ERJ
Engineering Research Journal
Faculty of Engineering
Minoufiya University

CONDITION MONITORING AND FAULT DIAGNOSIS OF ROTATING MACHINERY USING WAVELET AND NEURAL NETWORKS APPROACHES

Dr. Yehia El-Mashad

Associate Professor, Shoubra Faculty of Engineering

Abstract

This research investigates different techniques of condition monitoring and fault diagnosis of rotating machines. These techniques are the classical Fast Fourier Transform (FFT), Discrete Wavelet Transform (DWT) coupled with different topologies of Neural Networks. A method for extracting features signal that is a combination of the horizontal and the vertical vibration time series is proposed. A technique for signal pre-processing for calculating the input feature is also adopted. The cumulants of magnitude of the vibrations provide a useful set of features for the detection of unbalance and rub faults. Pre-processing of the vibration signal is showed to amplify relevant spectral features improving the classification success.

Results based on the data collected with a simple test rig that allow the simulation of rub and/or unbalance fault(s) are presented. For Neural Networks the results show that the performance of Self-Organizing Map (SOM) gives higher classification rate than the Feed-Forward Neural Networks (FFNN). A compound Neural Network with wavelet has classified the correct condition in over 99% of cases.

في هذا البحث يتم فحص طرقا مختلفة لمراقبة حالة وتشخيص أعطال الماكينات الدوارة. تشمل هذه الطريقة تحويلات فور ير السريع والتموج الرقمي (Wavelet Transform) المرتبط بهياكل مختلفة للشبكات العصبية. وقد تم اقتراح طريقة للحصول على إشارة ملامح عبارة عن مزيج من القياسات الزمنية للاهتزازات الأفقية والرأسية. كما تم طرح أساليب معالجة لهذه الإشارة وذلك لتحسين أداء منظومة اكتشاف الأعطال. وقد تم الحصول على بيانات حقيقية من خلال منصة اختبار بسيطة تسمح بمحاكاة عيوب الحك وعدم الاتزان ولقد أوضحت النتائج أن كفاءة الشبكات ذات النوع (SOM) تعطي نتائج أفضل من (FFNN) كما أوضحت النتائج أن دمج الشبكات العصبية مع التموج أعطى تصنيفا صحيحا بنسبة نجاح تتعدى ٩٩% من الحالات المختبرة.

Keywords: Fault Detection - Condition Monitoring – Vibration Analysis – Neural Network – Wavelet Analysis.

1. Introduction

Vibration is probably the most important indicator of the mechanical integrity of rotating machinery. Overall vibration levels when trended, give immediate indication of change in the condition of that machine; however the overall vibration levels don't allow identification of any specific underlying fault type or types.

The vibrations of machinery are produced by a variety of effects including friction between moving parts, small unbalances in rotating components and component resonant frequencies. Damage and wear to components can affect any of these and therefore change the nature of the vibrations. The problem of identifying the correct condition is therefore that of isolating the change in the signal from the background vibration [1], [2], [3].

Since the measured signal is constantly changing, a single vibration sample does not, in isolation, provide useful condition information. To obtain an indicator of machine condition, it is necessary to transform a large number of vibration signal samples to produce time-invariant features. It is necessary to extract time invariant features from the time series of data for input to the classifier. The choice of features has a significant impact of the method success. How much useful information an individual feature can provide depends upon the signal being analyzed and therefore depends upon the machine and the faults it is likely to suffer.

By following the trends in such features, failure can be predicted. The decision as to when maintenance need to be applied can be made when the feature crosses a threshold between one condition and another. By setting an appropriate threshold, a single feature can be used to discriminate between two different machine conditions. When there are a variety of conditions and features, a more advanced discriminate analysis technique can be used depending on the distribution of the features for each condition. For more general classification methods

such as Multi-Layer Perceptron Neural Networks (MLPs), Wavelet Transform can be employed [4], [5], [6]. One of the main problems faced when using ANNs is how to select the optimal set of inputs for the ANN, allowing the creation of compact, highly accurate networks that require comparatively little pre-processing.

In pure frequency analysis, the temporal information is obscured. To overcome this problem short time windows can be used for the frequency analysis, but the main drawback here is that it is not possible to obtain high resolution in time and frequency simultaneously as shown in Figure 1. Wavelet analysis overcomes this limitation and is very useful for the analysis of non-stationary signals [7],[8],[9].

With recent advances in computational hardware many advanced signal processing algorithms can now be applied to vibration signals to extract features which are not dependent upon these assumptions. These algorithms may also provide features which are more tolerant of the noisy conditions which arise in industrial situations and therefore can be used in situations where traditional approaches fail. The objective of the work described in this paper is therefore to explore the use of advanced signal processing algorithms for enhancing the feature extraction process.

The contributions in this work believed to be original is the comparison of the performance of a range of classification algorithms, including approaches of Neural Networks using a wide range of extracted features based on Wavelet Transform coefficients for a feature signal that is a combination of horizontal and vertical vibration signals.

2. Experimental Test-Rig

Two machine fault conditions which commonly occur in turbine-generator plant are radial rub and change of rotor balance.

Vibration data were available from a small test rig which had been used to simulate these two conditions. The rig consisted of a rotating shaft with a flywheel driven by an electric motor held in place by a bearing block. A small weight could be attached to the flywheel putting the shaft out of balance. A threaded brass rod inserted in an empty bearing block is used to apply rubbing to the shaft. Vibrations were measured using two accelerometers mounted in the horizontal and vertical directions on a bearing block. A diagram of this machine rig is shown in Figure 2 where the vibrations are measured orthogonalty they give an indication of the position of the rotating shaft center in two dimensions.

For many faults, this can be plotted over time to produce a stationary pattern described as an orbit plot [10]. This allowed the creation of four different machine conditions:

- No faults applied.
- Only rub is applied.
- Only a weight is added.

Both rub and the weight are applied.

The signal form the accelerometer was passed through a low pass filter with cutoff frequency of 18.3 KHz, and was then sampled at a rate of 48 KHz. This operation was repeated 5 times for 4 different speeds. With a total of 4 different conditions, this gives a total data set of 80 case with 20 cases for each condition.

The effect of a rotating shaft being out-of-balance is to produce motion in an elliptical orbit. This elliptical orbit can be measured using the vibrations which are proportional to this motion. Plotting the vibrations over time shows the path of the orbit.

3. Signal Pre-processing and Calculation of Input Features

A possible solution for dealing with variable machine speeds is to reduce the resolution of the spectral estimate so that changes in frequency do not shift the frequency peaks from one FFT bin to another. The spectra of the four conditions of the weight and rub fault machine are shown in Figure 3. These show that the main effect of the rub fault is to increase the high frequency spectral components. The unbalance affects the signal at the rotation frequency, which is comparatively low.

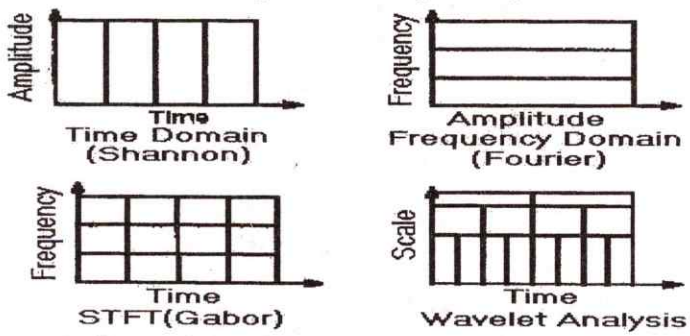

Figure 1: Comparison of Time-Frequency Resolution Obtained using STFT and Wavelet Transforms

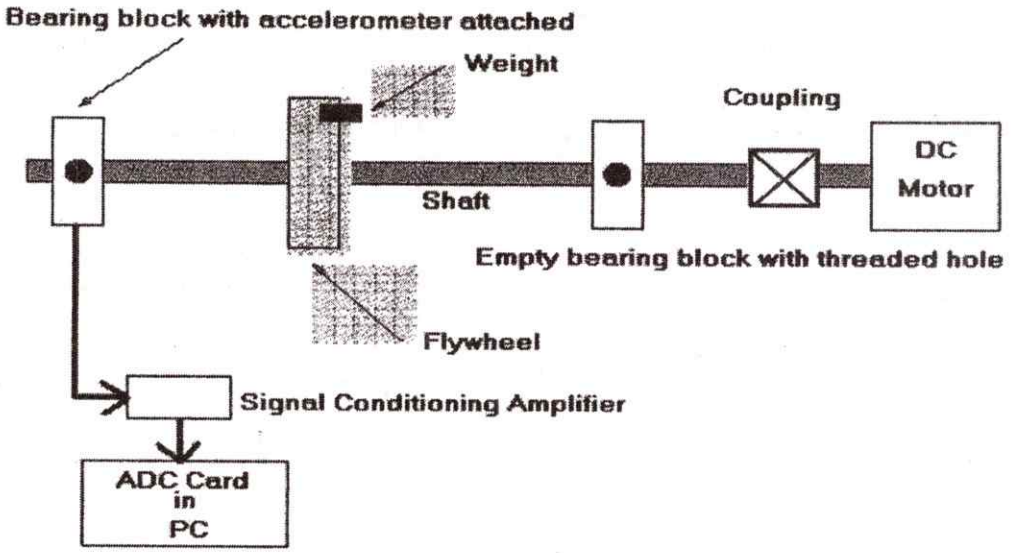

Figure 2: Diagram of Machine Test Rig

In the no-fault condition, the signal spectrum contains one fundamental spectral peak, unbalance increases the magnitude of this fundamental peak significantly. The presence of rub causes many spectral peaks to appear at harmonics of this fundamental frequency.

Four simple signal processing are considered: differentiation, integration, low-pass and high-pass filtering. Differentiation emphasis the high frequency part which arise in the rub signal. Integration emphasis the low frequency part and suppress the high frequency component leaving the signal largely composed of the fundamental frequency peak. A low-pass filter with cut-off frequency half way between the fundamental peak and the higher frequency component cuts out all the higher frequency component introduced by the rub effect making it easier to identify the unbalance if it is present. A high pass filter with the same cut of point removes the fundamental frequency peak leaving a signal with a significant amount of energy if the rub is present or a signal with almost no energy if the rub is not present.

4.1. The Power Spectrum

Spectral analysis considers the signal to be composed of periodic components. Many mechanical systems containing rotating parts and these can produce periodic components at harmonics of the rotation frequencies. Resonance of machine parts can also produce periodic signal components however the amplitude, frequency, and phase of these may vary with time. The development of a fault can affect the frequency structure of the signal by changing the magnitude or phase of a periodic component, or introducing new periodic components. Faults may also affect the resonant frequencies of machine parts thereby changing the frequency structure of the signal.

The standard method for determining the periodic structure of the signal is to estimate the power spectrum. The most direct way of computing this from a digital signal is by calculating the magnitude of the Discrete Fourier Transform (DFT).

This is limited to a window length of N samples and produces N frequency bin with spectral resolution of $\frac{f_s}{N}$ where f_s is the sampling frequency. The DFT can be computed efficiently using the Fast Fourier Transform algorithm (FFT). The effect of using a rectangular window of samples is to produce side lobes in the frequency domain function. By multiplying the signal section by a window function which tapers the ends of the data and removes the sharp cut-off, the side-lobes can be reduced improving the spectral estimate.

For signal, which have random components, it is desirable to estimate the mean value of the frequency content by averaging. Since the phase of periodic components will change between successive windows an average of the DFT's of the signals will average out to zero. The power spectrum of the signal is therefore computed as the average of the magnitude of the DFT of the signal sections.

4.2 Time-frequency and Time-Scale Distributions

The feature extraction method described so far has been based on the assumption that the signals are stationary. In many mechanical systems, the statistics and spectra of the vibrations will change over time. This non-stationarity can be caused by the development of a fault condition, a change in the

speed or loading of the system or even periodic changes within a single rotation cycle. In the third case the signal can be described as cyclostationary, and the cyclostationary nature of machine vibrations. The development of a fault may only affect the vibration for a short period of time within the cycle and therefore time-frequency techniques which measure the variation of spectral features can be used to locate the fault.

The simplest time-frequency decomposition of a signal is the short time **Fourier transform** (STFT). The Fourier Transform is estimated for successive short time intervals measuring the spectrum as it changes over time. The frequency resolution is restricted by the short-time intervals used in estimating the spectra.

In **Wavelet transforms** the signal is decomposed into orthogonal wavelets of the form;.

$$f(x) = \sum_{j=-\infty}^{\infty} \sum_{-\infty}^{\infty} c_{j,k} W(2^j x - k) \quad (1)$$

where $W(.)$ are wavelets. Wavelets are functions which have the property that they are orthogonal to versions of themselves dilated by a power of two.

$$\int_{-\infty}^{\infty} W(x) W(2^n x - m) dt = 0 \quad (2)$$

for $n \geq 0$ and $m \neq 0$ when $n = 0$. The sum of wavelets at scales of negative powers can be replaced by the dilation function $\phi(x)$ (which is also used in determining the Wavelet functions:

$$\sum_{j=-\infty}^{-1} \sum_{k=-\infty}^{\infty} c_{j,k} W(2^j x - k) = \sum_{k=-\infty}^{\infty} c_{\phi,k} \phi(x - k) \quad (3)$$

For practical implementation the wavelet transform is evaluated over a finite time interval, $0 \leq t < T$ which, is mapped onto the interval $0 \leq x < 1$. In this case the resulting signal decomposition can be expressed as [11]:

$$f(x) = a_0 + \sum_{j} \sum_{k=0}^{2^j - 1} a_{2^j + k} W(2^j x - k)$$

$$0 \leq x < 1, j \geq 0 \quad (4)$$

and the coefficients can be determined through the orthogonal properties of the wavelet functions:

$$a_0 = \int_0^1 f(x) \phi(x) dx \quad (5)$$

$$a_{2^j + k} = 2^j \int_0^1 f(x) W(2^j x - k) dx \quad (6)$$

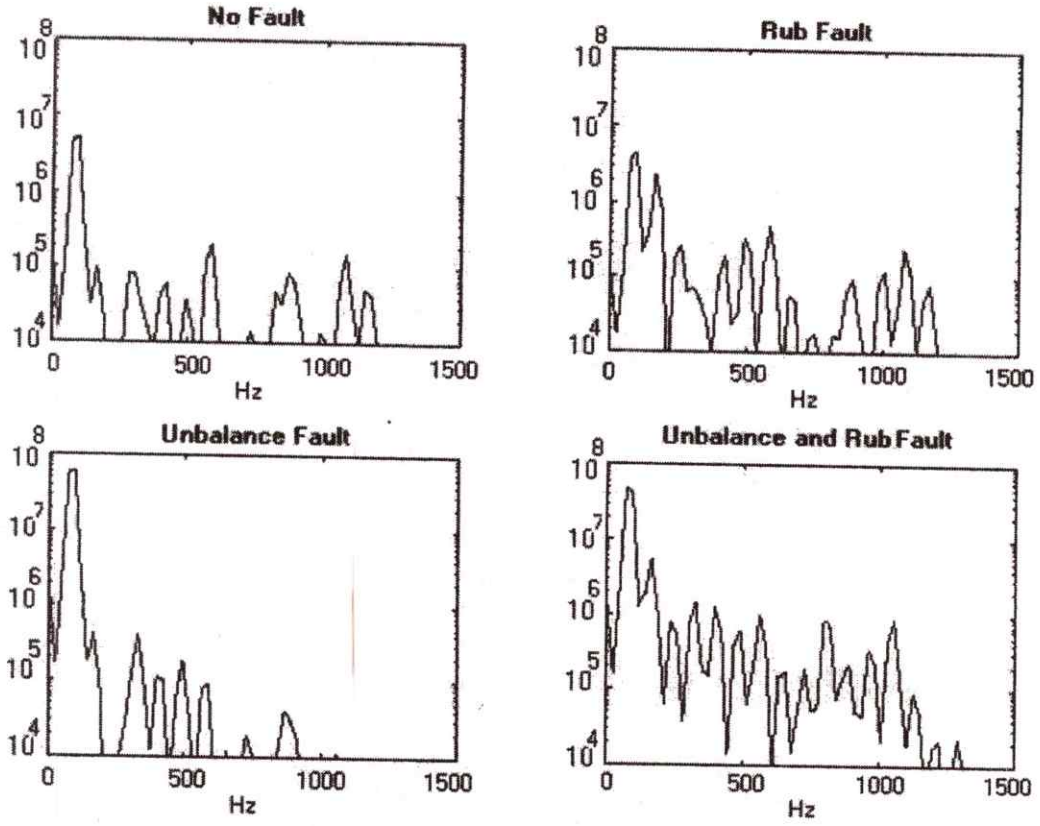

Figure 3: Spectra of Rub and Unbalance Fault Machine Vibrations

At low scales, the wavelets have good frequency resolution. At higher scales, there are more coefficients estimated improving the time resolution. This changing frequency resolution is illustrated in Figure 1 which illustrates the time-frequency resolution obtained with the STFT and with Wavelet Transforms for the same joint time-frequency resolution.

Matlab 6.5 wavelets toolbox was used for the decomposition which applies a recursive process of filtration and down-sampling to the signal till finally the signal length is reduced to a point.

Wavelets have been applied to many machine fault diagnosis problems including gear faults, electrical machine faults, automatic packing machines and machining processes[12], [13], [14].

5. Classification Algorithms Using Artificial Neural Networks

Feed-Forward Artificial Neural Networks (ANNs) as described in [15] have been used in a wide variety of pattern recognition applications including vibration monitoring as shown in [16].

Once a set of features which provide information about machine condition have been extracted it is necessary to use this information to estimate the condition of the machine. The decision function maps a range of many values in the feature space to one condition value. In a few cases, expert knowledge may be available to produce this decision; however in general the function will need to be learned from experimental data. The learning problem is to divide the feature space into volumes which represent each condition with decision boundaries placed along positions where two conditions are equally probable.

The fundamental building block of a neural network is an artificial neuron. The output, *y*, of this is a non-linear transformation of the scalar product of its weights, w and the input vector x(t) plus a bias, b:

$$y(t) = \phi(\sum_i w_i x_i(t) + b) \qquad (7)$$

The non-linearity $\phi(.)$ is usually a continuously differentiable approximation of a threshold function such as a hyperbolic tangent function or the logistics function.

While there are many possible ways of connecting neurons, the architecture most useful for implementing a classifier is the Multi-Layer Perceptron as shown in Figure4. This consists of two or more layers of neurons where the output of each layer is fed into the input of the next.

The weights are initially set to random values. Training patterns are presented to the network and the error between the actual output, y, and desired output, y_d, is measured.

Gradient descent is then used to provide the modifications to the neuron weights, which will reduce the error.

5.1 Feed-Forward Neural Networks

The feed-forward multi-layer perceptrons are networks with one or more layers of nodes between the input and output node layers. These additional layers contain neurons that are connected to all neurons in adjacent layers through links that are represented by synaptic weights. Selecting the number of nodes to be used in a multi-layer perceptron is an optimization problem[17].

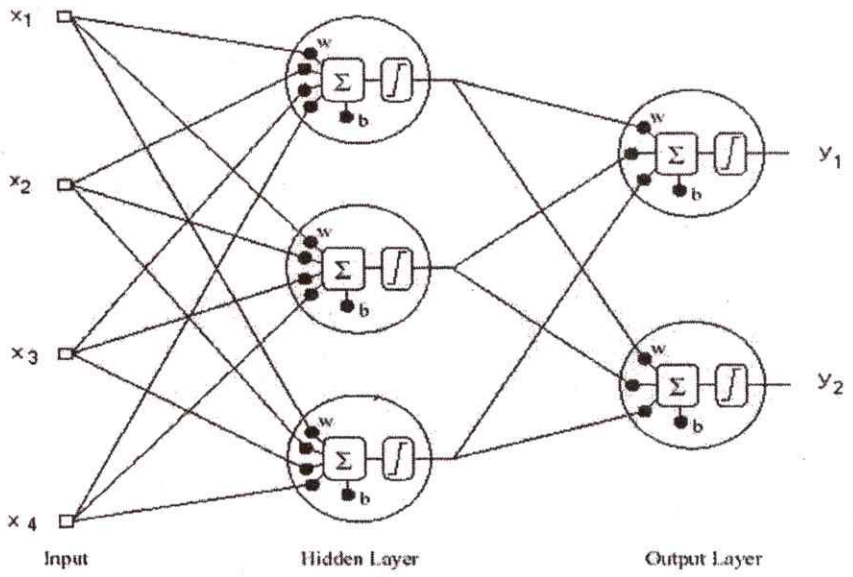

Figure 4: Schematic of Multi-Layer Perceptron ANN

The number of nodes must be large enough to form a decision region that is as complex as is required by the given problem. It had been shown that no more than three layers are required in perceptron like feed-forward networks because a three-layer network can generate arbitrarily complex decision regions. The MLP used in this work consists of one hidden layer and an output layer, the hidden layer having a logistic activation function, whilst the output layer uses a linear activation function. The size of hidden layer is determined by iteration. The size of the second layer is determined by the number of outputs required. This is a set of four neurons for the four simulated cases. Training the network is carried out using a standard back-propagation algorithm, and the network is trained for maximum 50000 epochs, using 40% of the data set as training data and the remaining 60% as the test and validation set.

The algorithm used to produce Feed-Forward neural network consists of the following five steps:

1. Initialize weights and offsets
Set all the weights and node offsets to small random values.

2. Present input and desired output
Present a continuous valued input vectors $x_0, x_1, \ldots x_{N-1}$ and specify the desired outputs $d_0, d_1, \ldots d_{M-1}$. If the network is used as a classifier, then all desired outputs are typically set to zero except for that corresponding to the class the input is from whose desired output is set to 1. The input could be new on each trial, or samples from a training set could be presented cyclically until weights stabilize.

3. Calculate the actual output
Use the sigmoid nonlinearity to calculate outputs y_0, $y_1 \ldots y_{M-1}$. The sigmoid function was used as it is differentiable everywhere.

4. Adapt weights
Use a recursive algorithm starting at the output nodes and working back to the first hidden layer. Adjust weights by:

$$w_{ij}(t+1) = w_{ij}(t) + \eta \delta_j x'_i \qquad (8)$$

In this equation,
$w_{ij}(t)$ is the weight from hidden node i or from an input to node j at time t.
x' is the output of node i or is an input.
η is a gain term.
δ_j is an error term for node j (local gradient).

If node j is an output node, then

$$\delta_j = y_j(1-y_j)(d_j - y_j) \qquad (9)$$

where d_j is the desired output of node j, and y_j is the actual output. If node j is an internal hidden node, then

$$\delta_j = x'_j(1-x'_j)\sum_k \delta_k w_{jk} \qquad (10)$$

where k is over all nodes in the layers above node j. internal node thresholds are adapted in a similar manner by assuming that they are connection weights on links from auxiliary constant-valued inputs. Convergence is sometimes faster if a momentum term is added and weight changes are smoothed by:

$$w_{ij}(t+1) = w_{ij}(t) + \eta \delta_j x'_i + \alpha\left[w_{ij}(t) - w_{ij}(t-1)\right]$$
$0<\alpha<1 \qquad (11)$

5. Repeat by going to step 2

5.2 Unsupervised Learning

Unsupervised Learning Neural Networks architectures have also been widely applied to condition monitoring problems [18]. These represent a very different approach to solving the condition monitoring problem, even though the resulting architecture of the classification system can be very similar. This is because knowledge of the class of the training data is no longer assumed. Data is clustered into groups with similar vectors which can then be related to machine conditions.

The most commonly applied unsupervised learning algorithm for artificial neural networks is the Kohonen Self Organizing Map. The basic structure for this is a two-dimensional map of neurons which compute the distance between the input feature vector and each neurons weights. This vector comparison operation means that this network is more closely related to methods such as vector quantization than to multi-layer perceptron neural networks, see Figure5.

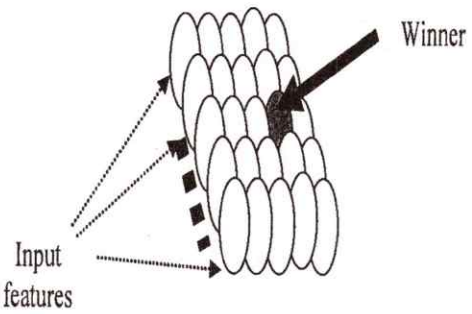

Figure 5: Network topology of the SOM

The algorithm used to produce Self-Organizing Feature Maps is as follows:

1. Initialize weights
Initialize weights from N inputs to M output nodes to small random values. Set the initial radius of the neighborhood.

2. Present new input
Let the input vector is $x_o, x_1, \ldots . x_{N-1}$.

Let the input vector is $x_o, x_1, \ldots . x_{N-1}$.

3. Compute distance to all nodes

Compute distance d_j between the input and each output node j using:

$$d_j = \sum_{j=0}^{N-1} (x_i(t) - w_{ij}(t))^2 \qquad (12)$$

where $x_i(t)$ is the input to node i at time t, and $w_{ij}(t)$ is the weight from input node i to output node j at time t.

4. Select output node with minimum distance

Select node j^* as that output node with minimum distance d_j

5. Update weights to node j^* and neighbors

Weights are updated for node j^* and all nodes in the neighborhood defined by $NE_{j^*}(t)$. New weights are given by:

$$w_{ij}(t+1) = w_{ij}(t) + \eta(t)(x_i(t) - w_{ij}(t)) \; j \in NE_{j^*}(t)$$
$$, 0 \le i \le N-1 \qquad (13)$$

6. Repeat by going to step 2

Appendix A explains the algorithm for implementing these methods.

These features, which indicate how closely the signal matches the reference data used to determine the model, can be used as features for trend analysis, if they are monitored over a long period of time. When several features are being monitored, it is necessary to have some method of deciding when a fault has occurred. This decision can be either when one feature indicates a fault, when all features indicate a fault or when a specific number of features indicate a fault. This would then allow discrimination between cases when all the features are indicating a significant likelihood of a fault and cases when only one feature is indicating a fault in isolation which would possibly be a false alarm.

6. Experimental Determination of Model Order and Network Structure

A simple two-layer neural network is sufficient for function approximation therefore only the number of neurons needs to be determined.

To investigate this, models were trained for a range of orders and structures using data from the rotating machine experimental rig. Initially only single speed data were used. Models were trained for four conditions using the first 1200 samples of horizontal vibration signals recorded from the machine rotating at 91Hz.

The ANN were trained and implemented with the Matlab Neural Network Toolbox [4] using the back-propagation with adaptive learning. Each input feature was scaled by an arbitrary value. The networks then had either one or two hidden layers. The output layer had four neurons and the target outputs for the network were chosen to be one for the correct class and zeros for all the other classes. For each choice of input features, networks were trained for a variety of different architecture of hidden nodes and the best architecture is chosen.

Data at each level of the wavelet decomposition was split into 3 complete sets consisting of a training set, validation set and test set. Several multiplayer perceptrons with 2 layers of weights, using the back-propagation algorithm, were trained. A variable learning rates from 0.1to 0.7 were simulated.

7. Results

7.1 Results of classification with Feed-Forward Neural Networks using time domain features

With one hidden layer, the best result was obtained using 32 neurons with 0.7 learning rate as shown in Table [1].

Table [1] Results for standard Feed-Forward ANN with one hidden layer.

Accuracy	No. of epochs	Learning rate	Hidden layer
0.9607	50000	0.1	5
0.9751	50000	0.3	5
0.9826	50000	0.5	5
0.9801	50000	0.7	5
0.9661	50000	0.1	10
0.9835	50000	0.3	10
0.9864	33756	0.5	10
0.9861	29487	0.7	10
0.9860	18028	0.1	32
0.9868	12054	0.3	32
0.9864	12054	0.5	32
0.9890	9819	0.7	32

With two hidden layers, the best result was obtained with 20 and 10 neurons in the first and second layer respectively as shown in Table [2].

7.2 Results of classification with Feed-Forward Neural Networks using Wavelet dilation features

The best results obtained when training the net with 16 neuron in the hidden layer and 0.5 learning rate with input feature extracted from the detail

coefficients at level 5 from the wavelet decomposition structure the of the original input signal. It reaches 99% of correct classification as shown in Table [3].

Table [2] results for standard Feed-Forward ANN with two hidden layers

Hidden layer		Learning rate	No. of epochs	Accuracy
1st hidden layer	2nd hidden layer			
10	10	0.3	50000	0.9861
10	10	0.5	38691	0.9863
10	10	0.7	22548	0.9870
20	10	0.7	18747	0.9863
20	10	0.5	23061	0.9892
20	10	0.7	20704	0.9863
20	20	0.3	29611	0.9865

Table [3] Results for standard Feed-Forward ANN with Wavelet

Hidden layer	Learning rate	No. of epochs	No. of Stages	Accuracy
8	0.5	50000	Stage 1	0.9735
8	0.5	50000	Stage 3	0.9773
8	0.5	50000	Stage 5	0.9855
16	0.5	19122	Stage 1	0.9865
16	0.5	32273	Stage 3	0.9865
16	0.5	15906	Stage 5	0.9906
32	0.5	17071	Stage 1	0.9864
32	0.5	16471	Stage 3	0.9867
32	0.5	26570	Stage 5	0.9867

7.3 Results of classification with Self Organizing Map Networks using Wavelet dilation features

A better result was obtained when training the net with 4 neurons with 0.5 learning rate as shown in the Table [4], it takes also the least time of training.

Table [4] Results for Self Organizing Map Networks with Wavelet

No. of neurons	Learning rate	No. of epochs	No. of Stages	Accuracy
4	0.1	500	Stage 1	0.9703
4	0.3	455	Stage 1	0.9801
4	0.5	100	Stage 1	0.9941
4	0.7	100	Stage 1	0.9861

8. Conclusion

Artificial Neural Networks provide a method for classifying the condition of rotating machinery if provided with input features which depend upon the faults under investigation.

The data used in this work was obtained from a small experimental test-rig that allows the simulation of three faults, unbalance only, rub only and both rub and unbalance faults in addition to the no-fault case.

The adoption of input feature that is the combination of the horizontal and vertical time series into a complex time series and then calculating the magnitude gives the distance by which the shaft is displaced from its center position. The performance of network trained using the feature selection was consistently higher than that trained without feature selection.

Two types of neural networks performing, as a classifier, with wavelet analysis in the pre-processing stage and without, have been investigated in this paper. The results of classification with Feed-Forward Neural Networks using time domain features shows that the best results obtained when using 32 neuron and 1 hidden layer with 0.7 learning rate and with 20 and 10 neurons when using two hidden layers with 0.5 learning rate

The results shows that by using Feed-Forward Neural Network the best results obtained when training the net with 16 neurons in the hidden layer and 0.5 learning rate with input feature extracted from the detail coefficients at level 5 from the wavelet decomposition structure of the original input signal. The success rate reaches 99% percent of correct classification. The results have been also improved when using Self-Organizing Map, as a classifier, and required fewer neurons and fewer numbers of epochs in the training process and takes less computation time.

The architecture of a feed-Forward ANN is not suitable for direct classification from the time series data as it has no memory of previous inputs. When using Artificial Neural Network it is needed to select the optimal set of inputs that allow the creation of compact, highly accurate networks that require comparatively little pre-processing. The results of this paper clarified the importance of the choice of the most relevant feature among the different measurements available for condition monitoring.

9. References

[1] J.T. Renwick, "Vibration Analysis- a proven technique as a predictive maintenance tool", IEEE Transactions on Industry Applications, vol. 21, pp. 324-332,1985.

[2] http://www.spd.eee.strath.ac.uk/, 'Machine Condition Monitoring'.

[3] W.T.W. Cory, "Overview Of Condition Monitoring With Emphasize On Industrial Fans", Pproceedings of the Industrial of Mechanical Engineering: Part A, vol.205, pp. 225-240,1991.

[4] R.M. stewart, "Application of signal processing techniques to machinery health monitoring",

Noise and Vibration", John Wiley and Sons, New York, 1986.

[5] A.C. McCormick and A.K, Nadi, "Rotating machine condition classification using artificial neural networks", Proceeding of COMADEM'96, held at the university of Sheffield, 16^{th} – 18^{th} July, 1996.

[6] J.R. Dickie, "An Investigation into Second and Higher-Order statistical Signal Processing Tool for Machine Condition Monitoring", Ph.D. thesis, University of Strathclye, 1994.

[7] The Mathworks Inc. Matlab Reference Guide (Neural Network and Wavelet Toolboxes).

[8] G.O. Chandroth, "Diagnostic Classifier Ensembles: Enforcing Diversity For Reliability In The Combination", Ph.D thesis, University of Sceffied, 1999.

[9] K. Bossley, R.Mckendrick, C.Harris, C.Merca, "Wavelet Packet Analysis in the Condition Monitoring of Rotating Machinery", Proceedings of AIRTC'98, IFAC International symposium on Artificial Intelligence in Real Time Control, 1998, USA [IFAC].

[10] A.C. McCormick and A.K. Nadi, "Cyclostationarity in Rotating Machine Vibrations", Mechanical systems and signal Processing, vol.12, March 1998.

[11] I., Daubechies, "Wavelets", Philadelphia: S.I.A.M., 1992.

[12] S. Vafaei, H. Rahnejat, "Indicated repeatable Runout with wavelet Decomposition For Effective Determination of bearing-induced vibration", Journal of Sound and Vibration, Feb. 2003.

[13] N. Aretakis, K. Mathiioudakis, "Wavelet Analysis for Gas Turbine Fault Diagnosis", Jornal of Engineering for Gas-Turbines and Power, vol. 119, Oct.1997.

[14] He. Zhengjia, Zi. Yanyang, Chen.Peng, Toyota, Toshio, "New Approach of Wavelet Fractal Analysis to Mechanical Fault Diagnosis", Proceedings of the ASME Design- Engineering-Technical Conference, Vol. 2001, pp 2905-2909.

[15] McCormick A.C and Nadi A.K, "Neural Network Autoregressive Modeling of vibration for Condition Monitoring of Rotating shaft", International Conference on Neural Network, Houston, Texas, June 1999.

[16] A.C. Mc Cormick and A.K. Nandi, "Classification of Rotating Machine Condition Using Artificial Neural Networks", Proceedings of the Institute of Mechanical Engineers, Part C, vol.11, No. 6, pp 439-450,1997.

[17] S. Haykin, "Neural Networks: A Comprehensive Foundation", Macmillan College Publishing Company, New York, 1994.

Appendix A

Algorithm for Machine Fault Classification using Wavelet and Neural Networks by using Matlab Toolbox

a. Feed-Forward

Step 1 to 4: load signals of the four conditions (NOFULT, RUBTUR, RUBWEIGH, WEIGH).

Step 5: Interact with user to ask for level of decomposition stage_no .

Step 6: Decompose the signal into stage_no. levels using db5 wavelet function of NOFULT signal.

Step 7: Extract the approximation coefficients at the stage_no level of NOFULT signal.

Step 8: Extract the dilation coefficients at the stage_no level of NOFULT signal.

Step 9 to 11: repetition of steps from 6 to 8 for RUBTUR signal.

Step 12 to 14: repetition of steps from 6 to 8 for RUBWEIGH signal.

Step 15 to 17: repetition of steps from 6 to 8 for WEIGH signal.

Step 18: compose the input signal extracted wavelet features to be the neural network input.

Step 19: Interact with user to ask for number of hidden neural.

Step 20: Create a feed-forward back-propagation network.

minmax(input_samples) :matrix of min and max values for input elements.

[hidden 4]: number of hidden and output layers neurons.

{'logsig','logsig'}: Transfer function of hidden and output layer

'traingd': Backprop network training function.

Step 21: constructing the output target matrix.

Step 22 to 25 adjusting the network training parameters

net.trainParam.goal=0.0001: adjusting the error goal to 0.0001

net.trainParam.lr = 0.5: adjusting the learning rate to any value such as 0.5.

Step 26: start network training.

Step 27 to 28: network output simulation with the input.

Matlab Commands

```
1.  load NOFULT;
2.  load RUBTUR;
3.  load RUBWEIGH;
4.  load WEIGH;
5.  stage_no=input('choose stage number') ;
6.  [NOFULT_c,NOFULT_l]=wavedec(NOFU
LT,stage_no,'db5');
7.  NOFULT_a3=appcoef(NOFULT_c,NOFUL
T_l,'db5',stage_no);
8.  NOFULT_d3=detcoef(NOFULT_c,NOFUL
T_l,stage_no);
```

```
9.  [RUBTUR_c,RUBTUR_l]=
wavedec(RUBTUR,stage_no,'db5');
10. RUBTUR_a3=appcoef(RUBTUR_c,RUBT
UR_l,'db5',stage_no);
11. RUBTUR_d3=detcoef(RUBTUR_c,RUBT
UR_l,stage_no);
12. [RUBWEIGH_c,RUBWEIGH_l]=wavedec(
RUBWEIGH,stage_no,'db5');
13. RUBWEIGH_a3=appcoef(RUBWEIGH_c,
RUBWEIGH_l,'db5',stage_no);
14. RUBWEIGH_d3=detcoef(RUBWEIGH_c,
RUBWEIGH_l,stage_no);
15. [WEIGH_c,WEIGH_l]=wavedec(WEIGH,s
tage_no,'db5');
16. WEIGH_a3=appcoef(WEIGH_c,WEIGH_l,'
db5',stage_no);
17. WEIGH_d3=detcoef(WEIGH_c,WEIGH_l,
stage_no)
18. input_samples = [NOFULT_d3
RUBTUR_d3 RUBWEIGH_d3 WEIGH_d3]
19. hidden=input('choose number of hidden
neurons');
20. net=newff(minmax(input_samples),[hidden
4],{'logsig','logsig'},'traingd');
21. t=[1 0 0 0 ; 0 1 · · ; · · · ;
    · · · ;]
22. net.trainParam.show=500;
23. net.trainParam.epochs=50000;
24. net.trainParam.goal=0.0001;
25. net.trainParam.lr=0.5;
26. net=train(net,input,t');
27. a2=sim(net,input)
28. a3=compet(a2)
```

b. Self Organizing Map

Step 1 to 18: the same as in the previous code.

Step 19: creating a self-organizing map neural network with 4 neuron and 0.5 learning rate.

Step 20: setting number of network training epochs.

Step 21: start network training.

Step 22 to 23: network output simulation with the input.

Matlab Commands

```
1.  load NOFULT;
2.  load RUBTUR;
3.  load RUBWEIGH;
4.  load WEIGH;
5.  stage_no =input ('choose stage number') ;
6.  [NOFULT_c,NOFULT_l]=wavedec(NOFU
LT,stage_no,'db5');
7.  NOFULT_a3=appcoef(NOFULT_c,NOFUL
T_l,'db5',stage_no);
8.  NOFULT_d3=detcoef(NOFULT_c,NOFUL
T_l,stage_no);
9.  [RUBTUR_c,RUBTUR_l]=wavedec(RUBT
UR,stage_no,'db5');
10. RUBTUR_a3=appcoef(RUBTUR_c,RUBT
UR_l,'db5',stage_no);
11. RUBTUR_d3=detcoef(RUBTUR_c,RUBT
UR_l,stage_no);
12. [RUBWEIGH_c,RUBWEIGH_l]=wavedec(
RUBWEIGH,stage_no,'db5');
13. RUBWEIGH_a3=appcoef(RUBWEIGH_c,
RUBWEIGH_l,'db5',stage_no);
14. RUBWEIGH_d3=detcoef(RUBWEIGH_c,
RUBWEIGH_l,stage_no);
15. [WEIGH_c,WEIGH_l]=wavedec(WEIGH,s
tage_no,'db5');
16. WEIGH_a3=appcoef(WEIGH_c,WEIGH_l,'
db5',stage_no);
17. WEIGH_d3=detcoef(WEIGH_c,WEIGH_l,
stage_no);
18. input_samples = [NOFULT_d3
RUBTUR_d3 RUBWEIGH_d3 WEIGH_d3];
19. net=newsom(minmax(input),4,0.5);
20. net.trainparam.epochs=100;
21. [net,tr]=train(net,input);
22. a=sim(net,input)
23. ac=vec2ind(a)
```