Chapter 7 Neural Network Implementation with Experimental Supervision

7.1. Purpose

The reason for using neural networks for blade damage quantification and qualification is the complex nature of the problem faced. Smit [45] was able to identify damage on a single blade with sensors installed on the blade itself, using only the frequency shifts of a single frequency obtained from ARMAX models identified from the measured data. The aim of using neural networks in this dissertation is to train and test neural networks to quantify and qualify blade damage for multiple blades without the sensors being installed on the damaged blades themselves. In this chapter, neural networks will be trained with supervision from experimental measurements. This will be the practical situation faced with when use is made of less than one sensor per blade.

Although suitable neural networks take a lot of effort to train, they are very fast and easy to implement.

7.2. Network Feature Selection

It was decided to look at as many as possible features of the signals in order to identify useful those useful for damage detection. A number of the features considered are normally used for condition monitoring of impulse-related rotor defects such as rolling element bearing failure. However, these features were still considered in order to determine whether they will be able do provide additional information along with the use of other signal features. All the features considered are discussed in the following paragraphs and fall into two categories namely time domain features and frequency domain features.

7.2.1. Time Domain Features

Norton [38] outlines a number of time domain signal features, some of which are described in this section. Other time domain features include mean value and standard variation.

7.2.1.1. Root-Mean-Square Value

The mean-square value of a time dependant signal $x(t)$ over an interval T is the average value of $x^2(t)$ and is given by Equation (7-1) as

$$E[x^2] = \frac{1}{T}\int_0^T x^2 \mathrm{d}t \qquad (7\text{-}1)$$

The Root-Mean-Square (RMS) value can then be calculated by simply taking the positive square root of $E[x^2]$ as shown in Equation (7-2):

$$X_{RMS} = \sqrt{E[x^2]} = \sqrt{\frac{\int_0^T x^2(t)dt}{T}} \qquad (7\text{-}2)$$

Simply stated, an RMS value is the square root of the average of a squared time dependant signal.

7.2.1.2. Crest Factor

The crest factor is the ratio of the peak level of $x(t)$ to the RMS value of $x(t)$ and is calculated using Equation (7-3):

$$\text{Crest Factor} = \frac{X_{\max}}{X_{RMS}} \qquad (7\text{-}3)$$

This serves as a measure of the impulsiveness of a time signal and is commonly used to detect impulsive vibrations caused by damaged bearings.

7.2.1.3. Kurtosis

The kurtosis of a signal is the fourth statistical moment of the signal's probability density function and is useful for detection of the presence of an impulse in a signal. This is given by Equation (7-4):

$$\text{Kurtosis} = \frac{1}{\sigma^4 T}\int_0^T x^4 dt \qquad (7\text{-}4)$$

with σ^2 being the variance of $x(t)$ given by Equation (7-5):

$$\sigma^2 = E[x^2] - \{E[x]\}^2 \qquad (7\text{-}5)$$

$E[x^2]$ is calculated using Equation (7-1).

Similar to the crest factor, the kurtosis of a signal is normally used for detecting discrete impulse faults in rolling element bearings.

7.2.2. Frequency Domain

7.2.2.1. Operational Mode Shape Comparison

A modal analysis is usually performed on a stationary structure using the FRFs of a number of measurement points with one or more known artificial excitation forces as references. There are however a number of ways to estimate mode shapes from output-only data, i.e. for an unknown input force such as is the case for a rotating fan.

Smit [45] made use of ARMAX models to do damage detection for single blade damage. With this method, natural frequencies and mode shapes as well as input forces can be estimated from output-only time domain data. The accuracy of these estimations depends on the accuracy of the ARMAX models used. Consequently, Smit found that the models need to be of very high order in order to get meaningful results e.g. for data with a sampling frequency of 5120 Hz, 48th order ARMAX models are needed. As an ARMAX model has to be identified for each measurement, this method results in long computational times.

For this reason, it was decided to rather make use of the mode shape extraction method as described by Felber and Ventura [17]. Felber and Ventura made use of Averaged Normalized Power Spectral Densities (ANPSDs) as well as MRFs.

An ANPSD, used for identifying natural frequencies, is calculated as the average of a group of p Normalized Power Spectral Densities (NPSDs) as given by Equation (7-6)

$$ANPSD(f_k) = \frac{1}{p}\sum_{i=1}^{i=p} NPSD_i(f_k) \qquad \textbf{(7-6)}$$

with

$$NPSD_i(f_k) = \frac{PSD_i(f_k)}{\sum_{k=0}^{k=n} PSD_i(f_k)} \qquad \textbf{(7-7)}$$

being the NPSD for f_k (the k^{th} discrete frequency) and n the number of discrete frequencies. The function of the NPSD function is thus to normalize the values of different PSDs to their respective PSD energy levels in order to be comparable.

A PSD is calculated by Equation (7-8) as

$$PSD(f) = \frac{2}{N\Delta t} X(f)X^*(f) \qquad \textbf{(7-8)}$$

for N the number of PSD data points, Δt the sampling increment, $X(f)$ the Fourier transform of $x(t)$ and $X^*(f)$ the complex conjugate of $X(f)$.

A MRF, used for estimating mode shapes, is calculated by

$$M_{a,b}(f) = |TF_{a,b}(f)|PW(f)CW(f) \qquad \textbf{(7-9)}$$

with $TF_{a,b}(f)$ the transfer function, or more correctly, the transmissibility function between two records $\ddot{x}_a(t)$ and $\ddot{x}_b(t)$, $PW(f)$ a Phase-Window function and $CW(f)$ a Coherence-Window function.

$PW(f)$ is calculated as

$$PW(f)=\begin{cases}1 \text{ for} & 0\le\theta(f)\le\theta_c\\ 0 \text{ for} & \theta_c<\theta(f)<180^o-\theta_c\\ -1 \text{ for} & 180^o-\theta_c\le\theta(f)\le180^o\end{cases} \quad \textbf{(7-10)}$$

with $\theta(f)$ the phase function.

$CW(f)$ is calculated as

$$CW(f)=\begin{cases}1 \text{ for} & \gamma_c^{\,2}\le\gamma^2(f)\le1\\ 0 \text{ for} & 0<\gamma^2(f)<\gamma_c^{\,2}\end{cases} \quad \textbf{(7-11)}$$

with $\gamma_c^{\,2}(f)$ the coherence function

Both θ_c and $\gamma_c^{\,2}$ are chosen arbitrarily.

This method is very simple to implement although it does not yield as accurate and clear results as the ARMAX method does.

7.2.2.2. Rotational Speed Harmonics

Boek et al. [4] mention that vibrational information useful for distinguishing between different rotor defects of rotating machinery can be found in the amplitudes of the rotational frequency and its associated harmonics. Boek and his co-workers trained neural networks that were able to distinguish between physically different faults such as imbalance and a cracked blade in an experimental setup by using the first three rotational speed harmonics information and the average RMS values of the high frequency end of the measured spectra.

7.2.2.3. Cepstrum Analysis

Cepstrum analysis is a spectral analysis technique that is used to identify harmonics from PSDs as described by Norton [38], Rao [41] and Wismer [55]. The power cepstrum of a PSD is the inverse Fourier transform of the logarithmic PSD as given in Equation (7-12):

$$C_{pxx}(\tau)=X^{-1}\{\log_{10}[PSD(f)]\} \quad \textbf{(7-12)}$$

The power cepstrum is useful for amongst others the detection of harmonic patterns in frequency spectra such as for turbine blade failures ([38], [41]). Norton recommends using this technique as a complementary tool to spectral analysis.

7.3. Neural Network Suitability Test

The suitability of neural networks for the purposes of this dissertation was tested on data as measured by Smit [45]. Smit took measurements on the FaBCoM TeSt for single blade damage using two piezoelectric strain sensors and one accelerometer position on the damaged blade as shown in Figure 7-1. An additional accelerometer was placed on an undamaged blade at the same position as the one installed on the damaged blade. For each damage level (0%, 10%, 20%, 30% and 40%), 12 time domain measurements of record length 4 seconds were taken for all four transducers at a sampling frequency of 5120 Hz.

Features which were extracted from the signals, included time domain features (as described in section 7.2.1) and the rotational harmonics values for the z-z direction orientated strain sensor, ANPSD features using the signals from all four transducers as well as MRF features for the two piezoelectric strain sensors with the x-x direction orientated strain sensor as reference.

It was decided to make use of back-propagation networks as used by Boek et al. [4]. A number of these networks were trained using these features, but the best results were obtained using only the rotational harmonic and MRF features. The network yielding the best results consisted of a 5 neuron Tan-Sigmoid Transfer Function (TSTF) input layer and a single neuron Linear Transfer Function (LTF) output layer. The network was trained on 10 of the 12 feature sets extracted for each blade damage level with a 0.01 learning rate and a performance goal of $1x10^{-5}$. The network was tested on the remaining feature sets, the results of which are shown in Figure 7-2.

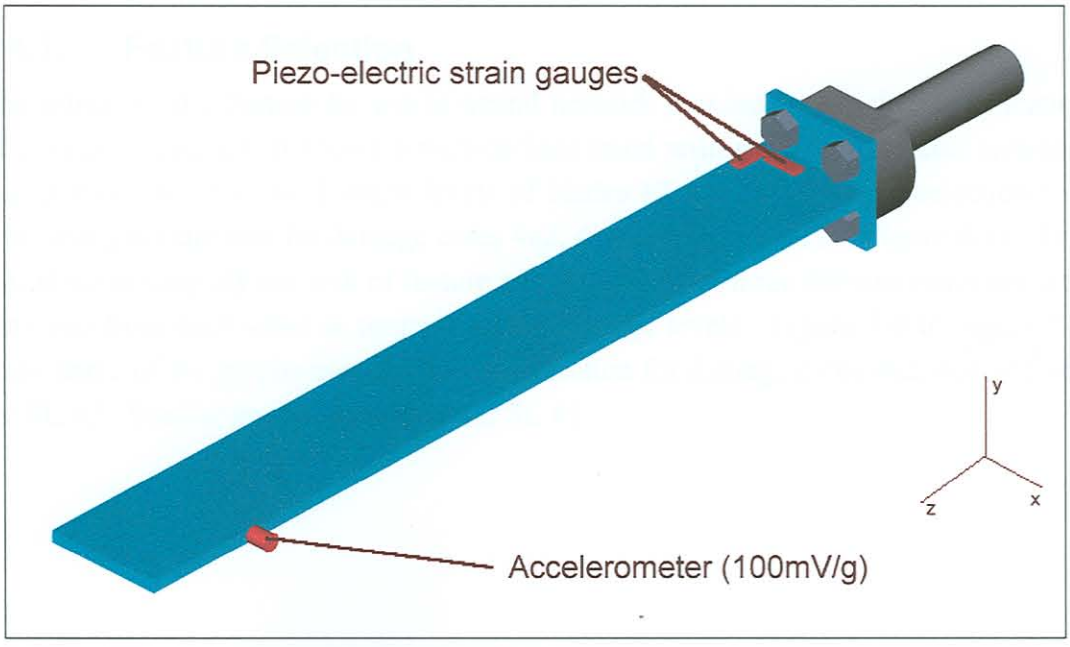

Figure 7-1: Sensor Orientation used by Smit

(from Smit [45])

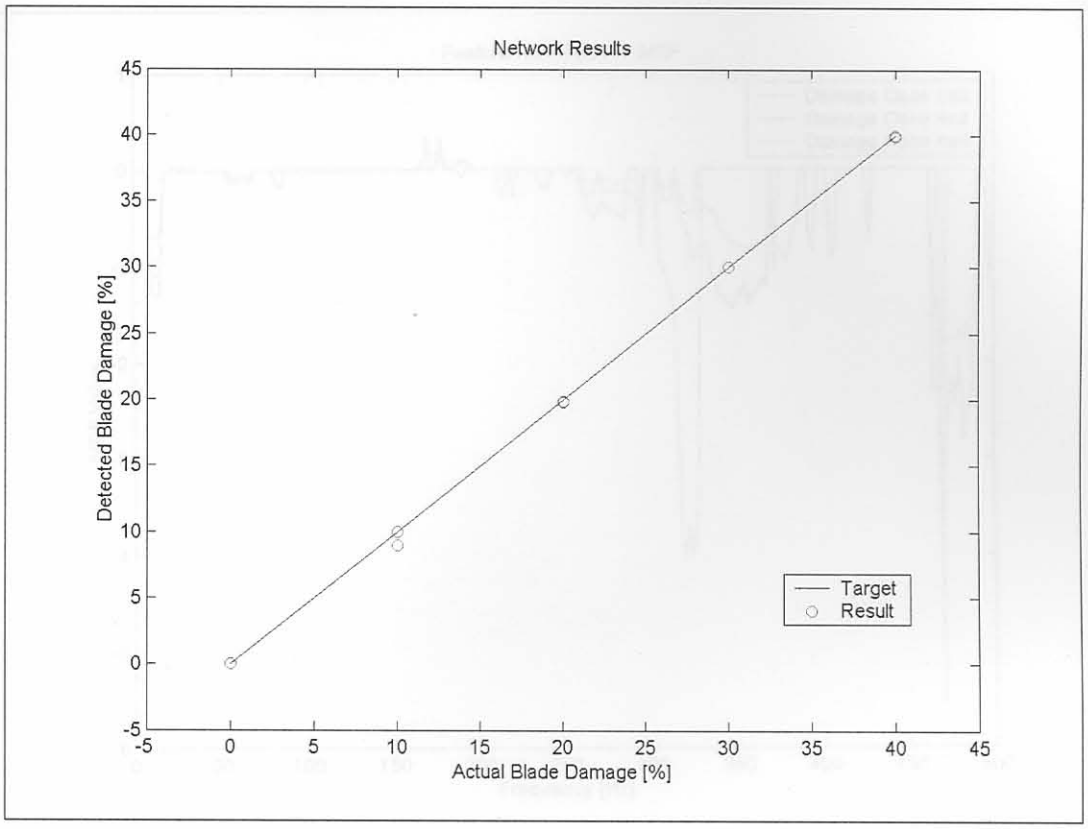

Figure 7-2: Results for Neural Network Suitability Test

From Figure 7-2 it can be concluded that a neural network was trained that is suitable for single blade damage detection when the setup of Smit [45] is used.

7.4. Neural Networks

7.4.1. Feature Selection

The selection of a feature for use in neural network training entails the identification of a dataset feature that shows a well-defined trend with regards to certain variables that in this case, are the damage levels of blades #3 and #4. Feature selection was done using feature sets for damage cases 4a2, 4c4 and 4e6 only (see Figure 6-1). This was done to simplify the task of feature selection as these three damage cases are well removed from each other in terms of blade damage levels. Figure 7-3 to Figure 7-6 show some of the graphs used for feature selection for damage cases 4a2, 4c4 and 4e6 for SL #3. Similar graphs were used for SL #1.

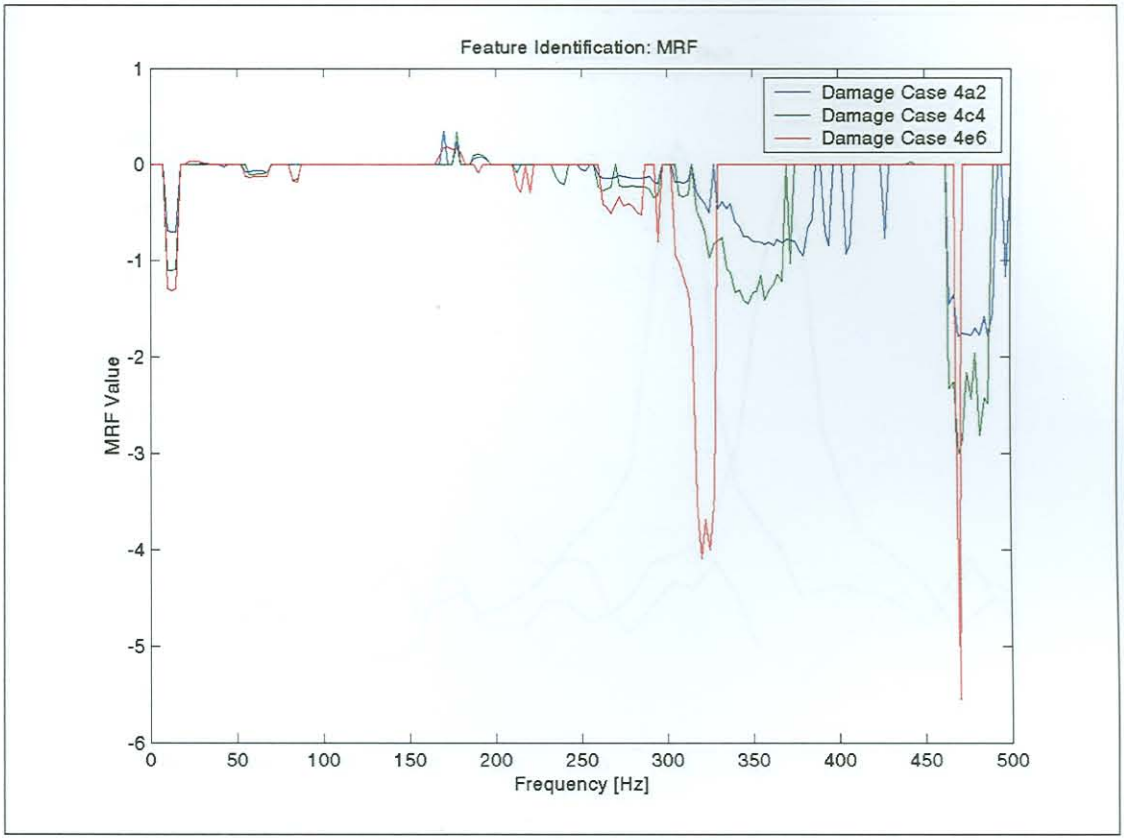

Figure 7-3: MRF Results over 500 Hz Bandwidth for SL #3

Figure 7-3 shows the MRF results for SL #3 over a 500 Hz bandwidth as calculated using Equation (7-9). From Figure 7-3, possible features include the MRF amplitude around 12.5 Hz, the minimum MRF amplitude between 460 Hz and 495 Hz as well as the frequency index of the MRF minimum between 255 Hz and 390 Hz.

Figure 7-4 shows the shift of peaks of the strain sensor PSD function between 1300 Hz and 1370 Hz, identified as feature #11 for SL #3. Figure 7-5 shows the use of PSD energy for the strain sensor data for SL #3 as in feature #8. Here, the PSD energy is obtained by calculating the area underneath the PSD graph from 1795 Hz to 1805 Hz. Definite trends are clearly seen on these figures.

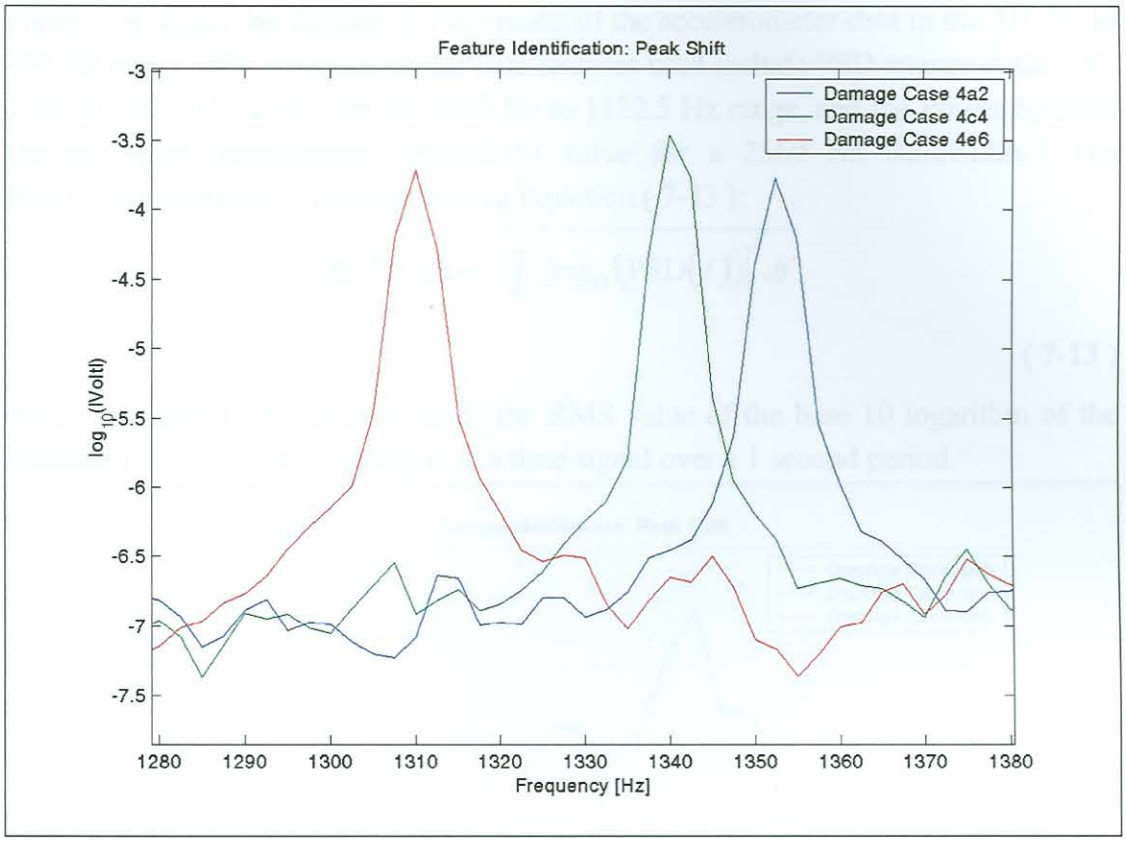

Figure 7-4: SL #3, Feature # 11

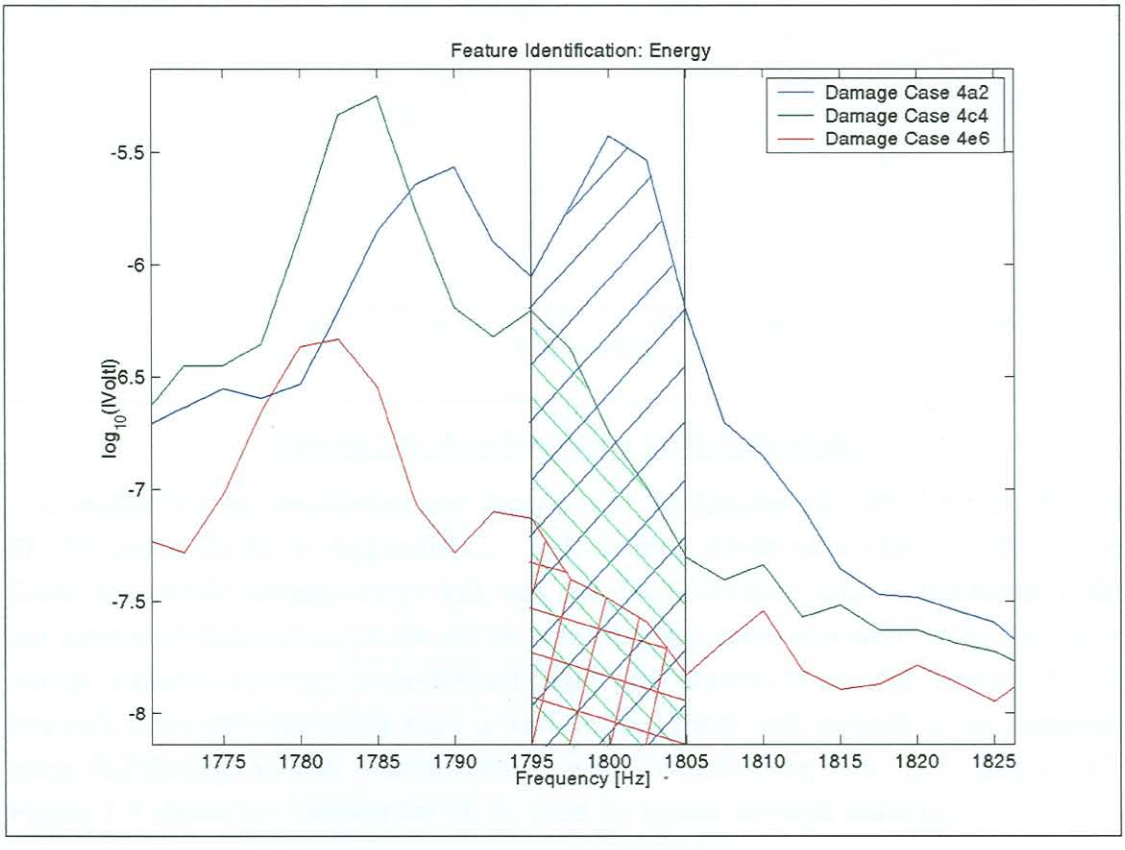

Figure 7-5: SL #3, Feature #8

Figure 7-6 shows the shifting of PSD peaks of the accelerometer data in the 310 Hz to 370 Hz range. Other accelerometer data features used include PSD energy in the 347. 5 Hz to 357.5 Hz as well as the 1085 Hz to 1122.5 Hz range, and the Power Spectral Density Root-Mean-Square (PSDRMS) value for a 2560 Hz bandwidth. The PSDRMS values were calculated using Equation (7-13):

$$\text{PSDRMS} = \sqrt{\int_{f_0}^{f_1} \left[\log_{10}\left(|\text{PSD}(f)|\right)\right]^2 df} \qquad \textbf{(7-13)}$$

which basically is the calculation of the RMS value of the base 10 logarithm of the absolute PSD values by treating it as a time signal over a 1 second period.

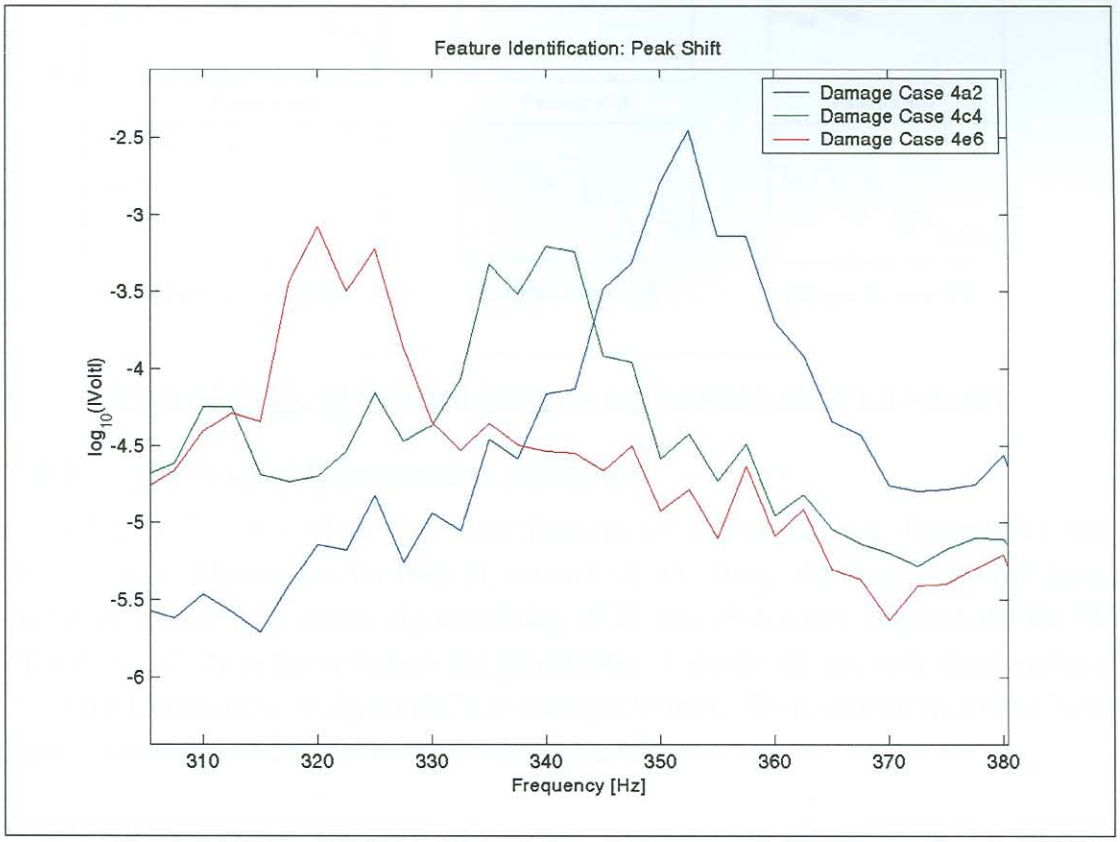

Figure 7-6: Accelerometer PSD Peak Shift

A complete listing, description and depiction of the features identified for SL #1 and SL #3 can be found in Appendix C. Well-defined trends were obtained for all the listed features for damage cases 4a2, 4c4 and 4e6. However, a lot of the features did not have well defined trends for all the other damage cases and were noisy. In other words, a feature having a well-defined trend with regards to damage cases 4a2, 4c4 and 4e6, does not necessarily have a well-defined trend with regards to say damages cases 4a2 through to 4a6. These features were discarded for both SL #1 and SL #3. Figure 7-7 shows the features for SL #3 used for neural network training.

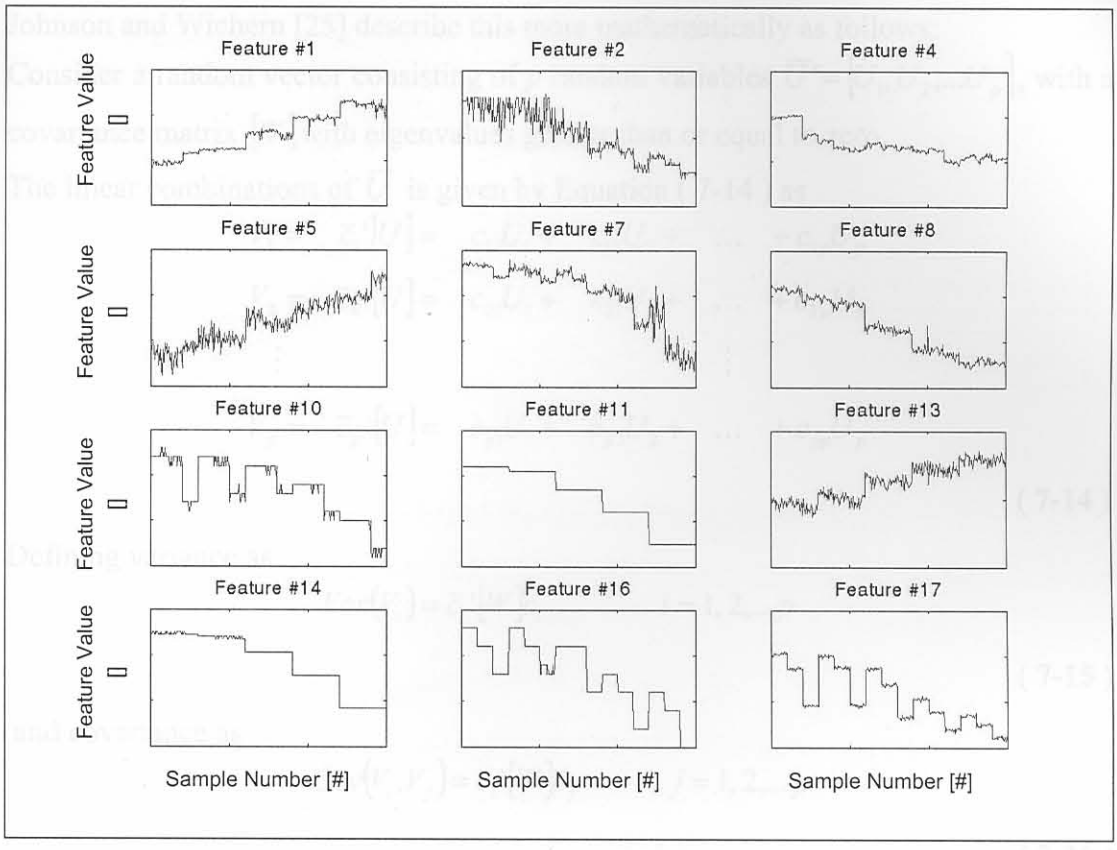

Figure 7-7: SL #3 Selected Features for Neural Network Training

7.4.2. Principal Component Analysis

From Figure 7-7, it is clear that some features are correlated (e.g. feature #11 and feature #14). This occurs for both SL #1 and SL #3. Also, the neural network input vector dimensions are rather big consisting of 20 and 19 features respectively for SL #1 and SL #3. In order to reduce the dimensions of the neural network input vectors, Principal Component Analyses (PCAs) were performed. This resulted in a final input vector dimension of 5x1 for both SL #1 and SL #3.

Principal components are particular linear combinations of a group of random variables considered and are solely dependant on the covariance matrix of the variables as described by Johnson and Wichern ([25]). According to them, PCAs often reveal unsuspected relationships and frequently serve as intermediate steps in larger investigations.

The technique has three effects namely input vector orthogonolisation, orthogonolized component sorting and component elimination ([11]). In other words, the technique first orthogonolizes the input vector so that its components are uncorrelated. It then sorts the resulting components so that those with the largest variation come first. After that, components that contribute least to the data set variation are eliminated.

Johnson and Wichern [25] describe this more mathematically as follows:
Consider a random vector consisting of p random variables $\overline{U}' = [U_1, U_2, \ldots U_p]$, with a covariance matrix $[W]$ with eigenvalues greater than or equal to zero.
The linear combinations of $\overline{U}$ is given by Equation (7-14) as

$$\begin{aligned} V_1 &= \overline{c}_1'[U] = c_{11}U_1 + c_{12}U_2 + \ldots + c_{1p}U_p \\ V_2 &= \overline{c}_2'[U] = c_{21}U_1 + c_{22}U_2 + \ldots + c_{2p}U_p \\ &\vdots \qquad\qquad\qquad\qquad \vdots \\ V_p &= \overline{c}_p'[U] = c_{p1}U_1 + c_{p2}U_2 + \ldots + c_{pp}U_p \end{aligned} \qquad \textbf{(7-14)}$$

Defining variance as

$$Var(V_i) = \overline{c}_i'[W]\overline{c}_i \qquad i = 1, 2, \ldots p \qquad \textbf{(7-15)}$$

and covariance as

$$Cov(V_i, V_j) = \overline{c}_i'[W]\overline{c}_j \qquad i, j = 1, 2, \ldots p \qquad \textbf{(7-16)}$$

the i^{th} principal component will be the linear combination $\overline{c}_i'\overline{U}$ that maximizes $Var(\overline{c}_i'\overline{U})$ for $\overline{c}_i'\overline{c}_i = 1$ and $Cov(\overline{c}_i'\overline{U}.\overline{c}_j'\overline{U}) = 0$ for $j < i$.

7.4.3. Network Architecture

Several neural networks were trained and tested in order to determine suitable network architecture. This was done using features extracted from data measured for the adjacent blade damage scenario as explained in Chapter 6. All the networks trained were feed-forward networks using back-propagation training. The best results were obtained for four layer networks with two outputs with the first two layers using TSTFs and the remaining two layers making use of LTFs.

Subsequently, all neural networks referred to further on in this paper, have this architecture. One could argue that due to the complex nature of the damage classification to be done, complex network architecture needs to be used. Figure 7-8 presents a graphical presentation of such network architecture for a 5-neuron input layer, two 5-neuron hidden layers, 2-neuron output layer (5x5x5x2) network.

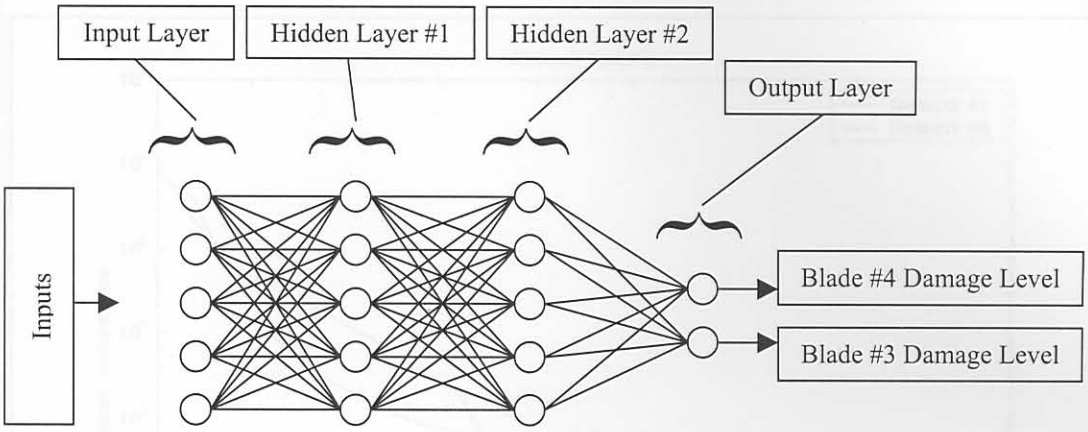

Figure 7-8: 5x5x5x2 Neural Network Architecture

For SL #1, a 20x20x20x2 network yielded the best results whereas for SL #3, a 10x10x10x2 network yielded the best results.

7.4.4. Network Training

As already mentioned, two networks were trained. The networks trained for SL #1 and SL #3 will be referred to as Network #1 and Network #3 respectively. The networks were trained on 20 feature sets extracted from 20 of the 24 measured datasets for each of the damage cases of the 1st, 3rd and 5th columns of the damage case matrix given in Figure 6-1. Both networks were trained using the same parameters listed in Table 7-1 using the Levenberg-Marquardt training algorithm.

Table 7-1: Neural Network Training Parameters

Training Parameter	Value
Performance Goal	0.001
Learning Rate	0.01
Epochs	2000

All other training parameters used the default values assigned by the Neural Network Toolbox for Matlab. Figure 7-9 shows the logarithmic network performance during training of Networks #1 and #3.

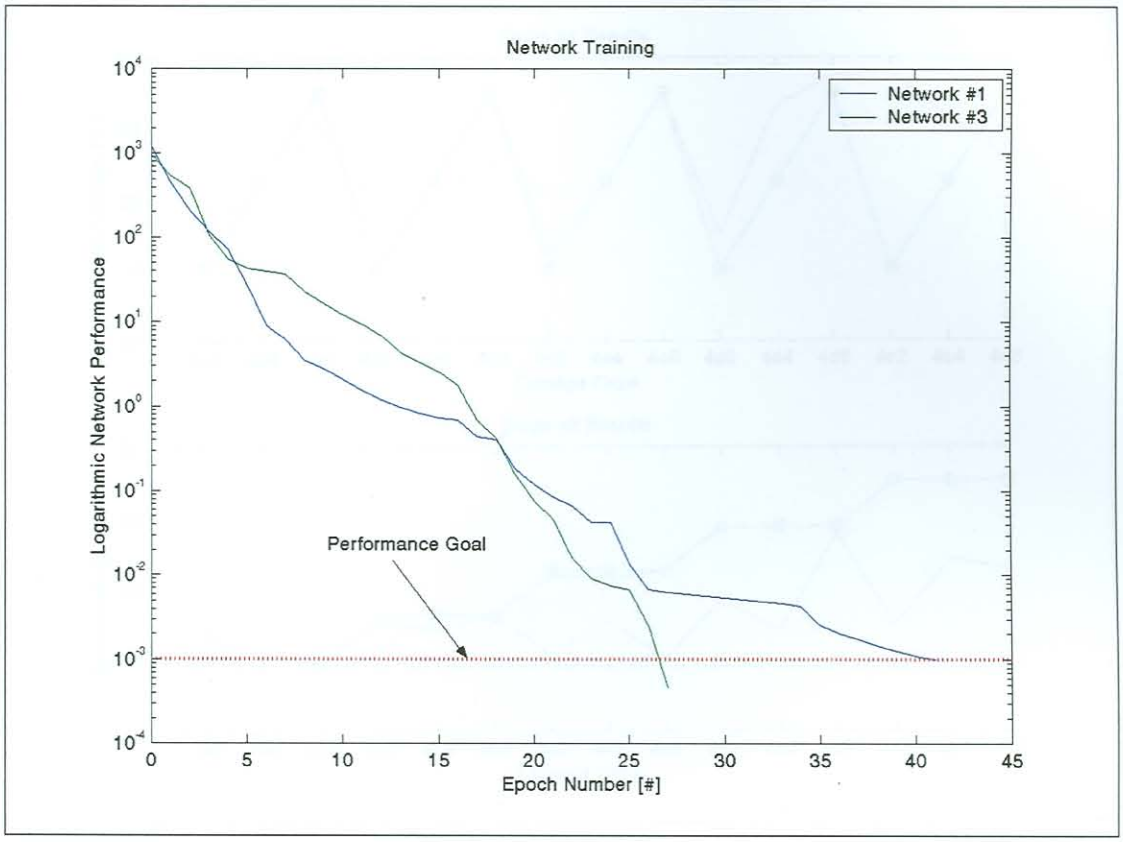

Figure 7-9: Training of Network #1 and Network #3

7.5. Results

In this section, the results of the neural networks are presented in terms of trained damage levels and untrained damage levels. The reason why the networks were tested for untrained damage levels was to test the ability of the networks to interpolate between damage levels for which they trained. Also, the damage level extrapolation capabilities of the networks were tested for damage cases 2a4 and 2a6, which represent a 3-blade damage scenario as described in Section 6.4. Also, to test the robustness of Network #1, it was tested with feature sets for SL #2. The same features were extracted as those for SL #1. The results obtained from Network #1 for SL #2 will be referred to as Network #2.

Figure 7-11 to Figure 7-13 (the key of which is given in Figure 7-10) give the mean results for Networks #1, #2 and #3. It was decided to average the network results as individual results proved to be a bit noisy.

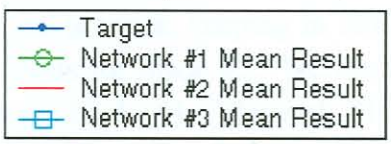

Figure 7-10: Key for Figure 7-11 to Figure 7-13

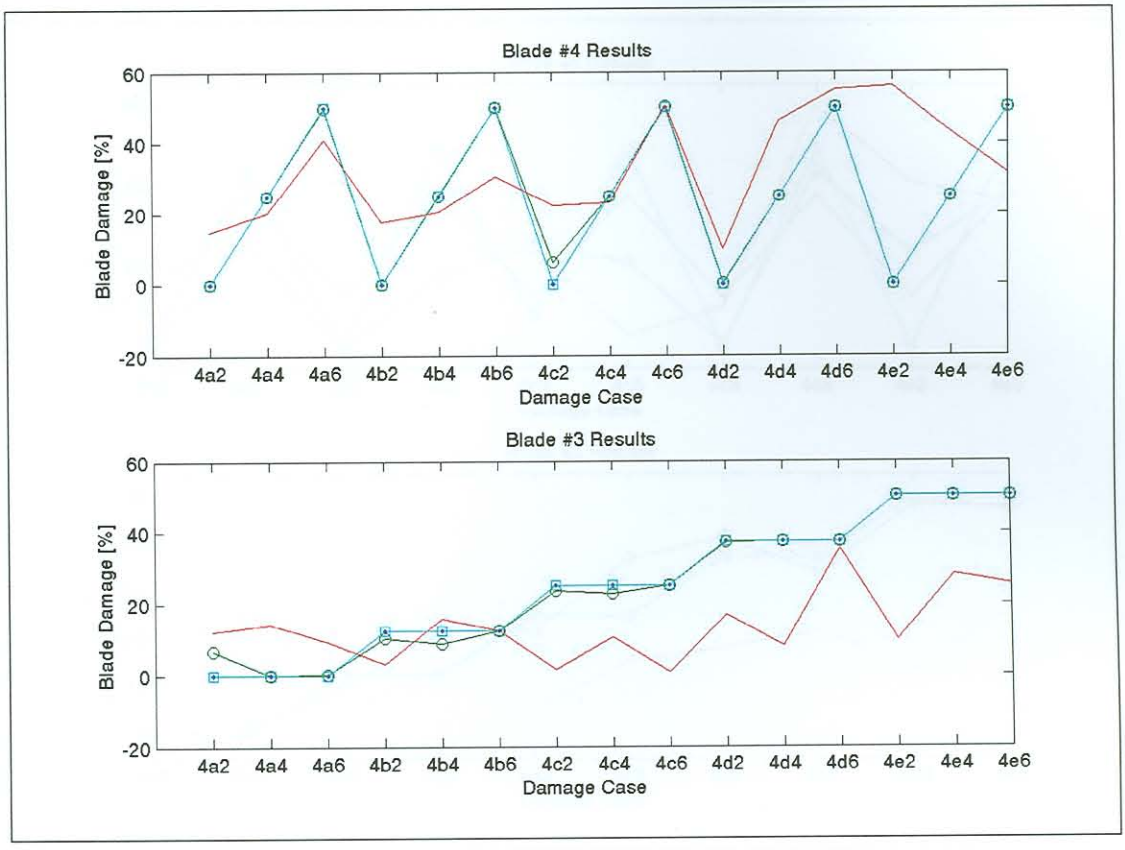

Figure 7-11: Results for Trained Damage Levels

Figure 7-11 gives the mean results for the trained damage levels of the networks. This was done as preliminary network tests. For Networks #1 and #3, the features sets used were those extracted from the remaining four of the 24 measured datasets as mentioned in section 7.4.4. For Network #2, the feature sets from all 24 measured datasets for each of the damage cases as described in section 7.4.4 were used for testing.

From Figure 7-11 it can be seen that Networks #1 and #3 perform very well. However, the results from Network #2 is not as good, especially for damage level detection of blade 3. The reason for the very poor results for damage cases 4e2, 4e4 and 4e6, is that the strain sensor on blade #2 got loose during testing of blade 4e, as mentioned in Chapter 6.

Figure 7-12 gives the mean results of the networks for damage level interpolation. From the three networks, Network #3 performs the best. Excellent damage detection is obtained for blade #3 by the network, keeping in mind that all the networks were trained for all the damage levels for blade #3. Network #3 performs rather well in terms of damage detection for blade #4. Network #1 performs less well but still acceptable for blade #3 damage detection. It does not perform very well for damage detection of blade #4. The results of Network #2 proved to be rather useless.

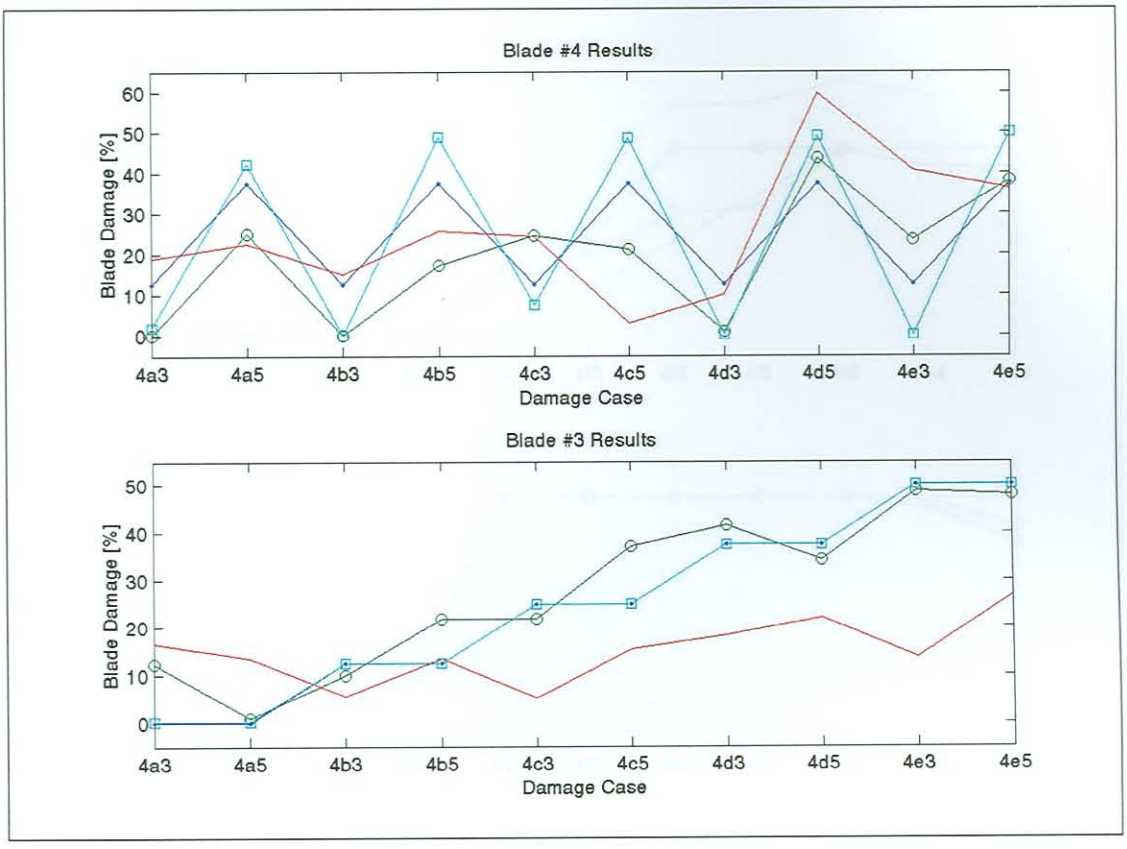

Figure 7-12: Results for Damage Level Interpolation

Figure 7-13 gives the results for extra feature sets not used in network training as well as two damage cases, namely 2a4 and 2a6, for which the networks were not trained. It can be seen for damage cases 2a4 and 2a6, that Network #1 and Network #3 interpret the additional damage as damage reduction both for blades #4 and #3 although this is more apparent for blade #3. From this, it is safe to conclude that the networks will be able to do damage classification for three blades and perhaps all four blades if trained to do so. Also, it can be concluded that if the networks find the structural health to improve, it is possibly due to the incorrect interpretation of additional damage.

Feature sets from damage cases 4b1, 4c1, 4d1 and 4e1 were used to test the robustness of the networks in terms of blade manufacturing and installation tolerances. These damage cases are all equal in terms of damage levels to the undamaged case 4a2, the feature sets of which was used to train the networks. As seen from Figure 7-13 Network #1 and Network #3 perform very well for these damage cases.

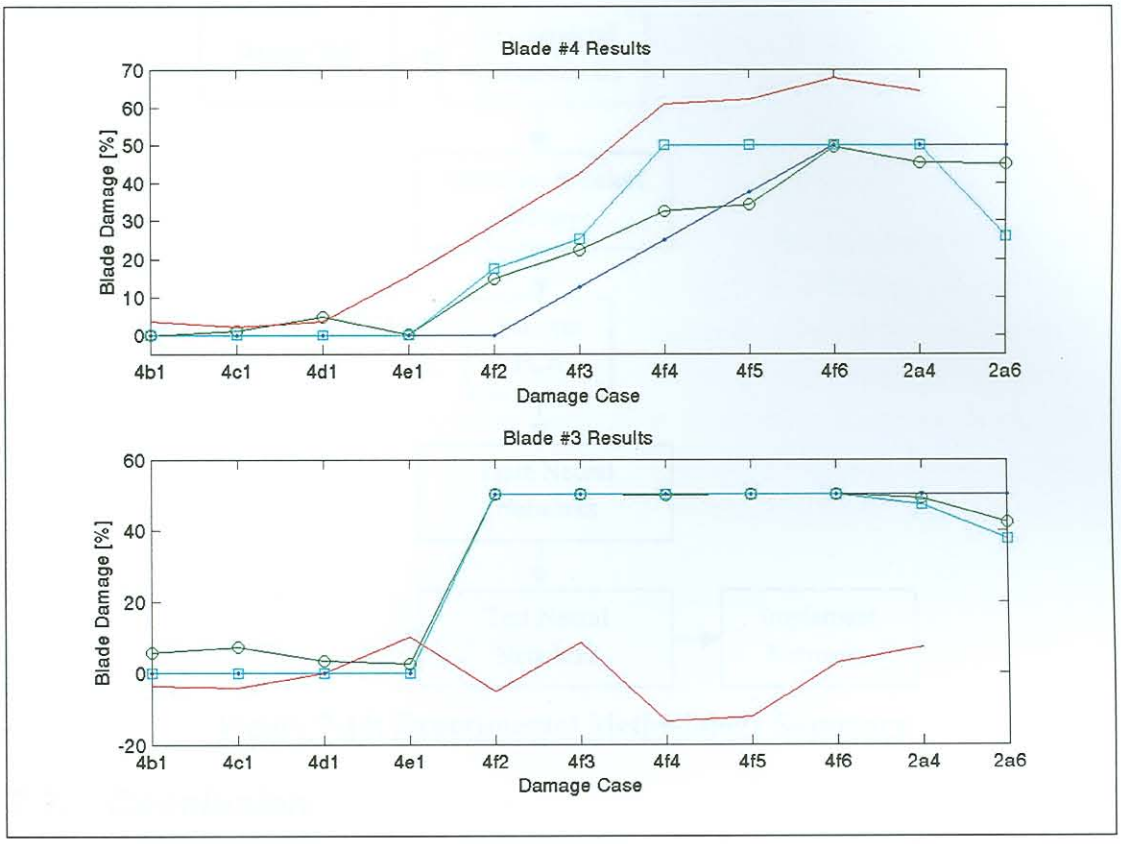

Figure 7-13: Results for Extra Datasets and Untrained Damage Cases

The measurements taken for blade #4f were basically a re-test for blade #4e as the strain gauge on blade #2 came loose during testing for blade #4e. Again, very good results are obtained from Network #1 and Network #3 for blade #3 damage detection. Less good results for Networks #1 and #3 are obtained for blade #4 damage detection. Network #2 performed poorly for the extra datasets as seen on Figure 7-13.

7.6. Experimental Methodology Summary

The experimentally supervised neural network damage detection methodology is summarized in Figure 7-14. The methodology consists mainly of a design, testing, network training and network implementation phase. As can be seen, this methodology is relatively simple.

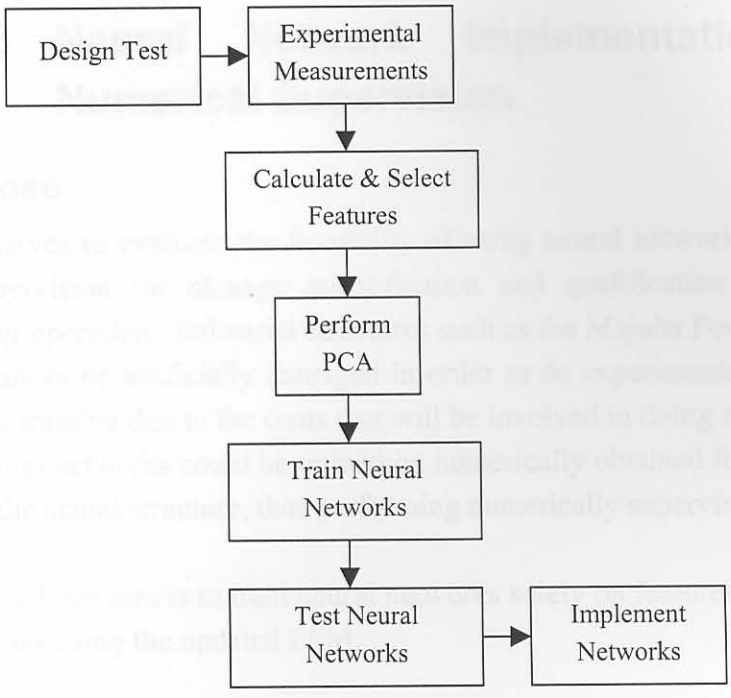

Figure 7-14: Experimental Methodology Summary

7.7. Conclusion

In this chapter, the ability of neural networks to do on-line blade damage quantification and qualification with data obtained from sensors not necessarily installed on the damaged blades, was proved for experimentally supervised training. Excellent results where obtained for all damage cases for which the networks were trained. However, the networks were found to be limited in terms of damage level interpolation and extrapolation. It was also found that a neural network trained for a specific sensor location, couldn't be used for a different sensor location.

Thus it can be concluded that in order to do accurate damage detection making use of neural networks, the networks must be trained for as many damage levels, damage scenarios and sensor locations as possible.