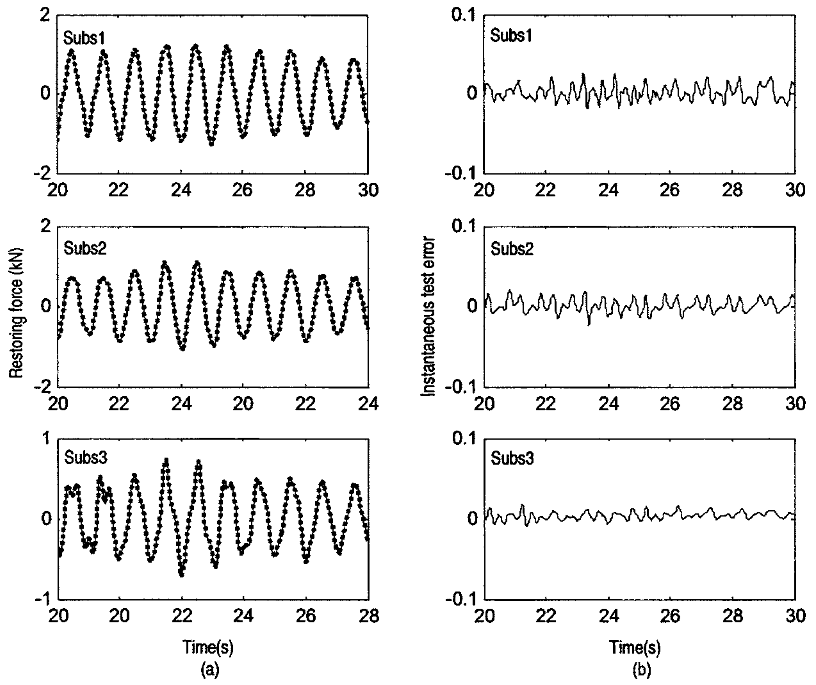

Fig. 15. Comparison between a segment of the actual output (solid line) and pruned H_∞-network (using Kobe responses with 5% noise injection in training data) output (dashed line) for the test set: (a) time history of restoring forces; (b) instantaneous test errors

for Subs 1, and 90 for Subs 2 and Subs 3. This phenomenon is consistent with our expectation.

In addition, comparative studies for two kinds of noise injection training and pruning are presented to verify the robustness and generalization capability of the proposed methodology using the Kobe earthquake responses. One is the Gaussian noise case where the training set was corrupted by 5% noise in RMS level. The other is the non-Gaussian noise case where the training set was corrupted by zero mean Gaussian noise (with 5% noise in RMS level) and three outliers (denoted by "+"). One example of the non-Gaussian noise for the Subs 1 network's output training data is shown in Fig. 13(a). The long, lower tail and plus signs in the box plot of Fig. 13(b) denote the strong outliers in the sample values.

Fig. 14 shows the estimated training error against the number of weights in the pruned network for the Gaussian noise case. It is found that about 80–90 weights are enough to capture the unknown Substructure System 1 without increasing the training error dramatically, and about 80–100 weights suffice for Substructure Systems 2 and 3. The results are almost identical with the results (Fig. 9) by using the El Centro responses. To demonstrate that the generation capability is not affected much if some weights are pruned, we feed in another 500 pairs of test data into the pruned network. Fig. 15(a) shows a comparison of the pruned H_∞-neural network outputs between the actual system outputs. Fig. 15(b) shows corresponding instantaneous test errors. These figures show that the generalization capability of the pruned networks is not affected much.

For the non-Gaussian noise case, the estimated training errors against the number of weights in the pruned network are shown in Fig. 16. Comparisons of the net weight prunings plotted in Fig. 14 with those in Fig. 16 show a similar "minimum" weight number.

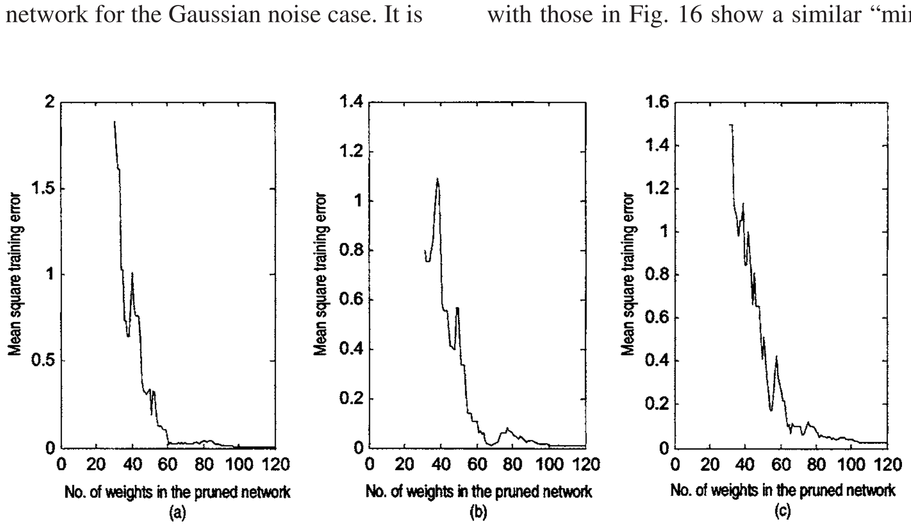

Fig. 16. Number of weights in the pruned network versus the training error (using Kobe earthquake responses with non-Gaussian noise injection in training data): (a) network for Subs 1 identification; (b) network for Subs 2 identification; and (c) network for Subs 3 identification

Downloaded from ascelibrary.org by Tongji University on 12/08/18. Copyright ASCE. For personal use only; all rights reserved.

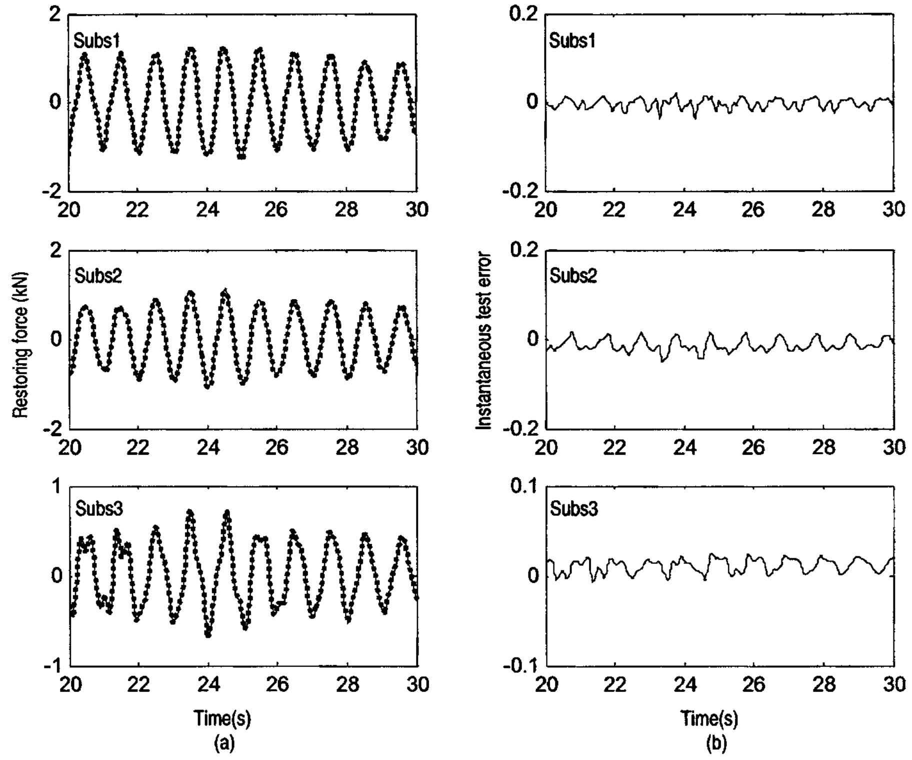

Fig. 17. Comparison between a segment of the actual output (solid line) and pruned H_∞-network (using Kobe earthquake responses with non-Gaussian noise injection in training data), output (dashed line) for the test set: (a) time history of restoring forces; (b) instantaneous test errors

Fig. 17(a) presents a comparison of the pruned H_∞-neural network outputs between the actual system outputs using another 500 pairs of testing data, and Fig. 17(b) shows the corresponding instantaneous test errors. There is good agreement between the output of the pruned network and the actual one. Fig. 17 shows that the generalization capability of the pruned network does not change much for strong outliers. Fig. 17 also shows that H_∞ filtering algorithm has a good robustness to such disturbances.

In general, the Kalman filtering algorithm is strictly optimal only for Gaussian distribution of noises, while the H_∞ filtering algorithm makes no assumptions about the noise distribution. In addition, the H_∞ theory suggests that the maximum energy (worst case) gain of the Kalman algorithm from disturbances (initial state, system, and observation noises) to the estimation error has no upper bound because the H_∞ algorithm when $\gamma \to \infty$ is formally identical to the Kalman algorithm (Hassibi et al. 1998). This is why we believe that the H_∞ training leads to greater robustness to variations in weight initialization or to deterministic disturbance in observation.

Conclusions

In this paper, we have derived an efficient neural network training and pruning methodology using the H_∞ filtering algorithm. The FNN 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. It leads to a lower hardware implementation cost and more efficient estimation of large complex structural systems. Examples of nonlinear system identification are carried out to evaluate the performance of the network. The results show the effectiveness and robustness of the neural network pruning and training by the H_∞ filtering algorithm.

Acknowledgments

One of the writers (H.S.T.) would like to thank Kyoto University for providing a Postdoctoral Fellowship. He also thanks Professor Rong Chen of Tongji University for his precious advice. Helpful comments and suggestions by anonymous reviewers are also greatly appreciated.

References

Abid, S., Fnaiech, F., and Najim, M. (2001). "A fast feedforward training algorithm using a modified form of the standard back-propagation algorithm." *IEEE Trans. Neural Netw.*, 12(2), 424–434.

Abrahart, R. J., See, L., and Kneal, P. E. (2001). "Investigating the role of saliency analysis with neural network rainfall–runoff model." *Comput. Geosci.*, 27(8), 921–928.

Akaike, H. (1974). "A new look at the statistical model identification." *IEEE Trans. Autom. Control*, 19(6), 716–723.

Bebis, G., Georgiopoulos, M., and Kasparis, T. (1997). "Coupling weight elimination with genetic algorithm to reduce network size and preserve generalization." *Neurocomputing*, 17(3-4), 167–194.

Bojarczak, O. S. P., and Stodolski, M. (1996). "Fast second-order learning algorithm for feedforward multilayer neural networks and its application." *Neural Networks*, 9(9), 1583–1596.

Chen, H. M., Qi, G. Z., Yang, J. C. S., and Amini, F. (1995). "Neural network for structural dynamic model identification." *J. Eng. Mech.*, 121(12), 1377–1381.

Chen, S., Cowan, C. F. N., Billings, S. A., and Grant, P. M. (1990).

"Parallel recursive prediction error algorithm for training layered neural network." *Int. J. Control*, 51(6), 1215–1228.

Didinsky, G., Pan, Z., and Basar, T. (1995). "Parameter identification of uncertain plants using H_∞ methods." *Automatica*, 31(9), 1227–1250.

Fukuda, W. K. T., and Tzafestas, S. G. (1991). "Learning algorithms of layered neural networks via extended Kalman filters." *Int. J. Syst. Sci.*, 22(4), 753–768.

Haddadnia, J., Faez, K., and Ahmadi, M. (2002). "*N*-feature neural network human face recognition." *Proc., 15th Int. Conf. on Vision Interface*, Calgary, Canada, 300–307.

Hassibi, B., Sayed, A. H., and Kailath, T. (1998). *Indefinite-quadratic estimation and control: A unified approach to H_2 and H_∞ theories*, SIAM, Philadelphia.

Hassibi, B., and Stork, D. G. (1993). "Second-order derivatives for network pruning: Optimal brain surgeon." *Advances in neural information processing 4*, S. J. Hanson, J. D. Cowan, and C. Lee Giles, eds., Morgan Kaufmann, San Mateo, Calif., 164–171.

Haykin, S. (1996). *Adaptive filter theory*, 3rd Ed., Prentice-Hall, Englewood Cliffs, N.J.

Haykin, S. (1999). *Neural networks: A comprehensive foundation*, Prentice-Hall, Englewood Cliffs, N.J.

Iiguni, Y., Sakai, H., and Tokumaru, H. (1992). "A real-time learning algorithm for a multilayered neural network based on the extended Kalman filter." *IEEE Trans. Signal Process.*, 40(4), 959–966.

Jaakkola, T., Jordan, M., and Singh, S. (1994). "On the convergence of stochastic iterative dynamic programming algorithms." *Neural Comput.*, 6(6), 1185–1201.

Kollias, S., and Anastassiou, D. (1989). "An adaptive least-squares algorithm for the efficient training of artificial neural networks." *IEEE Trans. Circuits Syst.*, 36(8), 1092–1101.

Krogh, A., and Hertz, J. (1992). "A simple weight decay can improve generalization." *Advances in neural information processing systems 4*, D. S. Touretzky, ed., Morgan Kaufmann, San Mateo, Calif., 950–957.

LeCun, Y., Denker, J. S., and Solla, S. A. (1990). "Optimal brain damage." *Advances in neural information processing 1*, D. S. Touretzky, ed., Morgan Kaufmann, San Mateo, Calif., 396–404.

Leung, C. S., Wong, K. W., Sum, J., and Chan, L. W. (1996). "On-line training and pruning for RLS algorithms." *Electron. Lett.*, 32(23), 2152–2153.

Leung, C., Wong, K., Sum, P., and Chan, L. (2001). "A pruning method for the recursive least squared algorithm." *Neural Networks*, 14(2), 147–174.

Makarynskyy, O., Pires-Silva, A. A., Makarynska, D., and Ventura-Soares, C. (2005). "Artificial neural networks in wave predictions at the west coast of Portugal." *Comput. Geosci.*, 31(4), 415–424.

Masri, S. F., Caffrey, J. P., Caughey, T. K., Smyth, A. W., and Chassiakos, A. G. (2004). "Identification of the state equation in complex nonlinear systems." *Int. J. Non-Linear Mech.*, 39(7), 1111–1127.

Masri, S. F., Nakamura, M., Chassiakos, A. G., and Caughey, T. K. (1996). "Neural network approach to detection of changes in structural parameters." *J. Eng. Mech.*, 122(4), 350–360.

Masri, S. F., Smyth, A. W., Chassiakos, A. G., Caughey, T. K., and Hunter, N. F. (2000). "Application of neural networks for detection of changes in nonlinear systems." *J. Eng. Mech.*, 126(7), 666–676.

Nakamura, M., Masri, S. F., and Caughey, T. K. (1998). "A method for nonparametric damage detection through the use of neural networks." *Earthquake Eng. Struct. Dyn.*, 27(9), 997–1010.

Prechelt, L. (1996). "A quantitative study of experimental evaluations of neural network learning algorithms: Current research practice." *Neural Networks*, 9(3), 457–462.

Rumelhart, D., Hinton, G., and Williams, G. (1986). "Learning internal representations by error propagation." *Parallel distributed processing 1*, MIT, Cambridge, Mass., 318–362.

Sato, T., and Qi, K. (1998). "Adaptive H_∞ filter: Its application to structural identification." *J. Eng. Mech.*, 124(11), 1233–1240.

Shah, S., Palmeieri, F., and Datum, M. (1992). "Optimal filtering algorithms for fast learning in feedforward neural networks." *Neural Networks*, 5(5), 779–787.

Singhal, S., and Wu, L. (1989). "Training multiplayer perceptrons with the extended Kalman algorithm." *Advances in neural information processing systems 1*, D. S. Touretzky, ed., Morgan Kaufmann, San Mateo, Calif., 133–140.

Smyth, A. W., Masri, S. F., Kosmatopoulos, E. B., Chassiakos, A. G., and Caughey, T. K. (2002). "Development of adaptive modeling techniques for nonlinear hysteretic systems." *Int. J. Non-Linear Mech.*, 37(8), 1435–1451.

Stone, M. (1977). "An asymptotic equivalence of choice of model by cross-validation and Akaike's criterion." *J. R. Stat. Soc. Ser. B (Methodol.)*, 39(l), 44–47.

Sum, J., Leung, C., Young, G. H., and Kan, W. (1999). "On the Kalman filtering method in neural-network training and pruning." *IEEE Trans. Neural Netw.*, 10(1), 161–166.

Tang, H., and Sato, T. (2004). "Structural damage detection using the neural network and H_∞ algorithm." *Proc. SPIE*, 5394, 454–463.

Wen, Y.-K. (1976). "Method for random vibration of hysteretic systems." *J. Engrg. Mech. Div.*, 102(2), 249–263.

William, P. M. (1995). "Bayesian regularization and pruning use a Laplace prior." *Neural Comput.*, 7(1), 117–143.

Wu, Z., Xu, B., and Yokoyama, K. (2002). "Decentralized parametric damage detection based on neural networks." *Comput. Aided Civ. Infrastruct. Eng.*, 17(3), 175–184.

Downloaded from ascelibrary.org by Tongji University on 12/08/18. Copyright ASCE. For personal use only; all rights reserved.