A Review of Vibration Signal Processing Techniques for use in a Real Time Condition Monitoring System

David Birch
Department of Electrical Engineering
University of Cape Town
Dave@dip1.ee.uct.ac.za

Submitted to the University of Cape Town in partial fulfillment of the requirements for the degree of Master of Science in Engineering.

Cape Town, May 1994

Declaration

I declare that this dissertation is my own unaided work. It is being submitted for the degree of Master of Science in Engineering at the University of Cape Town. It has not been submitted before for any degree or examination at this or any other university.

D.G. Birch

Synopsis

The analysis of the vibrations produced by roller bearings is one of the most widely used techniques in condition determination of rolling element bearings. This project forms part of an overall plan to gain experience in condition monitoring and produce a computer aided vibration monitoring system that would initially be applied to rolling element bearings, and then later to other machine components. The particular goal of this project is to study signal processing techniques that will be of use in this system. The general signal processing problems are as follows. The vibration of an undamaged bearing is characterised by a Gaussian distribution, and a white power spectral density. Once a bearing is damaged, the nature of the vibration changes, often with spikes or impulses present in the vibration signal. By detecting these impulses, a measure of the condition of the bearing may be obtained. The primary goal in machine condition determination then becomes the detection of these impulses, in the presence of noise and contaminating signals, and to discriminate between those caused by the component in question, and those from other sources. A wide range of signal processing techniques were reviewed, and some of these tested on vibrations recorded on the Mechanical engineering departments bearing test rig. It was found that the time domain statistics (RMS, kurtosis, crest factor) were the simplest to use, but could be unreliable. On the other hand, frequency domain analysis techniques, such as the power spectrum were more reliable, but more difficult to apply. By making use of a variety of these techniques and applying them in a systematic manner, it is possible to make an assessment of bearing condition under a wide variety of operating conditions. A small number of the signal processing techniques were programmed for a DSP processor. It was found that all of the techniques, with the exception of the bispectrum could be programmed for the DSP chip. It was found however that the available DSP card did not have sufficient memory to allow analysis and preprocessing routines to be combined. In addition to this, the analogue to digital conversion system would benefit from a buffered IO system. The project should continue, with the DSP card being upgraded and all the necessary signal processing routines programmed. The project can then move to the next phase,

which would be inclusion of display and interface software, and Artificial Intelligence analysis aids.

Acknowledgements

I would like to thank my supervisor, Mr Adrian Jongens, for his guidance.

I would also like to thank:

- Fred Hoare, for his assistance as system administrator of the workstations, upon which I was greatly dependent.
- Steven Schrire, Craig White and Etienne Kruger.
- My parents, for their continued support and encouragement.
- Cathy Jobson, for her editorial assistance.

Contents

List of Figures

List of Tables

Chapter 1

Introduction to Vibration Monitoring

1.1 Introduction

Condition monitoring has become a well established industrial practice. Most large industrial companies practice a range of techniques to establish the state of their plant and machinery, with vibration measurement and analysis an important feature in determining the condition of rolling element bearings. An ideal condition monitoring system would be one that :

- is easy to use.
- is reliable.
- does not require permanent machine monitoring.
- gives early warning of failure.
- can provide an accurate assessment of a machine's condition from a single short reading.
- does not require a large amount of information about the machine being monitored.
- works in real time.
- is portable.

This thesis proceeds with an overview of the condition monitoring process and various condition monitoring techniques. The vibrations of good and damaged bearings are discussed, followed by a review of vibration signal processing and signal enhancement techniques. A practical test of some of these techniques follows, using data recorded from bearings run on a bearing test rig. The difficulties encountered in a real time condition monitoring system are discussed, and some sample computer programs follow.

1.2 Objectives

The Central Acoustics Laboratory at UCT is researching condition monitoring techniques. We aim to achieve the assessment of machine condition and remaining machine life using a one–off measurement of the vibrations, without a need for detailed knowledge of the machine in question and with minimal reliance on data trending. While a great many techniques for condition monitoring exist, they each have their own strong points and weaknesses. It is hoped that by utilising a combination of these techniques, a system of analysis can be developed that will be suited to a wide range of conditions. Condition monitoring techniques can be developed for analysis of a variety of machine elements, for example gears, fans, rolling element bearings and journal bearings. In this way most rotating machinery could be assessed. By making use of fast portable computers and signal processing equipment, the necessary tests can be carried out on site, in the shortest possible time, to provide immediate answers on machine condition.

The objectives of this thesis are to review the signal processing techniques available, to determine their effectiveness, ease of computation and ease of use. Different methods of combining these techniques for greater effectiveness will also be investigated. An assessment of the suitability of these techniques for incorporation into a computer based system for the determination of the condition of rolling element bearings will be made, as well as an assessment of the hardware requirements.

1.3 Clarification of Concepts

These terms are used throughout this thesis. In order to make their use in this thesis clearer, they are defined below.

1. Condition Monitoring. Condition monitoring is a blanket term used to describe the measurement, recording and analysis of data concerning the behaviour of a machine.

2. Machine Condition Determination. Condition Determination or condition analysis is a subsection of condition monitoring, it is the estimation of the condition of a machine from measured data.

3. Residual Life Prediction. This is the estimation of how long a machine will remain in an acceptable running condition, given its current condition and the expected operating conditions.

1.4 Overview of Condition Monitoring Techniques

Various techniques are available to monitor the condition of a machine component. A short summary of some of the techniques used in monitoring rolling element bearings is presented

in this section.

1.4.1 Vibration Measurement

Vibration measurement has become the preferred technique for bearing, gear and rotor monitoring. The three vibration parameters of displacement, velocity and acceleration have all been used in the past, but velocity and acceleration measurement are used for most modern condition monitoring. Velocity measurement is generally used in industry, but to measure the higher frequency components of the vibration, acceleration is used.

1.4.2 Tribology

Tribology is the study of wear processes, and in this application it is specifically the analysis of particles present in the lubricant in order to gain information about the wear processes taking place. Particles are usually found in the lubricant of both healthy and damaged machines, it is the shape, size, composition and quantity of the particles that is important [1]. Rolling fatigue, as found in rolling element bearings, produces three types of particles: Fatigue spall particles, spherical particles and laminar particles. Spall particles are the pieces of metal that come loose when a spall pit appears. The presence of these indicates spalling and damage. Spherical particles are present in most rolling situations, and there may be an increase in these prior to spalling. Laminar particles are believed to be other wear particles that have been rolled, and are flat.

1.4.3 Temperature Monitoring

Temperature monitoring of rolling element bearings is simply the measurement of the surface temperature of the bearing housing. The temperature of a bearing does not provide as wide a range of information as other types of condition analysis. A bearing must be in an advanced state of damage before the friction increases enough to raise the temperature, thus it does not provide an early warning of damage. The running temperature of a bearing may also be affected by other factors, such as load, thickness of the bearing casing, and the length of time a machine has been running. Temperature monitoring is nevertheless a very simple technique and it can provide useful confirmation of other data, without much extra cost or time.

1.4.4 Implanted Proximity Probes

Proximity probes can be mounted in a hole drilled into a bearing housing and used to detect the flexing of the outer race as the rolling elements pass over it. This probe can then be used to detect unusual flexing, caused by damaged rolling elements or pitting of the race near the probe. This can be used as an alternative to vibration monitoring, especially in equipment that is permanently monitored. The probe senses movement of the race itself, and a degree

of immunity to other vibration sources is achieved. One drawback of this method is that the probe must be placed near the load zone of the bearing. This is not always possible in practice as this region is often inaccessible, and holes drilled here may weaken the machine.

1.4.5 Sound Intensity Measurements

Sound pressure and sound intensity can be used in bearing condition monitoring [2]. This can be of use in cases where the surface of the housing is not accessible or not suitable for accelerometer mounting. In most cases vibration measurement is preferred, as it provides greater immunity to sources of interference and its effectiveness has been proven.

1.5 Overview of the Condition Monitoring and Assessment Process

- Mechanical signal generation: Acoustic signals are generated by the interaction and movement of the bearing parts.
- Mechanical signal propagation through the structure: The vibration is modified due to structural resonances, nonlinearities and multipath effects. Variations in the signal path may also produce modulation of the vibration.
- Mechanical/Electrical transduction: The mechanical vibration is converted to an electrical signal using an accelerometer, velocity sensor or other appropriate transducer. The signal may be modified due to transducer resonance and bandwidth limitations.
- Digitising – The signal is digitised to make digital processing possible. This also allows storage and manipulation of the signal without further contamination.
- Noise reduction: Rejection of signals caused by something other than the required component is achieved by filtering, time or frequency domain averaging or adaptive noise cancelling.
- Signal Restoration: The signal may be whitened or inverse filtered to compensate for modification by the signal path.
- Feature Extraction: Features of the signal are extracted by spectral analysis, envelope detection, cepstrum or time domain statistics.
- Fault Classification: This follows a several step process, of deciding:
    1. Does a fault exist?
    2. What is the type and location of the fault?

3. How severe is the fault?

This may be achieved by comparing features with predetermined typical cases, comparing with a signal generating model, or comparison with and trending of previously recorded vibrations from the same machine.

- Residual life Estimation: Based on the information gained during the classification stage, and models of the failure process, the remaining life of the machine is estimated. Maintenance can then be scheduled using this information.

1.6 The Vibrations of Good Bearings

The vibrations of a normal bearing are often described as having a Gaussian acceleration distribution, with a velocity spectrum that is flat (some sources claim a flat acceleration spectrum). This is generally based on empirical evidence, with little other justification. The acceleration recordings of good bearings taken at UCT generally show a distribution that is near normal, with a spectrum that is approximately flat. In any situation, the shape of the spectrum will depend on the bandwidth under consideration.

1.7 Failure Mechanisms

Several different types of damage may occur in bearings. Among the ones listed in section E of [3] are:

1. Fatigue : The eventual failure of correctly fitted bearings is usually due to fatigue. The constant cycling load of the rolling elements causes fatigue cracks to appear in the metal of the balls and races. This appears on the surface as a crack or pit. Damage proceeds rapidly as the loose metal pieces spread over the bearing and cause localised overloading and further pitting.

2. Lubrication Failure : An inappropriate lubricant may fail to provide adequate lubrication under the running conditions of the bearing, even if there is an adequate quantity of the lubricant. This leads to increased wear and friction, and thus increased looseness and higher temperatures.

3. Lubrication Contamination. Contaminants in the the lubricants such as pieces of metal, or abrasive particles can accelerate the wear and fatigue of bearing parts, accelerating the onset of failure.

4. Brinelling: Extreme loads on a bearing, particularly while it is stationary cause plastic flow of the metal in the races (and possibly of the balls). These loads may be caused by:

- excessive force used in inserting a bearing.
- heavy static loads on the bearing.
- overheating of the inner race when fitting a bearing to a shaft, causing it to expand and force the rolling elements into the races.

5. False Brinelling: This is caused when a stationery machine is subject to vibration. Fretting corrosion occurs at the point of contact between the balls and the race. Small pits are thus formed on the balls and in the races. If the pits formed are severe they may cause the bearing to run very roughly and fail. Less severe pits may act as stress concentrations and then become centres for fatigue cracks.

6. Electrical Damage: Electrical faults or arc welding on a machine may cause large currents to flow through a bearing. This causes heating and possibly arcing at the point contacts between the balls and the races. The arcing creates pits in the balls and races or flutes in the cases of roller bearings.

7. Other possible causes of damage include corrosion, misalignment, out of roundness of the bearing housing, scratches on the races, inadequate lubrication and overheating.

1.8 Types of Bearing Damage and the Resulting Vibrations

The vibrations of damaged bearings differ depending on the nature of the damage, but in many cases have the form of a series of periodic ringing impulses [4, 5]. Detection of these types of bearing faults can then be accomplished by detecting these impulses using various techniques. The frequencies at which these impulses occur are related to the motion of the rolling elements. This motion can be described in terms of fundamental frequencies of bearing parts:

- The Fundamental Train Frequency, (FTF). This is the rate of rotation of the bearing cage.
- Ball Pass Frequency Outer, (BPFO) This is the rate that the balls pass over a fixed point on the outer race.
- BPFI, Ball Pass Frequency Inner, (BPFI). This is the rate at which the rolling elements make contact with a fixed point on the inner race.
- Ball Spall Frequency (or ball spin frequency), (BSF). This is the rate at which a point on a rolling element makes contact with the races.

These characteristic frequencies can be calculated using the following equations, which may be found in most texts on ball bearings [4, 6].

$$FTF = R \times \frac{1}{2} \times \left(1 - \frac{B}{P}\cos(\phi)\right) \tag{1.1}$$

$$BPFO = N \times FTF \tag{1.2}$$

$$BPFI = N - BPFO \tag{1.3}$$

$$BSF = R \times \frac{1}{2} \times \frac{P}{B} \times \left(1 - \left(\frac{B}{P}\right)^2\right) \tag{1.4}$$

R The shaft rotation frequency.

N The number of rolling elements.

P Pitch diameter. The distance between the centres of two balls on opposite sides of the bearing.

B Ball Diameter.

ϕ Contact Angle.

For analysis purposes, we can divide the types of damage into three main categories:

1. Outer race damage.
2. Inner race damage.
3. Rolling element damage.

The assumption has been made here, and in the following sections, that the outer race is stationary, and the inner race is rotating. The same ideas can be applied to the case of the rotating outer race and stationary inner race such as one might find in a motor car wheel bearing. Similar equations to the ones above can be developed for the this situation, as well as for rotating inner and outer races, and bearings with parallel rather than concentric races (Thrust bearings).

1.8.1 Outer Race Damage

In a good bearing, the outer race will flex as the balls pass over it. This may cause a slight spectral peak at BPFO. If the outer race is slightly warped the peak will be higher, but still contain few harmonics of BPFO. The shaft will move up and down slightly, depending on the position of the balls beneath it. This phenomenon, known as varying compliance, leads to vibration of the shaft and the bearing at the ball passage frequency. Thus in an undamaged bearing, BPFO may still be present. One of the simplest and more common types of defect

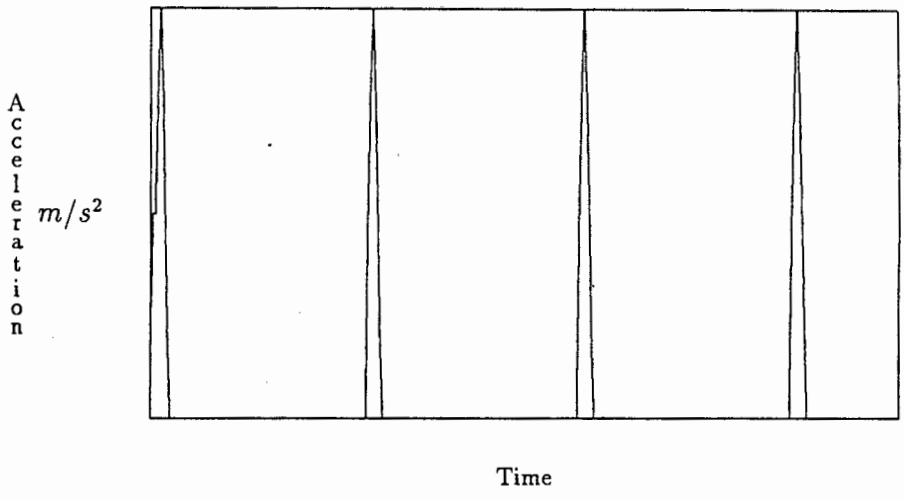


Figure 1.1: Series of Impulses

is a spall pit on the outer race. As the balls roll over this pit, impulses are generated at a frequency of BPFO as in Figure 1.1. The spectrum of this consists of peaks at BPFO and its harmonics (Figure 1.2). If there is imbalance in the machine shaft, then the radial force in the load zone will fluctuate, creating pulses that are modulated by the rotation frequency (figure 1.3). In this case the spectral peaks will have sidebands at the rotation frequency and its harmonics (1.4). Other forces acting on the shaft may also cause modulation of the signal.

1.8.2 Inner Race Damage

Damage on the inner race is less common than damage on the outer race. Due to the rotation of the shaft the load is taken by each point on the inner race in turn, and not concentrated in the load zone. While the vibrations from the inner race defects are similar in nature to those of the outer race, they must be transmitted through the rolling elements to the outer race where they can be sensed. This transmission path attenuates vibrations and varies with rotation. The vibrations are therefore of lower amplitude than those made by outer race defects, and strongly modulated by the rotational frequency, to the point where sidebands and second order sidebands may be visible. Modulation of the signal by BPFO may also occur.

1.8.3 Rolling Element Defects

This type of damage does not often occur, but may occasionally be found. A defect on a ball or roller will generate pulses as it makes contact with the inner and outer race. Due to the

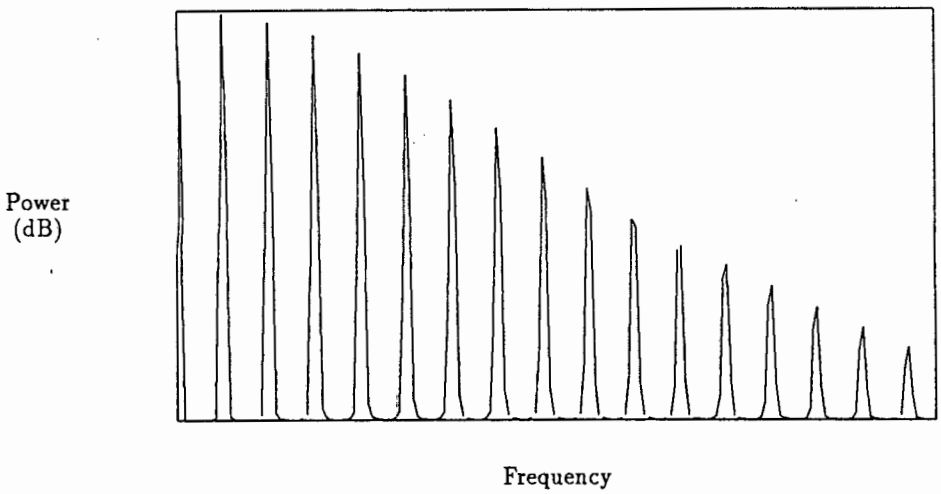


Figure 1.2: Spectrum of a Series of Impulses

transmission path, the signals from contact with the inner race will be attenuated more than those from contact with the outer race. This will emphasize the even harmonics of BSF. As the damaged ball moves in and out of the load zone, the impulses will be modulated by the cage rotation frequency, FTF. The resulting power spectrum has peaks at BSF and its harmonics, with sidebands at FTF. This signal may also be modulated by the rotational frequency, due to imbalance. This will create further sidebands in the power spectrum. When the damaged ball is not in the load zone, it will not rotate evenly, thus when it re-enters the load zone the phase of the peaks will have changed by some random amount, causing a blurring of the spectral peaks, which may make recognition of a damaged rolling element difficult. Another situation that may occur with ball bearings is that the ball may turn slightly, and the defect no longer comes into contact with the races. The defect appears to vanish temporarily, further frustrating detection of damage.

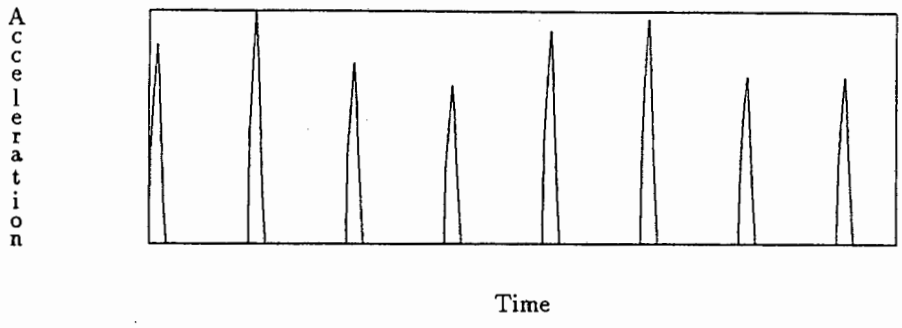


Figure 1.3: Modulated Impulses

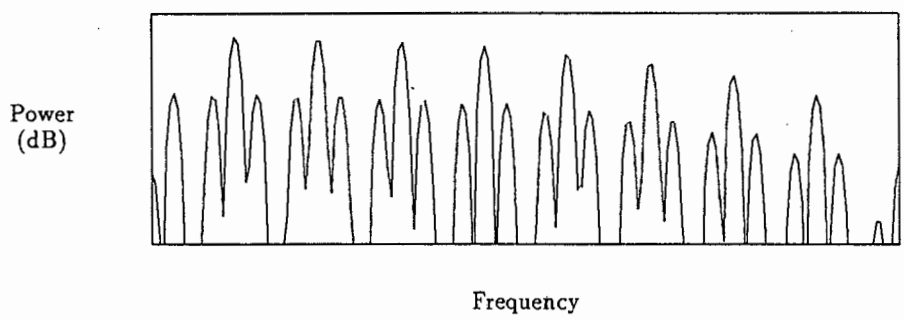


Figure 1.4: Spectrum of Modulated Impulses

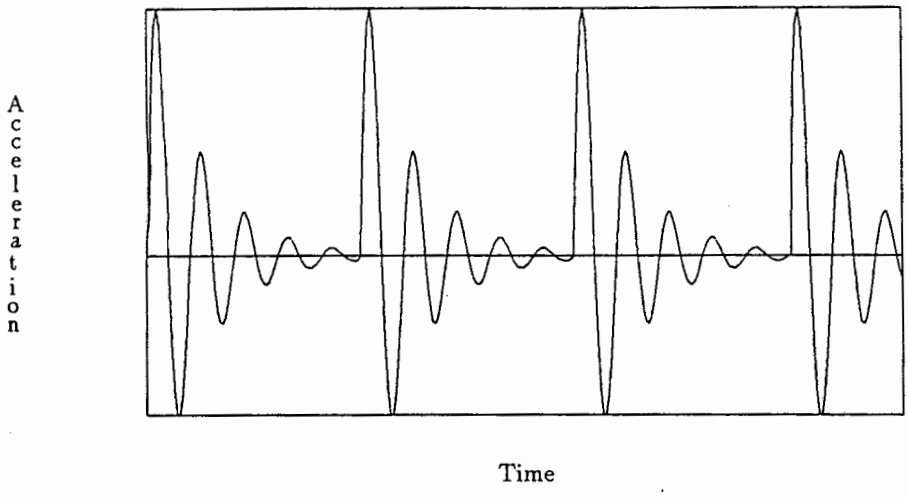


Figure 1.5: Ringing Impulses

1.9 The Effect of Resonances

The impulses generated by the defects will excite resonances in the bearing and housing. The signals will therefore not be pure impulses, but will ring at some higher frequency as in Figure 1.5. The effect of this is that the spectrum may show peaks in the lower part of the spectrum, at the passage frequency and its first few harmonics, as well as harmonic peaks higher up, around the resonant frequency as in Figure 1.6. Small variations in the time delay between impulses will tend to blurr the spectral peaks, particularly at the higher frequencies. The spectral peaks at the impact frequency can be enhanced by using the spectrum of the envelope of the vibration signal (Figures 1.7 and 1.8). It is possible to calculate the resonant frequencies of the balls and races (Appendix E). The bearing housings may however have other resonances which will modify the vibrations. These resonances may be difficult to predict, due to the complex shape of the bearing housing. For this reason, the resonance is not generally predictable by theory, and will differ from case to case.

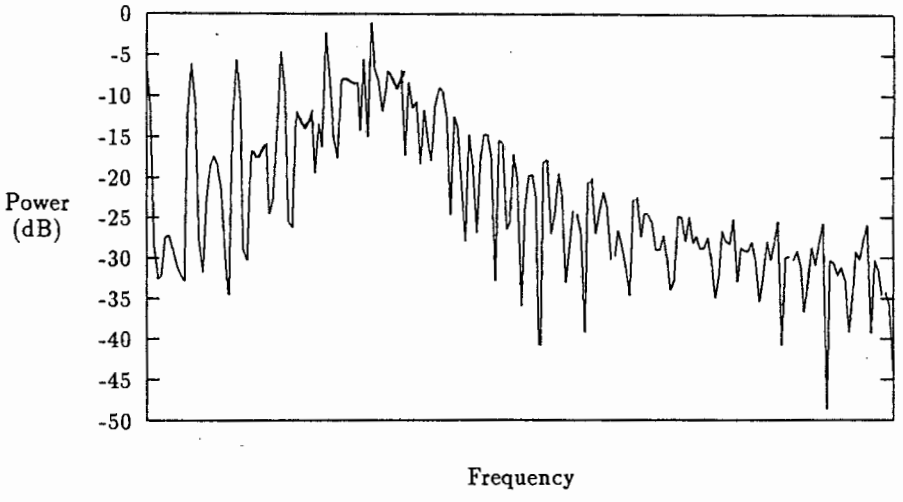


Figure 1.6: Spectrum of Ringing Impulses

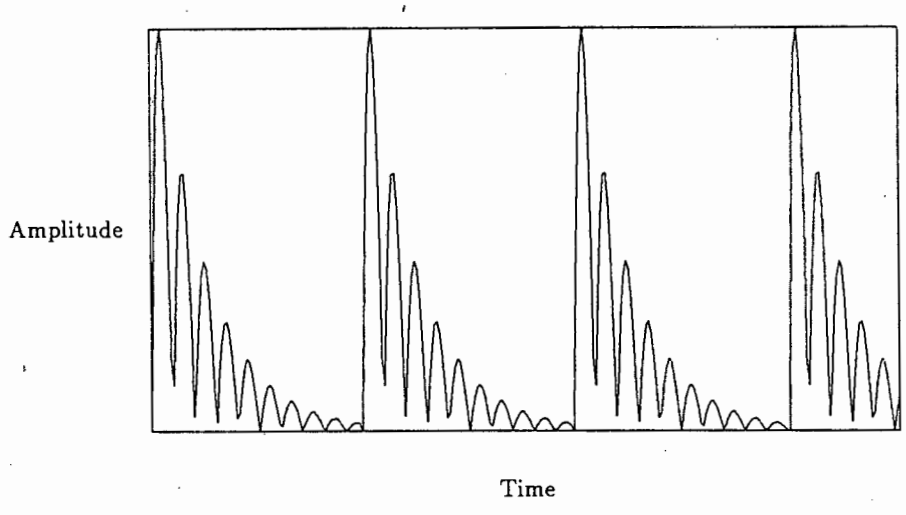


Figure 1.7: Rectified Impulses

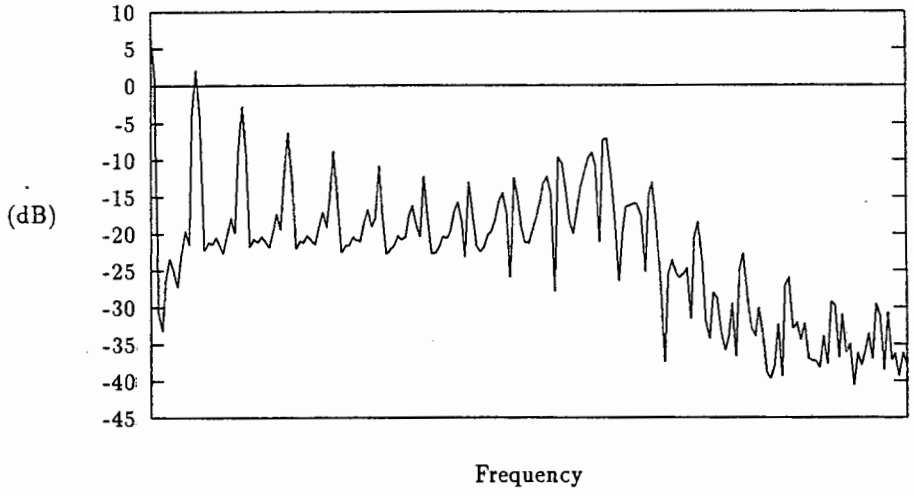


Figure 1.8: Spectrum of Rectified Impulses

1.10 Other Types of Defects

1.10.1 Multiple Defects

More than one defect may appear in a bearing. To a certain extent the superposition principle applies, and the two signals can be considered to be added together. If the defects are both of the same type, then destructive and constructive interference will occur. This will enhance some of the spectral peaks, and reduce others, depending on the spacing of the defects. This may cause the spectrum to be misleading, as harmonic relationships may no longer be obvious. In addition, certain analysis techniques such as kurtosis may be less sensitive in the case of multiple defects.

1.10.2 Defects Covering a Large Area

Once a race defect becomes large, approximately the diameter of a ball, then discrete impacts are no longer produced. Instead a large number of impacts occur, and seemingly random vibrations are produced. The amplitude of this vibration will depend on the load on the rolling elements that pass over the defects, which in most cases will fluctuate at the rotational frequency. The noise will be modulated by this rotational frequency, creating a signal that appears as bursts of noise. In theory, white noise modulated in this way would produce a spectrum that is flat, however in practice the rotational frequency appears as a peak and also as side bands around other frequencies. A possible reason is metal will be eroded from the load zone, increasing internal clearances in the bearing. This will allow more shaft movement, resulting in symptoms of mechanical looseness which are vibrations at the rotational frequency and its harmonics.

Chapter 2

Signal Processing Techniques

2.1 Trending Versus Single Measurements

Recording a single parameter and charting its behaviour over time is a long established method of determining the condition of a bearing, and by extrapolation of the data, predicting the remaining life. This does however require measurement of a machine on a regular basis. For a large number of machines this could be time consuming and costly. On the other hand we may attempt to establish machine condition from a single set of measurements. The most obvious problems with this technique are that we have no knowledge of what the measured parameters were when the machine was in good condition, thus we do not know if the condition has changed. Secondly, although we may be able to establish the condition of a machine, we do not know how long it took to reach that condition or the rate at which it is deteriorating, thus we cannot predict the remaining life. These problems can be approached in various ways. The lack of baseline data when trending is not used can be overcome by making use of the fact that the vibration parameters for a machine in good condition do not normally exceed a certain range. By checking to see that the parameters are within this range, we can establish whether or not the machine is in good condition. By checking the values of these vibration parameters against established norms, we can then determine the condition. Not all vibration parameters indicate damage for all its stages, thus a variety of parameters must be used. In addition some parameters are more reliable that others in indicating damage. Thus one must be aware of the limitations. In assessing the usefulness of vibration parameters, we must consider the following:

- Ease of measurement.
- Ease of calculation.
- What stages of damage does the parameter indicate?
- Is the parameter reliable or is it prone to missing signs of damage or giving false alarms?

Without the ability to trend the data, and establish the rate of deterioration, we must rely on knowledge of the machine and the normal life of equipment under those conditions to predict remaining life.

2.2 Time Domain Techniques

The time domain techniques of condition analysis, although not the most reliable or informative, do have several advantages:

- They are computationally efficient, as all the techniques mentioned here require of the order of N calculations (N is the number of samples, usually approximately 1000), and are therefore very fast.
- They do not require large amounts of memory to compute, as there are very few temporary variables needed, and the calculations can be performed as the data is captured, eliminating the need for storage of all N points.
- They reduce the volume of data considerably, usually from N values to one. This simplifies trending of vibration parameters.
- They produce single number statistics, which although not always very reliable, are simple to interpret.

2.2.1 RMS Vibration Measurement

Measurement of RMS vibration is one of the simplest techniques of vibration analysis and is therefore in widespread use. As the damage in a bearing increases, so the RMS vibration level rises, with a tenfold increase in vibration level suggested as an indicator of advanced damage [7]. This technique has limitations, as the RMS vibration is dependent on load, speed, mounting stiffness and bearing type. Advanced damage is necessary before the RMS vibration shows a definite change, making it an ineffective indicator of early damage [8]. A measurement of the machine in good condition is necessary for comparison purposes. If this is not available, various guidelines for acceptable levels of vibration have been suggested. These may not always be accurate, as the stiffness of the machine mounts has a strong influence on the RMS vibration level [9]. Measurement of vibration force would be a better technique [10], but lacks the simplicity of other vibration based monitoring techniques. As an aside, RMS measurement is a statistical measurement, as the RMS value of an AC signal is equivalent to its standard deviation.

2.2.2 Kurtosis

Kurtosis is the fourth moment about the mean, normalised by dividing by RMS squared. It is often referred to as a measure of the peakedness of a signal. Other explanations exist, and attempts have been made to find kurtosis estimators that measure a particular aspect of a signal [11]. In condition monitoring the standard definition is normally used. For a continuous zero mean signal , $F(t)$ with probability distribution $p(x)$ the kurtosis is defined as:

$$\kappa(F) = \frac{\int_{-\infty}^{\infty} x^4 p(x) dx}{\left(\int_{-\infty}^{\infty} x^2 p(x) dx\right)^2} \tag{2.1}$$

or:

$$\kappa(F) = \frac{\int_{t_1}^{t_2} F(t)^4 dt}{\left(\int_{t_1}^{t_2} F(t)^2 dt\right)^2} \times (t_1 - t_2) \tag{2.2}$$

Where t_1 to t_2 is a time interval long enough to obtain a statistically valid sample.
The equivalent discrete time domain formulation is:

$$\kappa(F) = N \times \frac{\sum_0^{N-1} F_n^4}{\left(\sum_0^{N-1} F_n^2\right)^2} \tag{2.3}$$

The use of kurtosis as a measure of bearing damage was first proposed by Dyer and Stewart [12]. In general, if the kurtosis of the acceleration signal departs from a value of 3 (Kurtosis of Gaussian noise) it is taken as an indication of damage. This is essentially an empirical threshold, based on the kurtosis of white noise, and is mentioned in many texts [12, 8]. In the method discussed in the original paper, the acceleration signal was band pass filtered into four bands: 3Hz to 5kHz, 5 to 10 kHz, 10 to 15 kHz and 15 to 20 kHz. The kurtosis in these bands was then used as a type of feature vector to estimate bearing condition. The value of kurtosis as an early indicator of damage is shown in Figure 5 of [12], where it can be seen that the kurtosis value of 6 indicates damage, but this is not yet visible on the RMS graph. Dyer also shows the independence of kurtosis from speed and load. For a good bearing with loads in the 0 to 11kN range, and speed in the range 800 – 2700 RPM, kurtosis was within +– 8 % of 3, whereas RMS acceleration varied by +– 50 % and peak acceleration by +– 65 %.

From the last entry of the table 2.1 it can be seen that a signal consisting of impulses narrow in relation to the spacing between them will have a high kurtosis value. This is the situation for a bearing in the early stages of damage. As the defects become larger and more numerous, the impulses will become wider and more frequent, resulting in a lower kurtosis value, that may be close to that of a good bearing. With this in mind, kurtosis is then of most use in detecting the early stages of damage. Another point to note is that a sine wave has a very low kurtosis value. Any signal that has a dominant sine component will also have

Waveform	*Kurtosis*
Gaussian noise	3
Evenly distributed noise, triangular and sawtooth waves	1.8
Sine Wave	1.5
Square wave	1.0
Symmetrical +− peaks, width τ, period T	$\frac{T}{2\tau}$

Table 2.1: Kurtosis of Selected Waveforms

a low kurtosis value, despite the presence of peaks in the signal. The test results in Chapter 3 show that resonance can have a substantial effect on kurtosis. The impulses ring, and are thus broader and in an extreme case could be almost sinusoidal. This is discussed further in Chapter 3.

2.2.3 Spike Energy Measurement

The block diagram (Figure 2.1) for a Spike energy meter is suggested in [13]. The high pass filter, combined with the natural resonance of the accelerometer forms a filter that passes frequencies from approximately 12 to 50 kHz. This filter removes the interference of normal machine vibrations, leaving only the high frequency energy present in the short impulses caused by early bearing damage. The rest of the system then records the energy present in these impulses. This system is used in a variety of commercial bearing test systems. As with other single number statistics, a comparison value for an acceptable bearing is needed. This is usually provided by the manufacturer of the meter, for that particular type of meter.

2.2.4 Spike Counting or Probability of Excedence Measurement

A signal with a normal distribution will, if measured over a period of time, exceed a given value a fixed percentage of the time. The percentage of time that a signal exceeds this given value can be used as a measure of the peakedness of the signal. To use this as as measure of bearing damage merely requires choosing a suitable value of the threshold. In [14] the threshold values of 2.5 or 3.0 times the variance are suggested. From the tables of [15] the probabilities of a Gaussian signal exceeding these thresholds are 1.24 and 0.26 percent respectively. The calculations for this measurement can be performed digitally with order N calculations, or using an analogue system. A related approach is to find the threshold that the signal exceeds a fixed percentage of the time. This will require considerably more calculations. For example, one possible method is to calculate the cumulative distribution of the signal. From this the required threshold can be determined. An alternative is to use an iterative technique, where a threshold is chosen, the percentage samples exceeding this

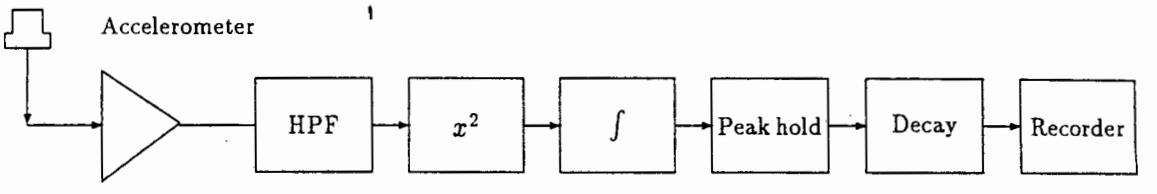


Figure 2.1: Spike Energy Measuring System

calculated, and from this a new threshold approximated. This has the advantage of requiring less data storage capability. This process can also be performed using an analogue system as before, with an additional feedback loop to vary the threshold.

2.2.5 Peak Vibration Levels and Crest Factor

The peak vibration levels are another statistic that can be used in a similar manner to the others. The peak readings may be normalised by dividing them by the RMS. The result of this is a parameter known as crest factor. The crest factor of a signal tends to follow the same patterns as kurtosis, but measurement of crest factor can be accomplished using simpler devices than are needed for measuring kurtosis. A meter that can measure true RMS and peak values is all that is needed. As with kurtosis, values of crest factor for various types of signals can be calculated from the definition.

One point to note is that using this formulation for crest factor, it is not possible to calculate a value for the crest factor of Gaussian white noise, and several other similar types of function. Since a Gaussian distribution has tails that extend to $\pm$ infinity, the crest factor of Gaussian noise calculated for an infinite time span tends to infinity. For a non-infinite time span, the crest factor of Gaussian white noise can be expressed as a probability of the crest factor being within a certain range. While this may make calculation of theoretical crest factor values difficult, real world signals with infinite peaks do not exist, thus crest factor is still a usable measure, with Table 2.2 providing a basis for estimation of bearing condition. Crest factor can be very sensitive to single peaks, but ways to reduce this have been proposed, such as recording several peaks and using the median value of these peaks.

Waveform	Crest factor
Evenly distributed noise, triangular and sawtooth waves	$\sqrt{3}$
Sine Waves	$\sqrt{2}$
Square waves	1.0
Symmetrical $\pm$ peaks, width τ, period T	$\sqrt{\frac{T}{2\tau}}$

Table 2.2: Crest Factor of Selected Waveforms

2.2.6 Histograms or Probability Distribution

Kurtosis, Skewness, RMS and Crest factor all give some information about the distribution of the vibrations. There are however techniques for examining the probability distribution in a more direct manner. One way of interest when dealing with normal distribution is to take the log of the probability function. This converts a Gaussian distribution to a parabola, as shown in the equations below. This curve can be visually compared to a parabola, or mathematical curve fitting techniques can be utilised. Alternatively, standard statistical tests may be applied (chi squared etc).

$$\begin{aligned} p(x) &= e^{-x^2} \\ f(x) &= log\,(p(x)) \\ f(x) &= -x^2 \end{aligned}$$

2.3 Frequency Domain Techniques

2.3.1 Spectral Alarm Bands

The use of spectral alarm bands is a refinement of the RMS vibration monitoring process. There are two variations of the technique. In both cases the power spectrum is divided into a series of bands, usually between four and ten bands are used. In the first method, damage is considered to exist if the total power in a specified band exceeds the threshold set for that band. In the second variation, if any single frequency in a band exceeds the threshold then it is regarded as a warning of damage. Provided the alarm levels for each band are set correctly, this technique is more sensitive to damage than a simple RMS measurement. It can also be adjusted to individual machines, techniques of doing so are discussed in [16]. This technique retains the simplicity of the time domain techniques, and does not require knowledge of the bearings and machine speeds. Once the band alarm levels are set correctly it can be applied automatically, and condition monitoring software often incorporates this facility. The two

main drawbacks to this technique are that no diagnostic information is available, and that false warnings may be generated by sources of interference.

2.3.2 The Autocorrelation Function

The Autocorrelation function is essentially a time domain calculation, but it is used in a similar manner to some of the frequency domain techniques and for this reason has been grouped with them. The autocorrelation function is the correlation or convolution of a signal with itself, and can be used to find periodicity in a signal. This can also be performed by zero padding the time domain signal, Fourier transforming, squaring the magnitude and inverse transforming. This is similar to the computation of the cepstrum and the cepstrum is becoming the preferred technique. The log process used in calculating the cepstrum enhances small cepstral components as well as making deconvolution possible using the cepstrum.

2.3.3 Detection of Resonances

The vibrations caused by bearing defects may excite resonances in the bearing, as in Figure 1.5. If these resonant frequencies are known, or can be calculated from bearing dimensions as in Appendix E, then by examining the spectrum for a broad resonance peak at the resonant frequency, the presence of defects can be determined. This is the method described by [17]. In this case the resonant frequency observed was the outer race resonant frequency, which was the lowest resonance in the balls and races. For most bearings the ball resonant frequency is out of the frequency range of most recording equipment ($>$ 50 kHz). Which leaves the resonance of the races and the bearing housing. Due to its complex shape, the housing resonance is difficult to calculate, and this reduces the usefulness of this technique.

2.3.4 Power Spectral Analyses of Repetition Frequencies

The power spectrum of the vibration can be examined for spectral peaks at the defect frequencies mentioned in Chapter 1. The spectrum will seldom show simple peaks; intermodulation of the various frequencies will produce sidebands, or sum and difference frequencies as they are sometimes known. Analysis of these frequencies can help pinpoint the type of damage [4, 18]. This is a fairly well known and effective method, but like all the frequency domain techniques requires prior knowledge of the machine rotational speed and the bearing dimensions. Modern computer databases can be used to hold records of the dimensions of various types of bearings and the rotation speeds of the equipment, to be monitored, simplifying the task. The other drawback to this method is that the narrow time domain peaks caused by small defects contain very little power, and this power may spread over a large number of harmonics, thus no visible peaks appear in the spectrum.

2.3.5 Envelope Analysis

The generalised technique for envelope analysis [5] (or amplitude demodulation as it is sometimes known) is a three step process. The signal is band pass filtered, rectified and then the FFT is taken. The analysis then follows the same process of looking for the spectral peaks that is used in power spectrum analyses. This technique relies on the fact that the impulses from a damaged bearing contain a lot of high frequency harmonic energy, and ring at some high frequency (see fig 1.5 to 1.8). By filtering out the low frequency interference caused by other machines, imbalance or misalignment and then rectifying or demodulating the resulting signal, the spectral peaks caused by the damaged bearing can be enhanced. In the past the demodulation was performed before the signal was digitised, but now it and the filtering may be performed digitally. Another similar technique that has been suggested [8] is to raise the filtered signal to the fourth power, and then find the power spectrum. This serves two purposes, the fourth power of the signal has only positive values, so rectification is not needed. The peaks and transients of the signal are enhanced, making it easier to distinguish the harmonics caused by impulses from the single frequencies found in normal bearings.

2.3.6 Auto Transpectral Analysis

The power spectrum makes no use of the phase information in the Fourier transform, the coherence of the components of a signal. From this coherence we may gain insight as to whether two signals are derived from the same source, or one from the other or if they are totally independent of each other [19]. Consider two signals $x(t)$ and $y(t)$, divided into n segments 1 ...n:

$$x_m(t) = cos(2\pi ft + \phi_m) \tag{2.4}$$

$$y_m(t) = cos(2\pi ft + \theta_m) \tag{2.5}$$

The phase difference for any two segments in this set is:

$$\alpha_m = \phi_m - \theta_m \tag{2.6}$$

The coherence is then defined as:

$$\mathcal{J} = \sqrt{{C_1}^2 + {C_2}^2} \tag{2.7}$$

Where:

$$C_1 = \sum_{m=1}^{n} cos(\alpha_m)$$

$$C_2 = \sum_{m=1}^{n} sin(\alpha_m)$$

This concept can be extended to signals at different frequencies ω_1 and ω_2 by defining the transphase, τ of these two signals as:

$$\tau = l_1 \left(\phi - \frac{\omega_1}{\omega_2} \theta \right) \tag{2.8}$$

Where l_1 is an arbitrary constant. The coherence can then be calculated as before using this new transphase. This also allows the coherence between any two Fourier components of a signal to be calculated. This coherence, or auto coherence can be plotted as a two dimensional function $\mathcal{J}(\omega_1, \omega_2)$ or as a single dimensional function of coherence between frequencies and a given multiple of that frequency.

2.3.7 Higher Order Spectra - The Bispectrum

The bispectrum is one of a family of spectra, consisting of the power spectrum, the bispectrum, the trispectrum etc. The bispectrum is defined in [20] as:

$$B(\omega_1, \omega_2) = \sum_{\tau=-\infty}^{\infty} \sum_{\tau=-\infty}^{\infty} c_3(\tau_1, \tau_2) e^{-j(\omega_1 \tau_1 + \omega_2 \tau_2)} \tag{2.9}$$

Where C_3 is the third order cumulant series. For the discrete bispectrum, using K data segment of M points each, C_3 can be defined as follows:

$$C_3(m, n) = \frac{1}{M} \sum_{l=s_1}^{s_2} x(l)\, x(l+m)\, x(l+n) \tag{2.10}$$

Where

s_1 max (0, -n, -m)

s_2 min (M-1, M-1-m, M-1-n)

A method of calculating the bispectrum from DFT coefficients is given in [20], the details of which will not be discussed here. The bispectrum is used in a similar way to the transpectrum, namely to determine coherence. If we perform the calculation:

$$b(\omega_1, \omega_2) = \frac{B(\omega_1, \omega_2)}{P(\omega_1)\, P(\omega_2)\, P(\omega_1 + \omega_2)} \tag{2.11}$$

Where $P(\omega)$ is the power spectrum value, we obtain the the bicoherence function. This is a measure of the coherence between three frequencies, ω_2 ω_2 and $\omega_1 + \omega_2$. This gives an indication of the phase coupling between these frequencies. If the component at $\omega_1 + \omega_2$ is a result of modulation of ω_1 and ω_2 then the bicoherence of these two frequencies will be close

to 1. Regarding the use of the bispectrum in bearing condition determination, two references to papers on this subject were found. One of the papers was in Japanese, the second could not be obtained. The abstract of this second paper, by Li, Hwang and Nikerson reads:

Pattern recognition based bicoherence analysis for bearing condition monitoring. *For automatic detection/diagnosis of localised defects in rolling element bearings, a pattern recognition analysis scheme based on features extracted from bearing vibrations using bispectral analysis is developed. Features are extracted using the bicoherence spectrum to detect the sum frequency components of bearing characteristic frequencies and their harmonics. Employing these features, a linear discriminant classifier has been established to detect localised defects on a roller and outer race of a bearing. Experimental results show that the bispectral analysis for the detection of localised defects of rolling element bearings is effective even when power spectral analysis fails to distinguish the abnormal states from the normal one.*

While this suggests that this should be further investigated, one should bear in mind that computation of the bispectrum requires of the order of $k \times N^2$ calculations, where k might be in the range of 5 to 10, and N in the region of 400. This would put calculation out of the capabilities of many simple DSP systems.

2.3.8 Cepstral Analysis

The cepstrum of a signal is defined as the inverse Fourier transform of the log of the Fourier transform of the signal. This allows two classes of cepstrum, the real or power cepstrum, which is the inverse transform of the log of the square of the magnitude of the Fourier transform, and the complex cepstrum, which is the inverse transform of the complex log of the Fourier transform coefficients. The simplest way of using the cepstrum is to use the real cepstrum to detect harmonics of a signal. Should a signal consist of a sine wave and its harmonics, the cepstrum of a signal will show a peak at the time (quefrency) corresponding to the period of the signal. This peak in the cepstrum may be easier to detect than multiple peaks in the power spectrum, thus simplifying the task of detecting the defect frequencies. When used in this way the cepstrum is very similar to the autocorrelation, with the log function providing greater dynamic range.

The cepstrum can also be used to perform a deconvolution on a signal. A series of ringing impulses may not be as easy to detect as a series of simple impulses. If the signal could be inverse filtered, then the original impulses could be recovered and (perhaps) more easily detected. Since the frequency of the ringing is not known this may not be easy to do using simple inverse filtering, but a technique known as homomorphic deconvolution can be used to accomplish the same thing, with only basic knowledge of the signal.

The idea is as follows: Convolution in the time domain is equivalent to multiplication in the frequency domain and multiplication in the frequency domain is equivalent to addition

in the cepstral domain. Thus a convolution in the time domain such as the impulse forcing function exciting a resonance can be reversed by a subtraction in the cepstral domain. This is assuming that either the forcing function or the response function is known. In reality neither is known very well. If we make the assumption that the forcing function will occupy the high order components of the cepstrum and the resonance the low order components of the cepstrum, the two may then be separated by windowing the cepstrum. When transformed back the windowing will result in a smoothed spectrum of the resonances and a high passed spectrum of the forcing function. These may be transformed back to the time domain if desired.

If the cepstrum is to be invertible to the time domain then the complex cepstrum with unwrapped phase must be used. This involves taking the complex logarithm of the frequency data.

$$F(z) = log \mid z \mid + Arg(z) + k2\pi \tag{2.12}$$

This complex logarithm has a phase ambiguity, where the phase of the logarithm can vary by integer multiples of 2π and still correspond to the same real and imaginary components. The phase must be unwrapped to remove any discontinuities in the phase, which can be a slow and complex procedure. If the cepstrum does not need to be inverted to the time domain, but only the frequency domain, then the real log of the magnitude of the spectrum can be taken. In this case the output data of the homomorphic filtering will be spectral magnitude data only. If the cepstrum is used to filter the data, then the result must still be analysed using one of the techniques mentioned previously.

2.3.9 The Metacepstrum

In [21] R.M. Stewart gives details of what he terms the Metacepstrum. This is a multi step procedure as follows:

The envelope power spectrum is calculated (Figure 2.2), using the appropriate window and frequency span.

The envelope spectrum is median filtered. This median, shown in Figure 2.3, is then subtracted from the original spectrum to leave only the peaks.

The result is then thresholded, reducing the low level hash, which serves only to confuse the analysis. This leaves only the larger peaks (Figure 2.4). Threshold levels of 3 or 6 dB are suggested. Any known corrupting tones can also be removed, for example 50 Hz mains interference is a common problem.

Then, the autocorrelation is performed, this yields the metacepstrum (Figure 2.5). This is then examined for the presence of the same passage frequencies that one looks for in power spectrum analysis.

Several claims are made for the greater effectiveness of the metacepstrum over the cepstrum. The first operation, the subtraction of the median compresses the dynamic range and

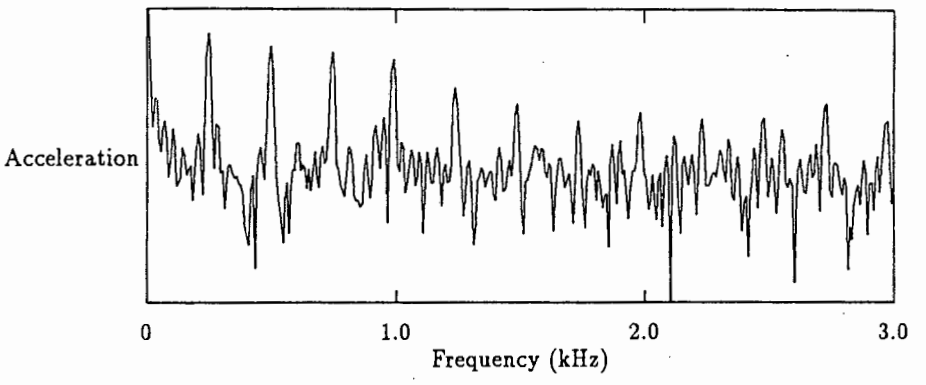


Figure 2.2: Envelope Power Spectrum

removes low frequency ripple. This in turn avoids the ringing that is present at the beginning of the cepstrum. The removal of corrupting tones and hash simplify the end analysis. While this technique has some worthwhile points, the most noticeable being the simplicity of the result (Figure 2.5) when compared with the envelope spectrum (2.2). All peaks are visible at their base frequencies, and not as harmonics, and the peaks are clearly visible. Despite this it does not seem to be widespread in its usage. A possibility that suggests itself is to use some of these techniques with other analysis methods, for instance the subtraction of the median may simplify power spectrum analysis.

2.4 Techniques Suited to Low Speed Machinery

Very large motors and motors for special applications may run at speeds from 10 to 60 rpm. At these speeds bearing faults become difficult to detect with the techniques used on normal machinery. Vibrations caused by defects will be of very low energy and signals may occur at very long intervals due to the low characteristic frequencies (see section 1.8). A bearing with a pitch diameter of 160mm, 16 balls of diameter 20mm and rotating at 60 RPM would have BPFO of:

$$\begin{aligned} BPFO &= 16 \times \frac{1}{2} \times \left(1 - \frac{20}{160}\cos(0)\right) \\ BPFO &= 7Hz \end{aligned}$$

The impulses caused by defect may occur at these low repetition frequencies, but the energy in them may be at a much higher frequency. A spectrum analyser used on a high enough frequency range to display this frequency may miss the events caused by damage. Envelope analysis can be used to demodulate the signal and reduce the frequencies of interest.

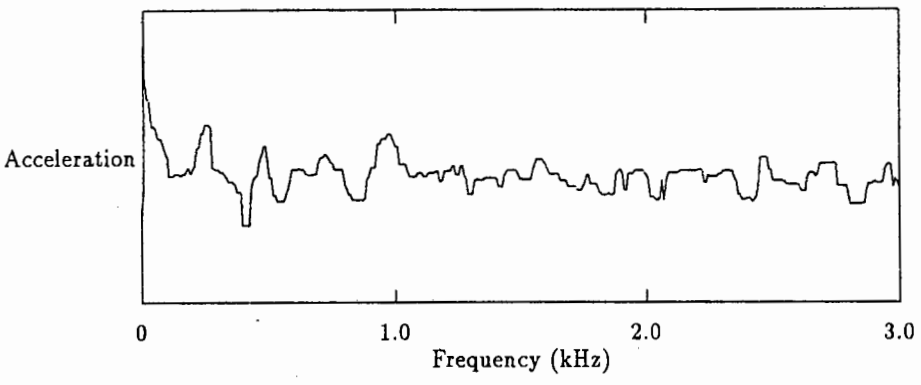


Figure 2.3: The Median Filtered Spectrum

These may then be displayed in time or analysed in the frequency domain using a spectrum analyser.

2.5 Additional Techniques

These techniques are used for reducing interference and enhancing signals. They can be used in combination with analysis techniques for simpler analysis and greater confidence in the results.

2.5.1 Synchronous Averaging

Synchronous averaging involves averaging the time domain signal from one machine rotation with subsequent rotations. This allows enhancement of vibrations that are synchronous with the rotation, while other vibrations average out to zero. This is not particularly useful in bearing vibrations, as the rotation of the cage is not synchronous with the rotation of the shaft. Averaging synchronous with the cage may be more successful [8] but access to the cage is extremely difficult in most practical situations. Synchronous averaging has however proven very successful in analysis of gears and can be used in bearing vibrations to distinguish between vibrations that are synchronous with the shaft rotation, and those that are at a frequency close to one caused by rotation, but not actually synchronous with it. This can then aid in determining the source of the various vibration components.

2.5.2 Order Tracking

Order Tracking is a technique that can be used in conjunction with other frequency domain techniques. Order tracking uses information from the shaft rotation to normalise frequencies

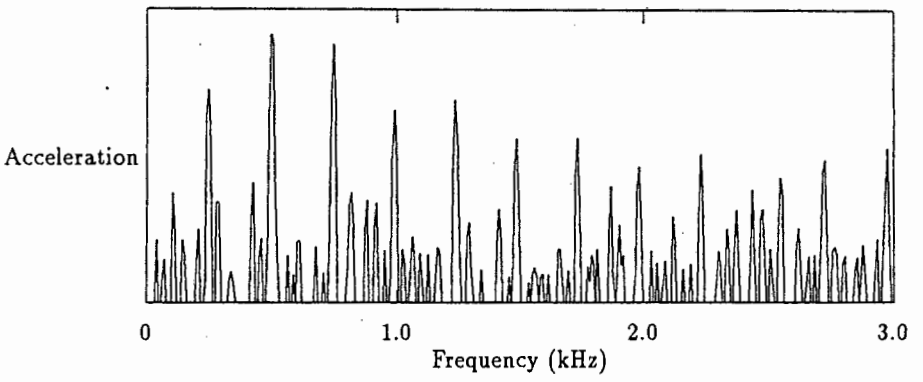


Figure 2.4: Spectrum with Median and Hash Removed

to orders of rotation, or multiples of the shaft rotation frequency. Thus even though the rotation frequency may change, any vibrations related to the rotation speed remain at the same order of rotation. This is useful in cases where the machine speed is not constant. Although order tracking is not new, advances in signal processing allow better frequency resolution and tracking over a greater range of speeds. Order tracking used to use a sampling clock phase locked to a multiple of the machine rotation. The problems associated with this were:

- The phase locked loop could only follow slow changes in the machine rotation, and always lagged behind the machine.
- Phase noise in the loop caused fluctuations in the sampling rate and thus blurring of spectral peaks.
- The phase locked loop had a limited capture and lock range and thus was limited in the range of allowable speeds.
- Anti-aliasing was performed with a tracking filter, which did not always have flat response.

Modern order tracking can be performed using digital sample interpolation [22]. This allows a greater speed range, as the system does not go out of lock at high or low speeds. The loop delay time in following speed changes is much reduced, as a digital system allows almost instantaneous response. The drawback to this technique is that it is computationally expensive, possibly requiring multiple processors in a real time system.

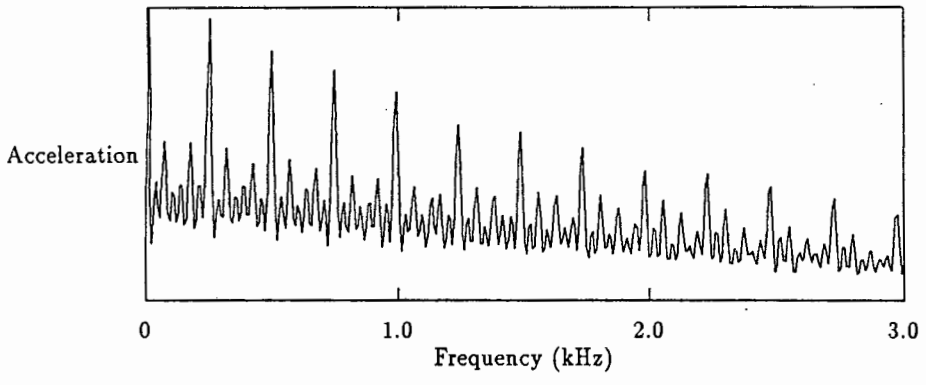


Figure 2.5: The Metacepstrum

2.5.3 Adaptive Filtering

Adaptive filtering is a technique that can be used to remove interfering signals produced by other vibration sources [23, 24, 25]. In adaptive filtering, two (or possibly more) accelerometers are used. One, the primary signal accelerometer is placed on the point where the vibrations would normally be measured. The second, the reference is placed further away near a source of interference, perhaps on the base of the machine, or on the other bearings. The reference signal is then passed through an FIR filter that filters and phase shifts the signal in order to make it as much like the primary signal as possible. It is then subtracted from the primary signal and the interference which is common to both is thus removed. The weights of the filter are continuously updated to provide the best match between reference and primary signal, and thus optimum cancelling. Tests at UCT [24, 26] and elsewhere [23] have shown adaptive filtering as a useful tool in reducing interference. Care should be taken when using adaptive filters that:

- The filter algorithm is stable.
- The filter adapts to changes in the signals quickly enough that the filter improves and not degrades the signal.
- The filter does not adapt so quickly as to modify the signal being measured, for instance should the primary signal and reference signal both consist of sinusoids at closely spaced frequencies, it is possible that the adaptive filter weights can adjust fast enough to phase modulate the reference sinusoid and thus cancel the primary signal, even though this is not desired.

2.6 Summary and Discussion of Signal Processing Techniques

A summary of the important aspects of the algorithms discussed has been made in Table 2.3. The aspects summarised are:

1. Ease of use 1 = good 5 = bad
2. Number of Calculations (N= number of points)
3. Ability to detect early damage (**P**oor **M**oderate **G**ood)
4. Ability to detect advanced damage(**P**oor **M**oderate **G**ood)

Not all the techniques discussed have been summarised, as some of them are not in common usage, or too little is known about their performance. This table tends to highlight one of the initial assumptions that not all the techniques are effective under all conditions. Some of these techniques have been tested using vibrations recorded on the bearing test rig in the mechanical engineering department. The results of these tests and the conclusions reached are discussed in Chapter 3.

Technique	Ease of Use	Computational Efficiency	Early Damage	Late Damage
RMS	1	N	P	G
Kurtosis	1	N	G	P
Crest Factor	1	N	M	P
Power Spectrum	4	$NlogN + N$	M	G
Envelope Spectrum	3	$NlogN + N$	M/G	G
Cepstrum	2	$NlogN + N$	M	M
Meta Cepstrum	2	$NlogN + N$	M	M
Bispectrum	4	N^2	–	–

Table 2.3: A Summary of Features of Different Techniques

From the review of signal processing techniques, the following conclusions can be made:

1. The time domain analysis techniques are the simplest to compute, use and analyse, but do not perform well under certain conditions.
2. The frequency domain techniques perform well under a wide range of conditions, but details of the bearing dimensions and shaft speed are required for them to be used effectively. The information provided by these techniques is also more difficult to interpret and apply than the time domain techniques.

3. The frequency domain techniques vary in terms of ease of use and the degree of detail presented to the user. The cepstrum, metacepstrum and envelope spectrum are easier to interpret than the power spectrum, but also provide a reduced amount of information.

4. The spectral coherence and bicoherence techniques provide a large amount of information about the vibration signal, but at present very little is known about their interpretation and use in machine monitoring.

5. The signal processing techniques tend to complement each other. RMS vibration monitoring and spectral alarm band techniques are useful in situations where measures of the impulsiveness of the signal (kurtosis, crest factor) fail to provide useful information. Frequency domain techniques provide the diagnostic information that is not available from time domain statistics.

Chapter 3

Bearing Tests, Results and Application of Algorithms

3.1 The Bearing Test Rig

The rig used was one built by Brad Leggat for his work on bearings [27]. The bearing rig uses three bearings mounted side by side on a shaft. The outer two are support bearings, the inner one is the test bearing, which is loaded using a hydraulic arrangement. The bearings are housed in plummer blocks and attached to the shaft using taper-lock adaptors. Because of this arrangement, the bearings used are all of the self aligning type, the outer two being spherical roller bearings, and the test bearing can be either a spherical roller bearing or a double row ball bearing. The shaft is attached via a flexible coupling to DC motor with speed control. This allows the shaft speed to be varied in the range 200 to 1900 rpm. A slotted disc and optical pickup allow the speed to be accurately read using a digital frequency counter.

3.2 Bearing Rated Life

The bearing rated life, or L_{10} life is defined as the number of revolutions that 90% of bearings can endure, under those particular conditions before showing the first signs of fatigue (flaking, spalling etc) [28] p.27. Note that this is a statistical average, and the life of individual bearings may vary. The ISO equation for L_{10} life can be found in [28] and is as follows:

$$L_{10} = \left(\frac{C}{P}\right)^{p} \times 1000000 \tag{3.1}$$

- p = 3 (ball bearings)
- p = 3.3 (roller bearings)
- P = Equivalent dynamic load rating

- C = Basic Dynamic load rating [1]

This is the simplest form of life equation. Other forms take into account factors such as temperature and materials.

The ratings of an SKF 1207 K bearing are as follows:
load 15900 N
maximum operating speed (grease lubrication) 9000 rpm
maximum operating speed (oil lubrication) 11000 rpm
From this it is possible to calculate the L_{10} life running under our test conditions.

$$L_{10} = \left(\frac{15900}{22000}\right)^3 = 0.37751 \tag{3.2}$$

Thus the number of rotations is 377505 and the time to failure at 1700 rpm = 222 min and at 1800 rpm = 210 min.
High running temperatures, axial loads and contaminated lubricants will all act towards reducing this life.

3.3 Testing Bearings by Running Under High Loads

Three test were run using Steyr 1207K bearings as the test bearings (These are similar to the SKF bearing). The tests were run with loads higher than rated to accelerate the wear process. A total of three tests were run, during which the vibrations were recorded using a Hewlett-Packard spectrum analyser and the Proto 56 DSP board. The first test was run at a shaft speed of 1800 rpm. The test bearing was accidentally brinneled by applying a high load to the stationary bearing while setting up the test rig. Thereafter the bearing was run for approximately two hours. The bearing was removed and examined, and although worn, no other serious damage had occurred. The wear is described in Appendix A. The second test was run at a speed of 1700 rpm. Spalling on the outer race occurred after about 10 minutes running. The damage then spread to both sides of the outer race over a period of a few hours. The pitting of the outer race can be seen in the photograph in Figure 3.1. The balls (Figure 3.2) showed no visible damage.

The third test was also run at a speed of 1700 rpm. Spalling of the outer race occurred after approximately two and a half hours. The test was run for a further hour, by which time the damage had spread to both sides of the outer race.

[1] The load at which 90% of bearings achieve 1000000 revolutions

The files selected from these tests have been renamed. One example was selected from test one, and from tests two and three an example of an undamaged bearing, initial damage and advanced damage have been selected. The names used in labelling the graphs correspond to the following data files:

Brinell Damaged Bearing	Test 1	TIME3.DAT	21 min
Fatigue Damaged Bearing 1	Test 2	CAPT1.DAT	10 min
Fatigue Damaged Bearing 1	Test 2	CAPT5.DAT	35 min
Fatigue Damaged Bearing 1	Test 2	CAP4.DAT	1hr 49 min
Fatigue Damaged Bearing 2	Test 3	CAPT2.DAT	12 min
Fatigue Damaged Bearing 2	Test 3	CAPT16a.DAT	2hr 55 min
Fatigue Damaged Bearing 2	Test 3	CAPT31.DAT	4hr 12 min

Figure 3.1: Pitted Race of Fatigue Damaged Bearing One, After 1Hr 23 min

3.4 Tests on Bearings with Spark Induced Defects

These tests, conducted by Etienne Kruger were performed using intentionally damaged bearings. The defects were created by discharging a capacitor against the bearing race. The data gathered in this way differs from the data from the previous tests in several ways.

- The load used in this test was much lower than in any other tests.

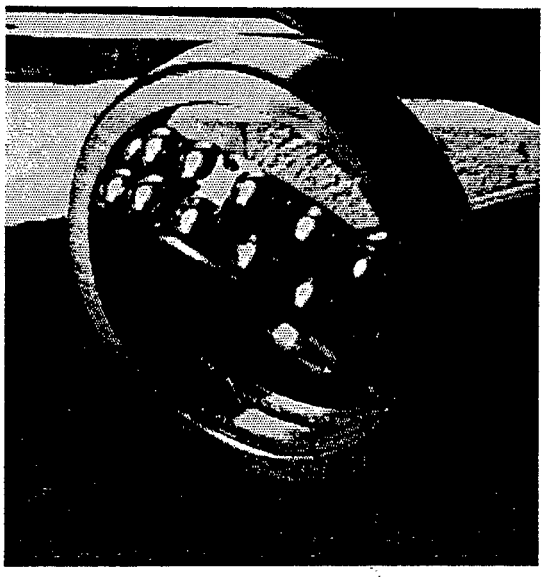


Figure 3.2: Undamaged Balls of Fatigue Damaged Bearing One after 1Hr 23min

- The shaft speeds of the test rig were lower.
- The nature of the bearing defect was different, as in these tests, it was a point defect on an otherwise undamaged bearing.

The details of the machine speeds and loads for this data are as shown below.

Test Conditions			
Test	Speed (RPM)	Load (kN)	low pass cut-off
C1	1000	6.9	3 Hz
C2	1000	6.9	2 kHz
C3	1000	4.6	3 Hz
C4	1000	4.6	2 kHz
C5	1000	2.8	3 Hz
C6	1000	2.8	2 kHz
C7	600	4.6	3 Hz
C8	600	4.6	2 kHz
D1	1000	4.6	3 Hz
D2	1000	6.9	3 Hz
D3	1000	9.3	3 Hz

The data file names are made up of two parts. The first is the test number, for example D1. The second part is the angle of the defect from the load zone, this gives the full file name such as D1-0. The graphs of the test data are labelled using the names below.

Graph Names	Original Name	Load (kN)	Low Pass Fr.	Speed
Spark Damaged Bearing 1	D1-0	4.6	3 Hz	1000
Spark Damaged Bearing 2	D1-180	4.6	3 Hz	1000
Spark Damaged Bearing 3	C5-0	2.8	3 Hz	1000
Spark Damaged Bearing 4	C7-0	4.6	3 Hz	600
Spark Damaged Bearing 5	C1-0	6.9	3 Hz	1000
Spark Damaged Bearing 8	C1-150	6.9	3 Hz	1000
Spark Damaged Bearing 9	C1-30	6.9	3 Hz	1000
Spark Damaged Bearing 6	C8-0	4.6	2 kHz	600
Spark Damaged Bearing 7	D3-0	9.3	3 Hz	1000

These names may be shortened to the abbreviation SDB 1 etc.

3.5 Analysis of the Test data

Several analysis techniques have been considered. Some of these have been demonstrated on a section of the data, but it was decided that a few of the techniques would be used on a wide range of data to compare the effectiveness of these techniques under various conditions. The techniques chosen are as follows:

- Kurtosis
- Crest factor
- Power Spectrum
- Envelope Spectrum
- Spectrum of acceleration to the fourth power (referred to as y^4 spectra).
- Power Cepstrum

The data was analysed in MATHCAD as follows:

1. A data file of 4095 samples were read in. The data was divided into four segments of 1024 samples each.
2. The kurtosis and crest factor was calculated for each of these spans, as well as for the whole block.

3. The FFT of each of these spans was calculated and these were averaged to calculate the power spectrum of the signal.

4. The time data was filtered using a sixteen point band pass filter. This signal was rectified and the resulting envelope spectrum obtained. The spectrum of the fourth power of the signal was also calculated.

5. The Cepstrum of the signal was obtained as follows: The log of the magnitude of the power spectrum was obtained (A 1024 point real symmetrical signal). The inverse FFT of this spectrum was performed to produce a cepstrum with 1024 REAL data points which was symmetrical.

The data used was from two sources: The data captured using the HP spectrum analyser during the first tests, and the data captured by Etienne Kruger using an A to D board during his tests. Since the sampling rates were not the same, the data from Etienne Krugers tests was low pass filtered and decimated to give an equivalent sampling rate of 15 kHz. This is comparable with the 16.4 kHz sampling rate of the spectrum analyser.

3.6 Kurtosis and Crest Factor Analysis of the Data

The analysis techniques were tested on several sets of test data. Table 3.1 shows the kurtosis and crest factor values for the test data. The calculations were performed on four individual segments of 1024 samples, as well as on the full section of 4096 samples. The data can be divided into two broad classes, with several sub classes.

1. The data from the fatigue damaged bearings.
   (a) The data from undamaged bearings: FDB 1u and FDB 2u
   (b) The data from the moderately damaged bearings: FDB 1m and FDB 2m
   (c) The data from the bearings with advanced damage
2. The data from the spark damaged bearings.
   (a) The tests with the defect outside the load zone, SDB2 and SDB 8
   (b) The tests with lower shaft speeds, SDB 5 and SDB 6

	Kurtosis					Crest Factor					
File Name	Whole Span	Part 1	Part 2	Part 3	Part 4	Whole Span	Part 1	Part 2	Part 3	Part 4	Comment
SDB 4	4.8	4.6	4.5	4.9	5.0	3.7	3.8	3.2	3.6	3.5	Spark Damage
SDB 9	4.8	4.6	5.3	4.7	4.6	4.6	3.9	4.0	4.7	4.4	Spark Damage
SDB 8	2.9	2.7	3.1	2.8	2.9	4.0	3.0	3.8	3.4	3.8	Undamaged [2]
SDB 3	4.4	4.5	4.4	4.3	4.3	3.6	3.4	3.6	3.4	3.4	Spark Damage
SDB 5	6.9	7.1	7.0	6.9	6.7	4.9	4.5	4.5	4.9	4.3	Low Speed
SDB 6	10.0	9.3	10.0	10.0	10.5	6.2	4.3	4.4	4.3	4.3	Low Speed
SDB 1	6.3	6.0	6.1	6.4	6.7	4.2	4.1	4.1	4.2	4.2	Spark Damage
SDB 2	3.0	3.1	2.7	3.1	3.1	3.7	3.1	3.0	3.6	3.6	Undamaged [2]
SDB 7	5.3	5.5	5.1	5.3	5.3	3.8	3.5	3.4	3.4	3.6	Spark Damage
BDB 1	3.4	2.8	3.8	3.8	3.1	5.4	3.3	3.5	4.8	3.5	Brinelled
FDB 1u	3.0	3.0	2.9	3.2	2.9	4.1	3.4	3.0	4.1	3.1	Undamaged
FDB 1m	2.6	2.6	2.7	2.5	2.6	3.1	2.7	3.1	3.0	3.0	Moderate
FDB 1a	7.6	7.2	9.7	6.3	7.1	6.1	3.9	4.7	4.4	3.9	Advanced
FDB 2u	3.2	3.0	3.5	3.2	3.0	3.9	3.1	3.8	3.3	3.5	Undamaged
FDB 2m	4.3	3.3	3.4	4.0	5.2	6.2	4.0	3.8	4.8	5.7	Moderate
FDB 2a	2.7	2.7	2.6	2.8	2.7	3.2	2.6	2.7	2.9	3.0	Advanced

Table 3.1: Kurtosis Values For the Test Data

From these results, we can note the following points:

- The vibrations of undamaged bearings (FDB 2u and 3u) have kurtosis values of approximately 3, and crest factor values of approximately 4. The vibrations of bearings with defects outside the load zone produce similar results (kurtosis values of 2.9 and 3.0 crest factor values of 4.0 and 3.7)
- The kurtosis values of the vibrations from bearings with point defects were higher than those of the undamaged bearings, varying from 4.4 to 10.0. The crest factor results were not always higher than those of undamaged bearings, and varied from 3.7 to 6.2.
- The vibrations of the bearings with moderate and advanced damage (FDB 1m, FDB 1a, FDB 2m, FDB 2a) had a range of kurtosis values, from 2.6 to 7.6. The crest factor followed the same trends as kurtosis and varied from 3.1 to 6.2.
- Test 1, the brinelled bearing had a kurtosis value of 3.4 that was slightly above the value expected of a normal bearing [3], and a crest factor value of 5.4 which was also above the normal.

The following conclusions can be drawn:

[2]This bearing has the defect outside the load zone, and produces vibrations similar to those of an undamaged bearing

[3]Dyer [12] gives the range of kurtosis as being $\pm$ 8 %, or between 2.76 and 3.24

1. The kurtosis value for vibrations of an undamaged bearing is approximately 3.0, while the crest factor value is approximately 4.0.

2. The kurtosis and crest factor values follow the same trends in most but not all the cases.

3. These two statistics failed to indicate damage in two of the cases of moderate or advanced damage.

4. Neither crestfactor nor kurtosis indicated damage in the case of undamaged bearings.

3.7 Test Data Selected for Further Analysis

A small number of the tests used in the review of kurtosis and crest factor were used for further tests using other techniques. The tests were selected to give a variety of different load and speed conditions. The tests selected are: Spark Damaged Bearings 1 ... 5, Brinell Damaged Bearing 1 and Fatigue damaged Bearings 1 and 2.

The vibration recordings for these tests were not calibrated against any reference, and for this reason the graphs do not have Y axis units. The power spectrum plots are marked in decibels. This is a scale relative to the top position on the graph, and not relative to any fixed reference. The power cepstrum plots are also marked in dB. This is an absolute scale, as the scale of the cepstrum is based on the relative dB scale of the power spectrum.

3.8 Graphs of Spark Damaged Bearing 1

For these tests, the shaft speed was 1000 RPM. The load was 4.6 kN and the low frequency cut-off 3 Hz. The characteristic frequencies are therefore:

R 16.7 Hz

FTF 7.08 Hz

BPFO 113.3 Hz $\equiv$ 8.83 ms

The kurtosis value is 6.3 which indicates definite impulses in the signal. The impulses caused by the outer race pit are clearly visible in figure 3.3. The impulses are spaced 8.7 ms apart, and ring at approximately 1kHz and 3kHz.

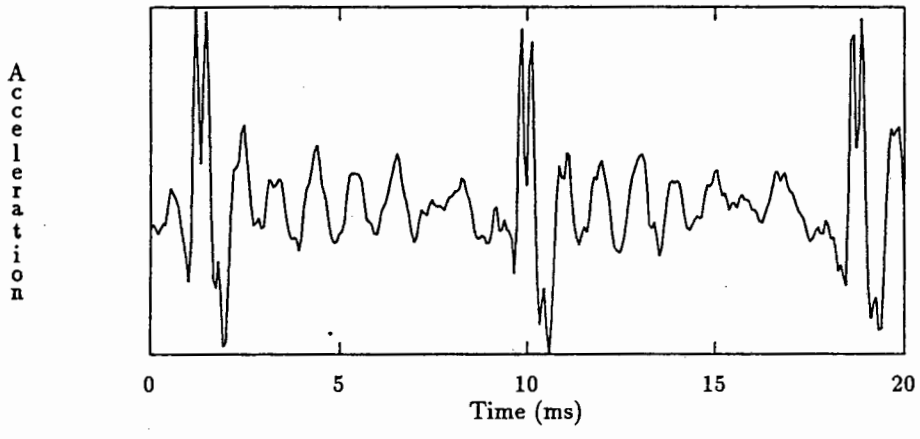


Figure 3.3: SDB 1, Time Signal

In the power spectrum, Fig 3.4, the resonances at one kHz and three kHz are clearly visible, as well as the large number of harmonics of BPFO. These appear to be spaced 112 Hz apart, which is consistent with the calculated BPFO and the observed time domain signal.

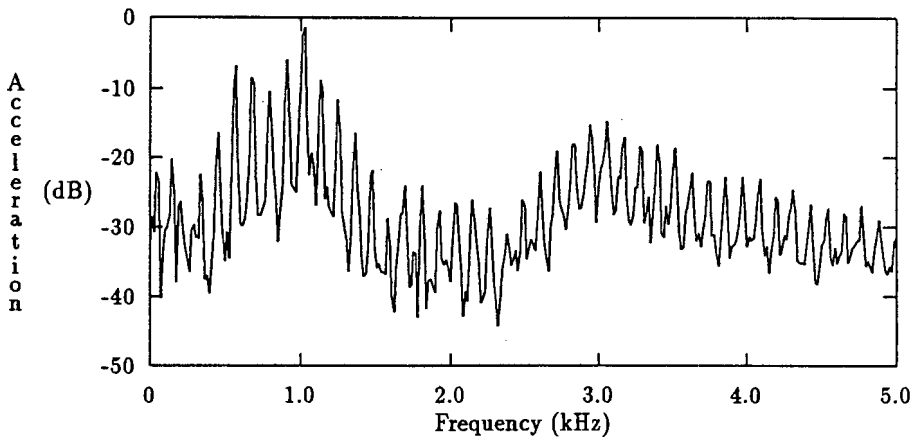


Figure 3.4: SDB 1, Power Spectrum

The harmonics of 112 Hz that were visible in the power spectrum are again visible in the envelope spectrum, Fig 3.5 [4]. Some sidebands are visible, these are more obvious in the next figure, Fig 3.6.

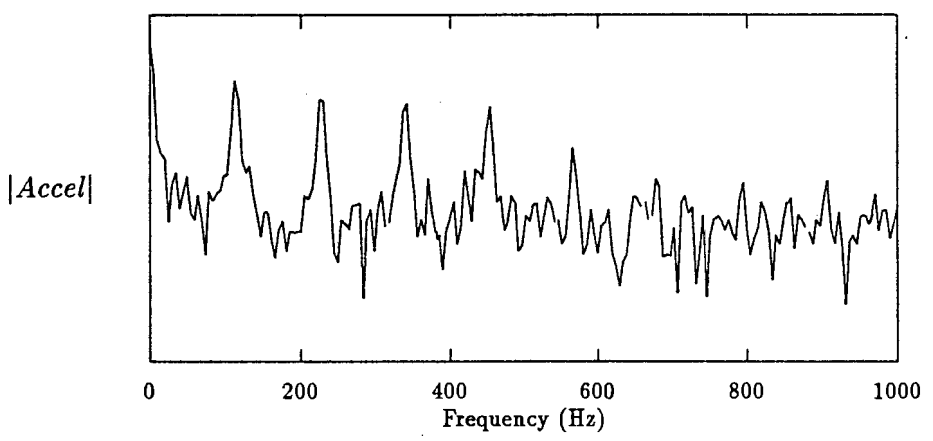


Figure 3.5: SDB 1, Envelope Spectrum

[4]Note that the frequency scale is not the same as that used in the power spectrum

The BPFO harmonics in the spectrum of acceleration to fourth power (Figure 3.6) show clear sidebands at 16 and 32 Hz, which is one times and two times the rotational frequency respectively. This shows the presence of a vibration component or modulation at the rotational frequency which was not obvious in the power spectrum.

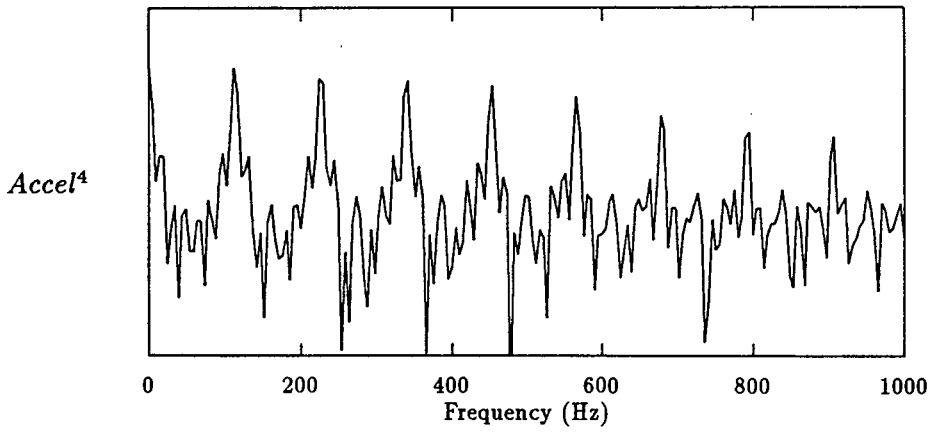


Figure 3.6: SDB 1, Spectrum of Acceleration to the Fourth Power

Components at 8.8 and 17.6 ms are very clearly visible in the cepstrum, Fig 3.7 a further indicator of the repetitive BPFO signal.

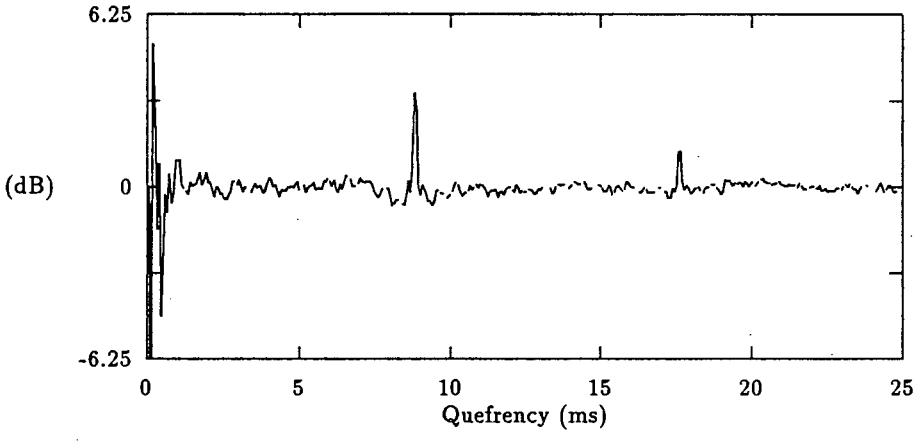


Figure 3.7: SDB 1, Cepstrum

3.9 Graphs of Spark Damaged Bearing 2

For these tests, the shaft speed was 1000 RPM. The load 4.6 kN and the low frequency cut-off 3 Hz. The characteristic frequencies are therefore:

R 16.7 Hz

FTF 7.08 Hz

BPFO 113.3 Hz $\equiv$ 8.83 ms

The defect was 180 degrees out of the load zone, and the signals appear very much like those of a good bearing. There are no obvious impulses in the time domain signal, and it appears much like white noise with a kurtosis value of 3.0 .

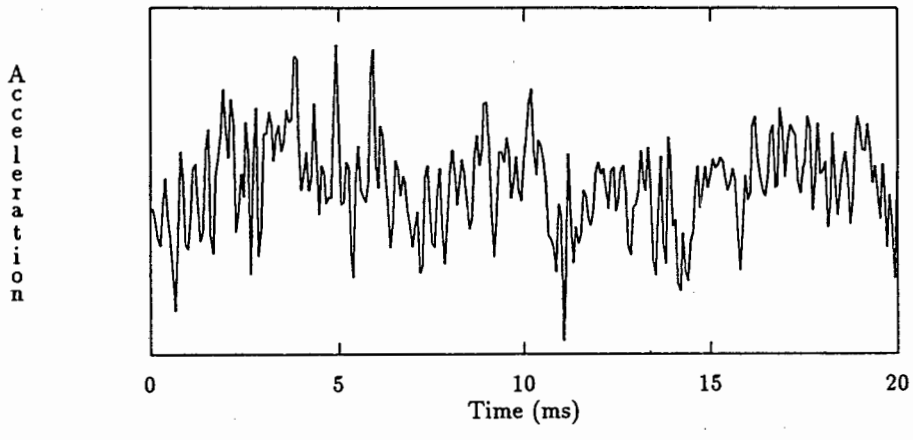


Figure 3.8: SDB 2, Time Signal

The power spectrum, Fig 3.9, does not show the obvious BPFO harmonics of the case of the defect at zero degrees (figure 3.4) but some frequencies are visible, the first is at 47 Hz, then there are others at 156, 438, 581 and 688 Hz. At higher frequencies, just above the 1 kHz mark some harmonics of BPFO are visible..

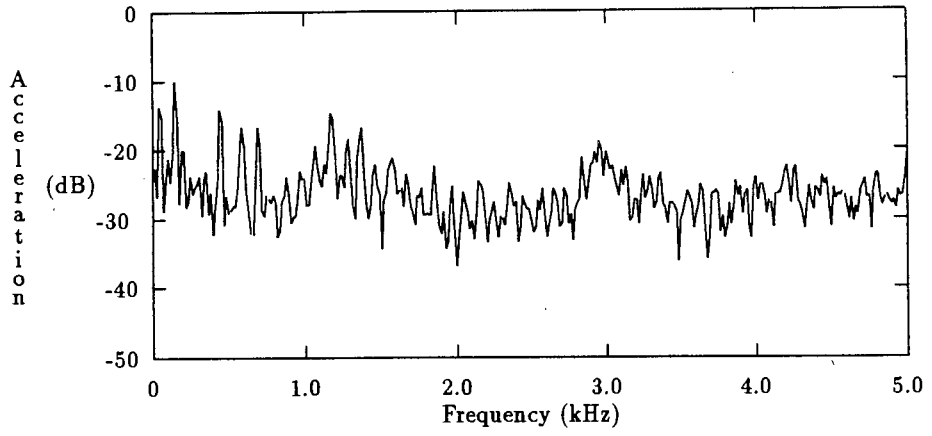


Figure 3.9: SDB 2, Power Spectrum

The envelope spectrum, Figure 3.10, does not show any obvious BPFO harmonics and in general lacks distinct frequency components

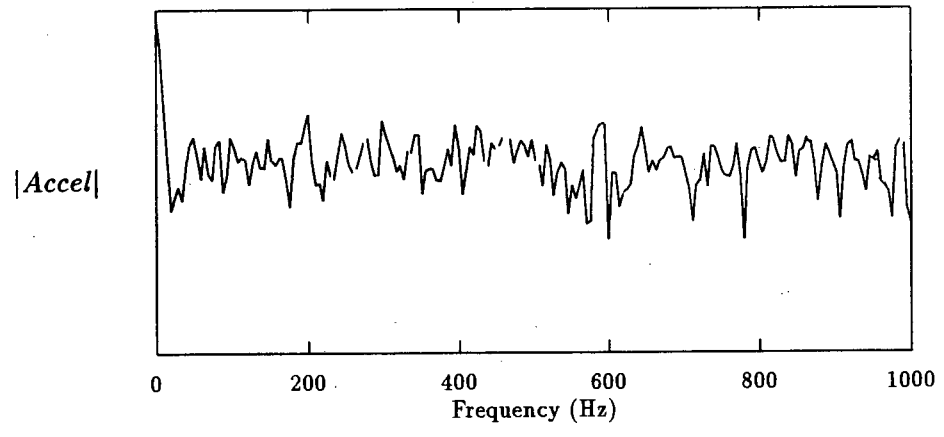


Figure 3.10: SDB 2, Envelope Spectrum

Figure 3.11, the Y^4 spectrum, like the envelope spectrum does not show any obvious defect frequencies.

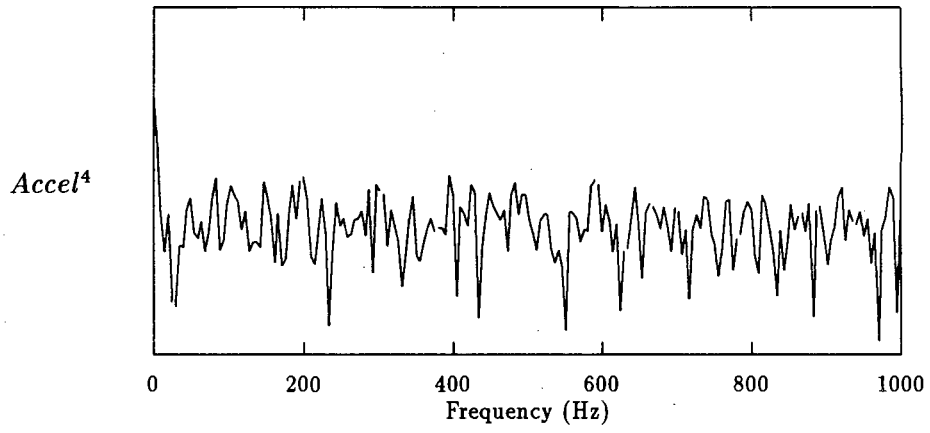


Figure 3.11: SDB 2, Spectrum of Acceleration to the Fourth Power

A rahmonic is visible at 10.19 ms in Fig 3.12, the cepstrum, but even in an undamaged bearing, some vibration at the ball passage frequency is to be expected.

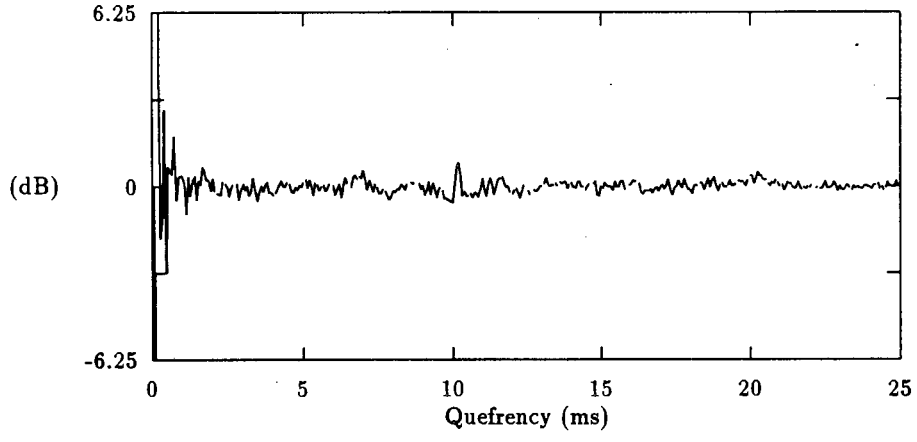


Figure 3.12: SDB 2, Cepstrum

3.10 Graphs of Capture Spark Damaged Bearing 3

For these tests, the shaft speed was 1000 RPM. The load was 2.8 kN which is lower than the 4.6 kN of SDB 1. The low frequency cut-off 3 Hz. The characteristic frequencies for this shaft speed are:

R 16.7 Hz

FTF 7.08 Hz

BPFO 113.3 Hz ≡ 8.83 ms

The time signal, Fig 3.13 is very much like that of Spark Damaged Bearing 1 which can be seen in Figure 3.3. The defect causes the expected impulses 8.6 ms apart with the ringing at 1 kHz more pronounced than that in test SDB 1. The kurtosis is moderately high, at 4.4 as a result of the dominant ringing tone (the kurtosis drops toward 2.8 of a sine wave).

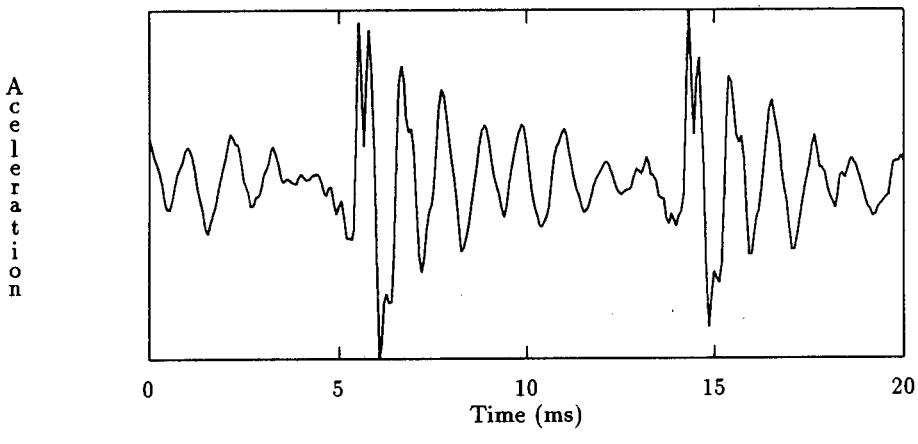


Figure 3.13: SDB 3, Time Signal

The power spectrum (Figure 3.14) shows a more pronounced resonance at 1 kHz and 3kHz than was visible in the power spectrum of SDB 1. There are harmonics throughout the spectrum spaced at 114 Hz intervals, equal to BPFO.

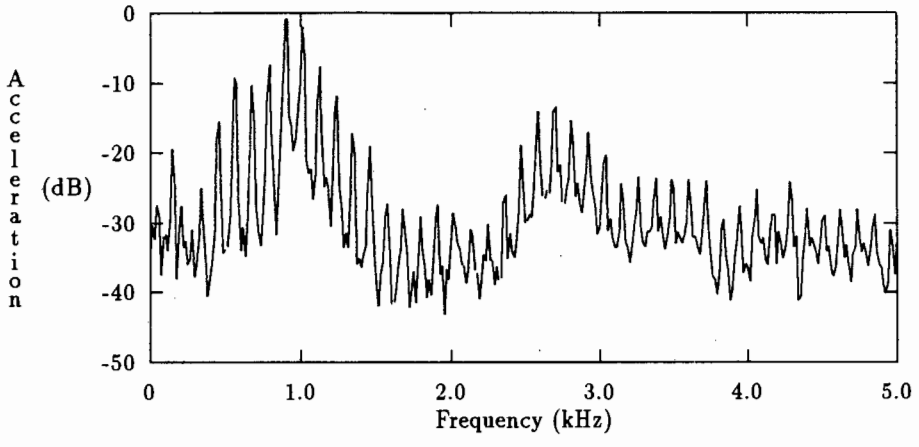


Figure 3.14: SDB 3, Power Spectrum

The envelope spectrum, Fig 3.15, shows clear harmonics of BPFO, with some small sidebands.

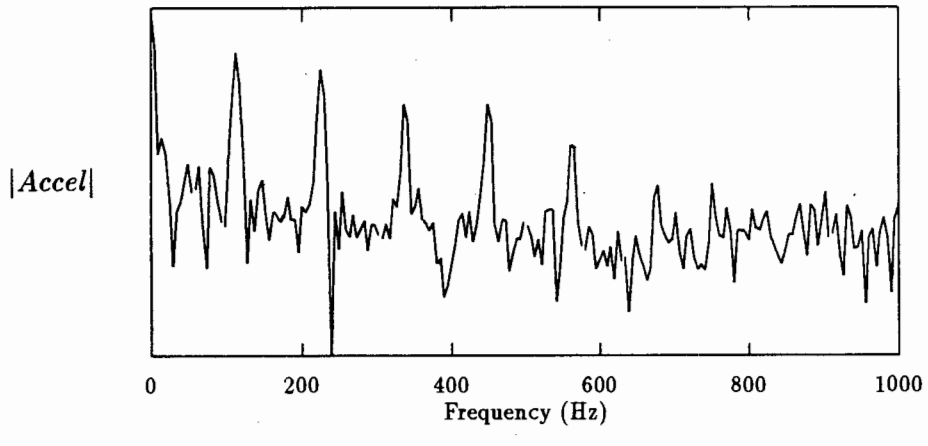


Figure 3.15: SDB 3, Envelope Spectrum

The Y^4 spectrum, Figure 3.16, has the same BPFO harmonics as the envelope spectrum, with more obvious sidebands at 32 and 16 Hz, the shaft frequency.

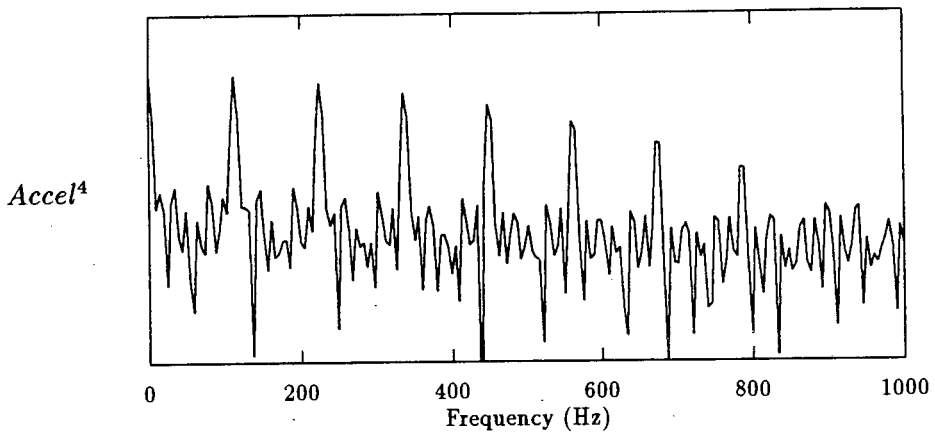


Figure 3.16: SDB 3, Spectrum of Acceleration to the Fourth Power

In fig 3.17 the cepstral component at 8.8ms is very large, indicating the large number of BPFO harmonics.

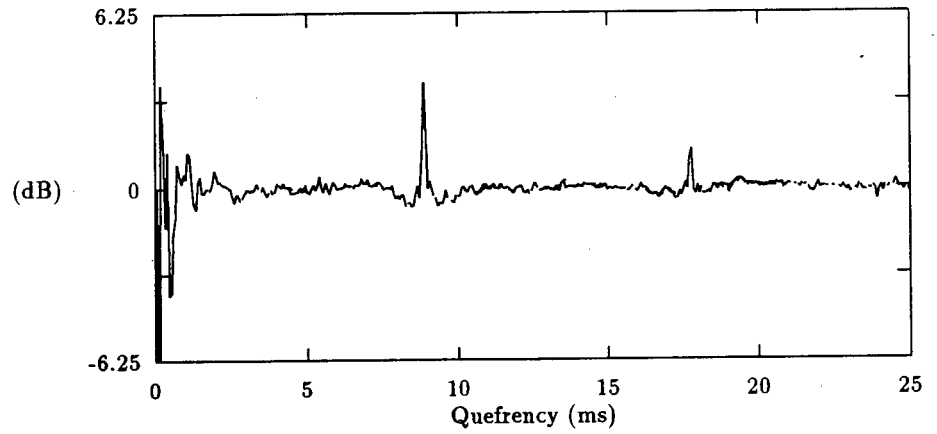


Figure 3.17: SDB 3, Cepstrum

3.11 Graphs of Spark Damaged Bearing 4

The shaft frequency is lower than that used in the other tests on spark damaged bearings, at 600 RPM with the load 4.6 kN. The characteristic frequencies are:

R 10.0 Hz

FTF 4.3 Hz

BPFO 68.0 Hz ≡ 14.7 ms

As with the other tests, the impulses are visible in the time domain, spaced 12 ms apart and ringing at approximately 1kHz. Kurtosis is high, at 6.9. This can be compared with the time signal of SDB 3, fig 3.13 which has more ringing of the impulses, and shorter spacing between them [5].

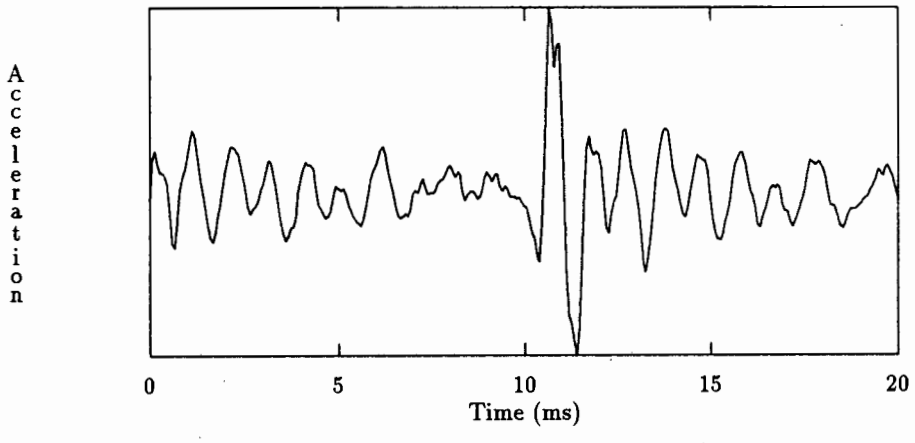


Figure 3.18: SDB 4, Time Signal

[5]The relationship of kurtosis to pulse width and pulse spacing is shown in table 2.1

The harmonics of BPFO in the power spectrum, Fig 3.19 are clearly visible, this time spaced 87 Hz apart. From this it appears that the shaft was running at a slightly higher speed that the nominal 600 RPM, and the speed was closer to 780 RPM.

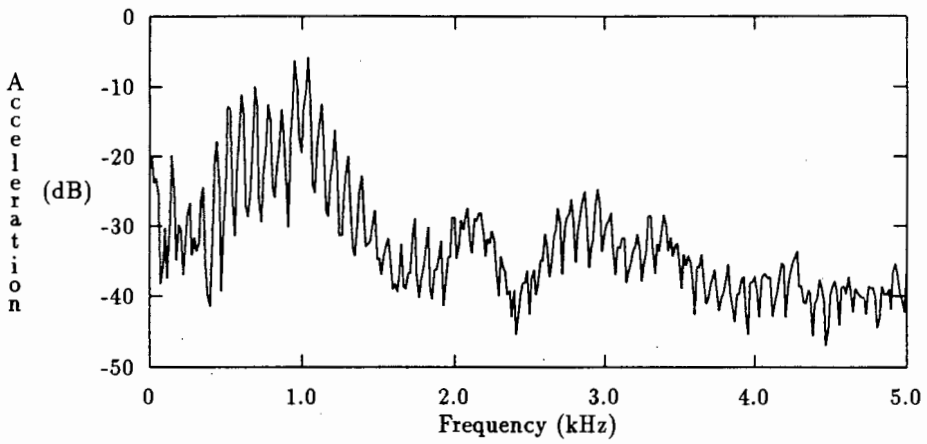


Figure 3.19: SDB 4, Power Spectrum

The envelope spectrum, Figure 3.20 shows these BPFO components, without any obvious sidebands.

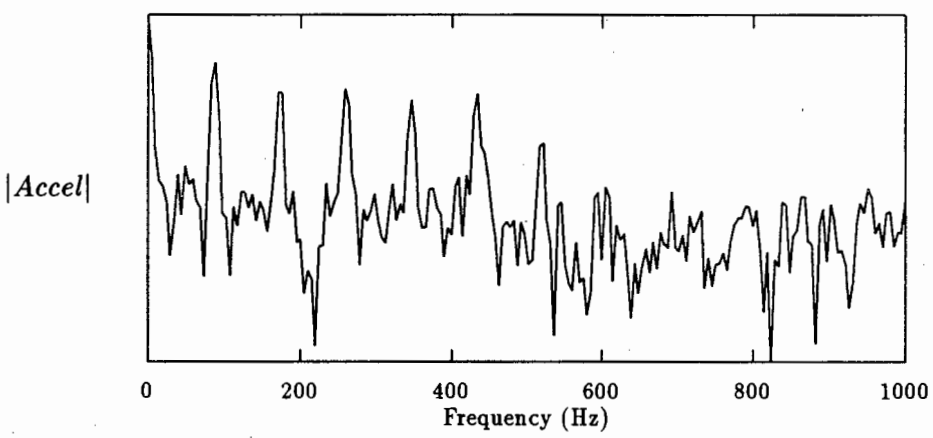


Figure 3.20: SDB 4, Envelope Spectrum

The harmonics are also visible in the y^4 spectrum, this time accompanied by 13 and 26 Hz sidebands. Since the actual shaft speed was approximately 780 RPM, these sidebands are at one and two times the rotational speed.

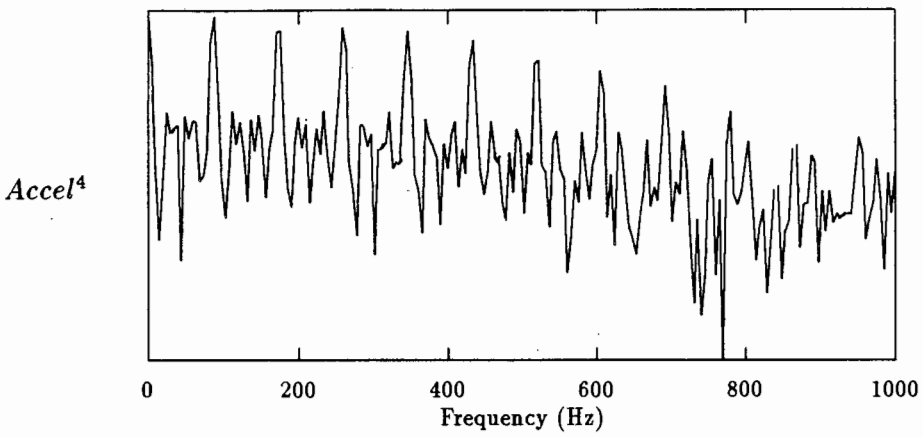


Figure 3.21: SDB 4, Spectrum of Acceleration to the Fourth Power

In Figure 3.22 the cepstral component at 11.5 ms (equivalent to BPFO) is not as pronounced as in D1-0, but is still very clearly visible.

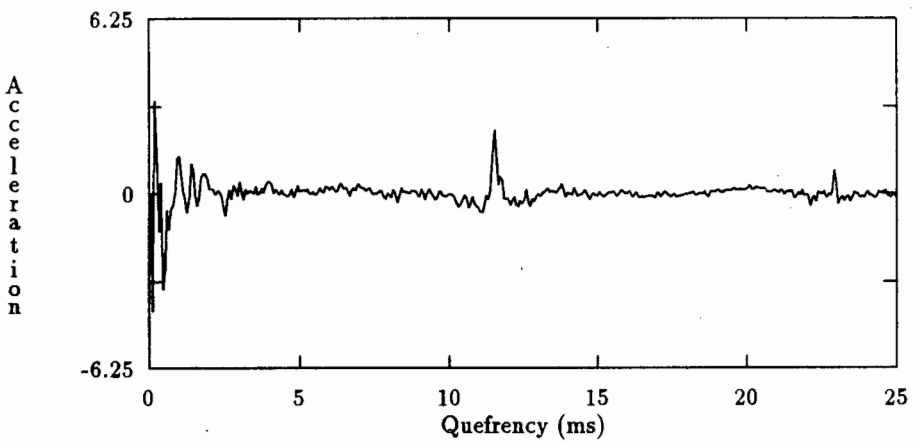


Figure 3.22: SDB 4, Cepstrum

Throughout these tests it can be noted that the amplitude of the cepstral component is dependent on three features of the spectrum:

- The number of harmonics of the fundamental that are visible in the spectrum.

- The height of these harmonics above the base line noise of the spectrum.
- The shape of these harmonic peaks.

The influence of these factors can be seen in the following examples. Figure 3.19 Page 51, the power spectrum of test SDB 4 shows a large number of harmonics of the ball pass frequency. The resulting cepstrum has corresponding rahmonics, yet these are not as large as those in the cepstra of tests SDB 1 fig:3.7 and SDB 3. fig:3.17 The harmonic components in the power spectrum of SDB 4 are however closer together than those of tests SDB 1 and SDB 3, and thus have begun to merge together, due to the limited resolution of the power spectrum. The peaks are therefore not as high above the base level of the spectrum, resulting in a reduced cepstral rahmonic.

Another aspect of the cepstrum can be seen in Figure 3.61, the cepstrum of Fatigue Damaged Bearing 2m. The second rahmonic is very prominent, especially in relation to the other tests. When one looks at the corresponding power spectrum, Fig 3.58 one can see that although the harmonics are not more numerous than in other tests, these harmonic peaks are especially narrow and distinct. Just as a time domain signal that consists of a series of narrow impulses will generate a power spectrum with higher harmonics, thus a power spectrum with narrow harmonics generates a cepstrum with higher rahmonics.

3.12 Graphs of Spark Damaged Bearing 5

For these tests, the shaft speed was 1000 RPM. The load was 6.9 kN, the highest load of the spark damaged bearings. The low frequency cut-off 3 Hz. The characteristic frequencies are therefore:

R 16.7 Hz

FTF 7.08 Hz

BPFO 113.3 Hz ≡ 8.83 ms

The time domain impulses are spaced 9ms apart. The ringing at 1kHz is very pronounced. The kurtosis value for this signal is 4.8.

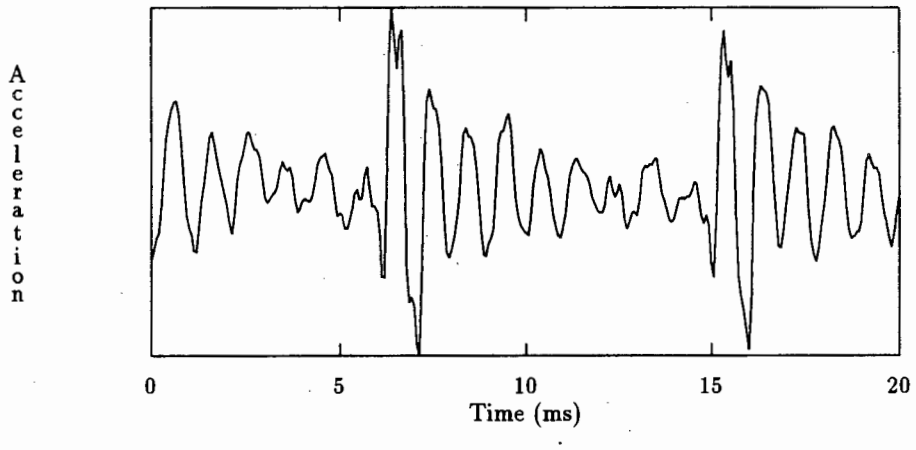


Figure 3.23: SDB 5, Time Domain Waveform

The signal was then filtered to give the waveform shown in figure 3.24. The ringing is reduced, and the kurtosis is higher, at 6.9, confirming that ringing is responsible for the low kurtosis value. The filtering was performed by windowing the signal in the frequency domain, to reduce the dominant frequencies by approximately 6 dB. An equivalent digital filter can be constructed to filter the signal in the time domain.

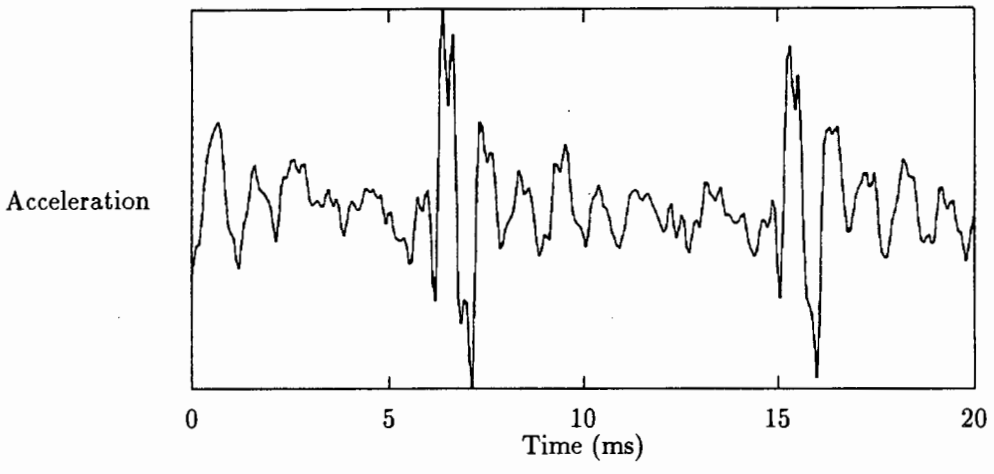


Figure 3.24: SDB 5, Frequency Compensated Signal

The power spectrum, Figure 3.25, shows the 1kHz resonance and also the BPFO harmonics spaced 110 Hz apart. Some small sub-harmonics are visible between the larger harmonics.

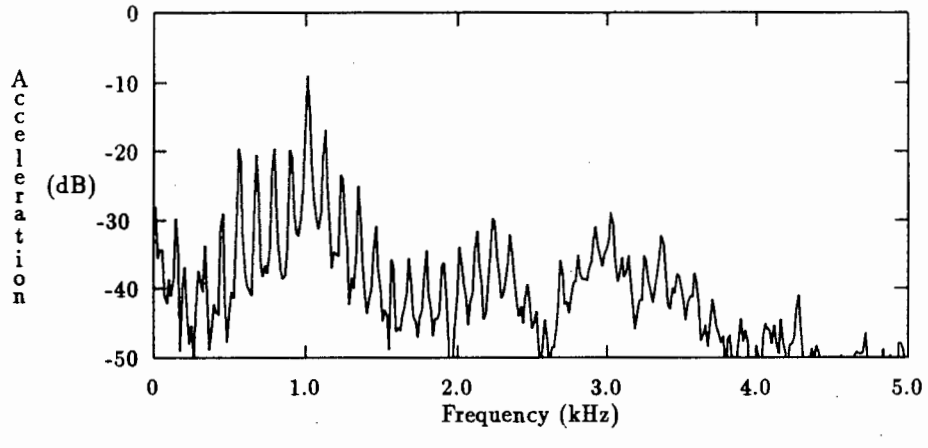


Figure 3.25: SDB 5, Power spectrum

The 110 Hz BPFO harmonics are visible in the envelope spectrum, but no side bands are in evidence.

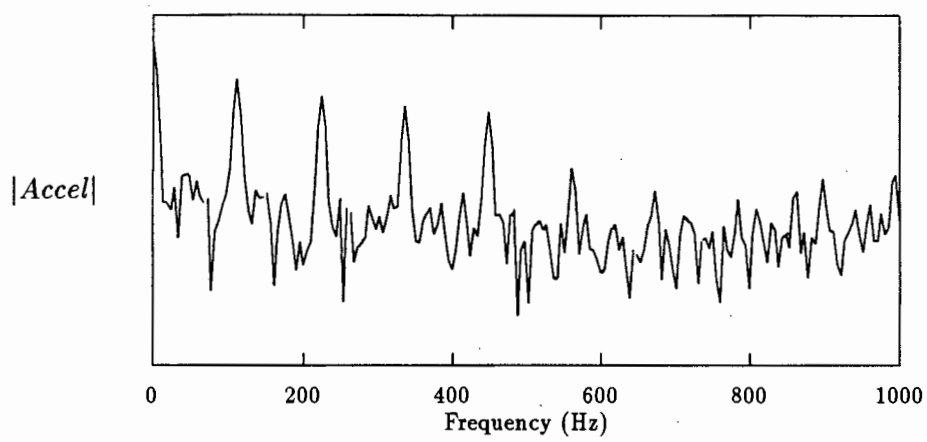


Figure 3.26: SDB 5, Envelope Spectrum

The Y^4 spectrum shows some uneven sidebands around the 110Hz components.

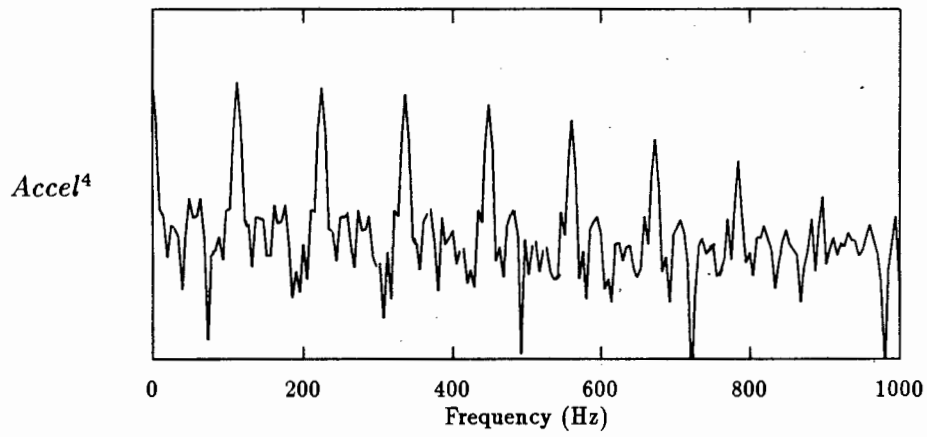


Figure 3.27: SDB 5 Spectrum of Acceleration to the Fourth Power

The BPFO component in the cepstrum, Fig 3.28, is at 8.9 ms as expected and is of similar amplitude to that of Spark Damaged Bearing 1.

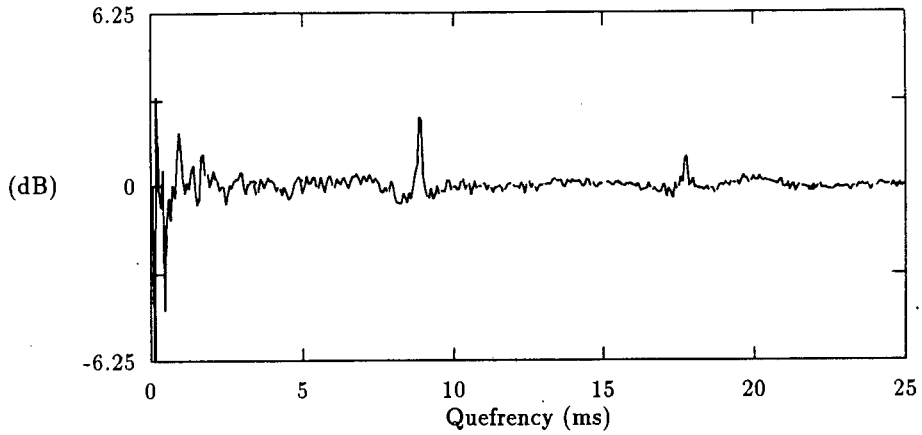


Figure 3.28: SDB 5, Cepstrum

3.13 The Graphs of Brinelled Bearing 1

These graphs are from the first test on bearings damaged by running under extreme loading. This bearing was accidentally brinelled while the rig was being set up. This left a series of small indentations around the outer race of the bearing. It is anticipated that these will give rise to symptoms of outer race damage.

For these tests, the shaft speed was 1800 RPM. The load was 22 kN and the low frequency cut-off 3 Hz. The characteristic frequencies are therefore:

R 28.33 Hz

FTF 12.04 Hz

BPFO 192.7 Hz

The kurtosis is slightly higher than for an undamaged bearing, at a value of 3.4

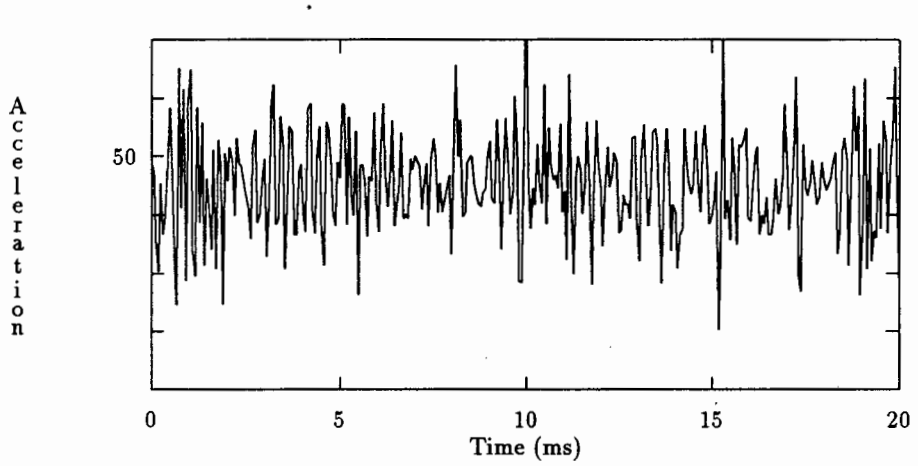


Figure 3.29: Brinelled Bearing Time Signal

Harmonics of 185Hz (outer race ball pass frequency) are visible throughout the spectrum. There are also frequency components at approximately 90 Hz spacing in the region below 1 kHz. It is not apparent whether these are subharmonics, or caused by some independent effect.

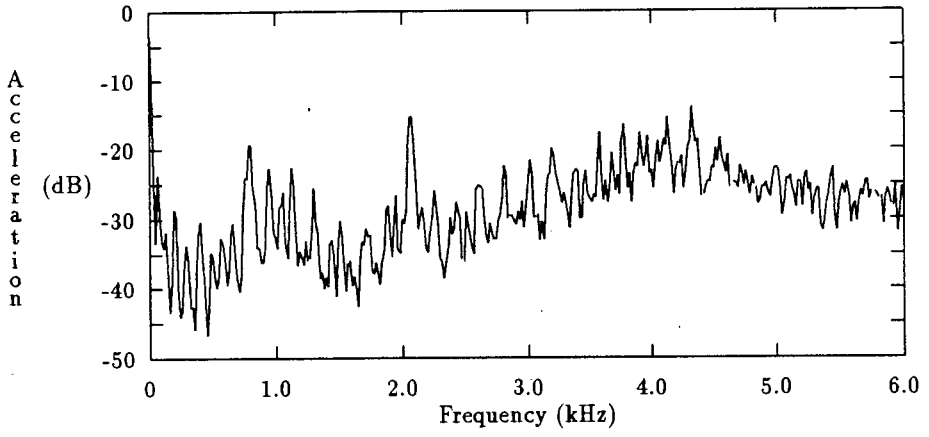


Figure 3.30: Brinelled Bearing Power Spectrum

The envelope spectrum does not show any components at the ball pass frequency, or any other frequencies.

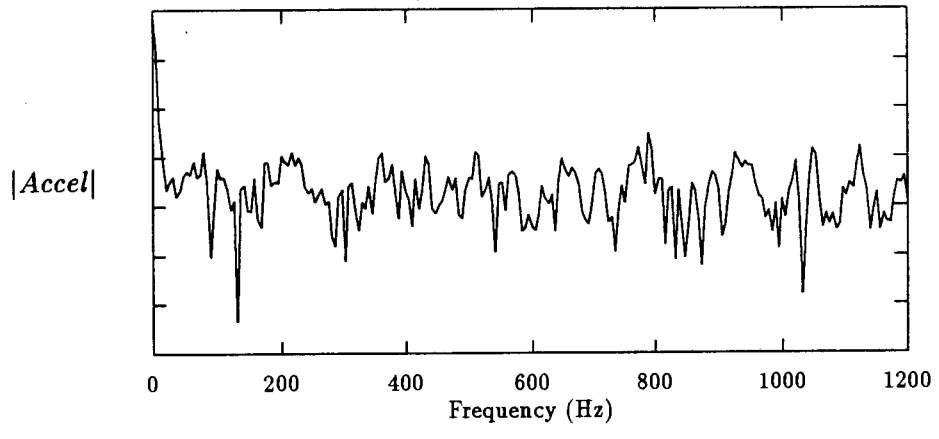


Figure 3.31: Brinelled Bearing Envelope Spectrum

The y^4 spectrum, like the envelope spectrum shows no noticeable components.

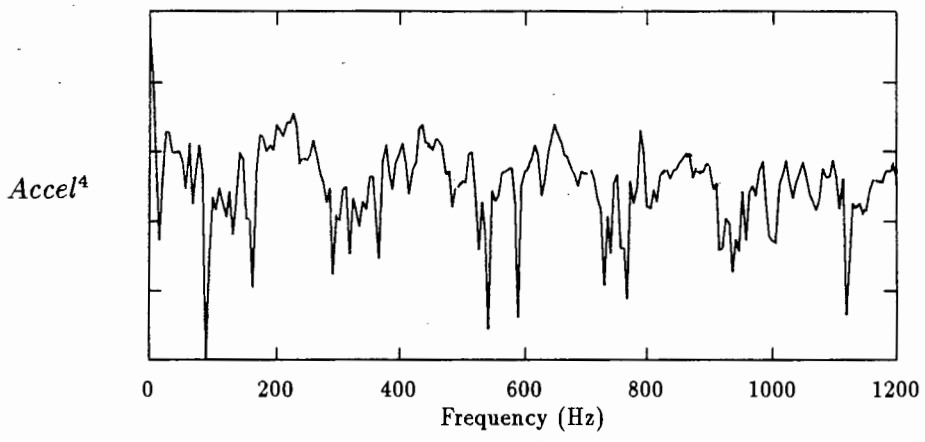


Figure 3.32: Brinelled Bearing Spectrum of Acceleration to the Fourth Power

There is a definite peak in the autocorrelation plot, (Fig 3.33) at 5.31 ms, which is equivalent to 188 Hz. This should be compared to the cepstrum (Fig 3.34). The cepstrum has a similar component, but in the cepstrum it is narrower, with less oscillation on either side of it.

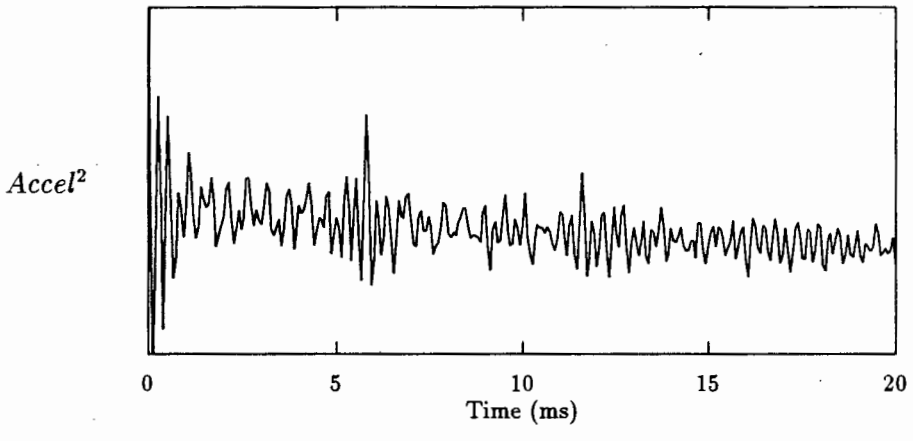


Figure 3.33: Brinelled Bearing, Autocorrelation

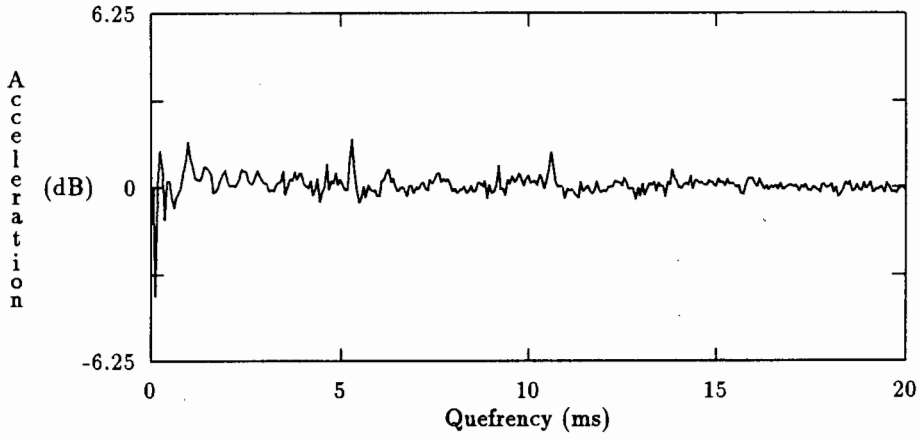


Figure 3.34: Brinelled Bearing, Cepstrum

3.14 Graphs of Fatigue Damaged Bearing 1

The shaft speed was 1700 RPM, the load 22 kN. The expected characteristic frequencies are therefore:

R 28 Hz

FTF 12 Hz

BPFO 193 Hz

3.14.1 Graphs of Fatigue Test 1, Undamaged Bearing

At this stage the bearing had been running for 10 minutes and was still in good condition, with no ball or race damage. The kurtosis is normal at 3.0

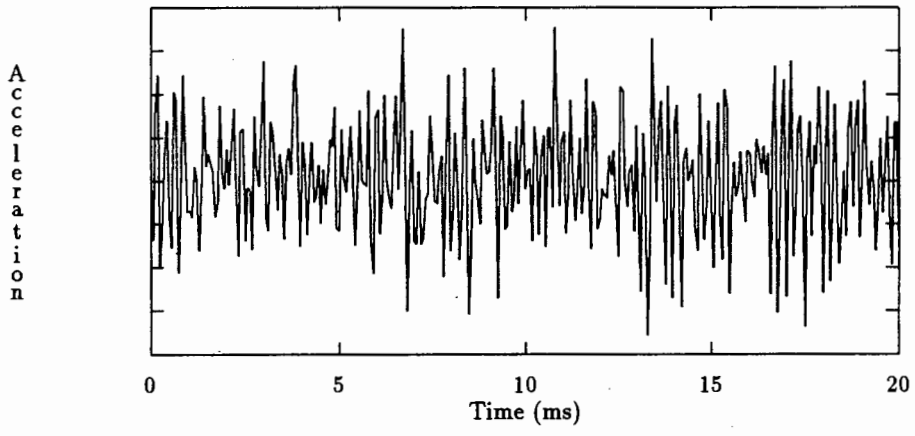


Figure 3.35: Fatigue Damaged Bearing 1u Time signal

In the lower portion of the power spectrum there are visible spectral components at 28 Hz (Rotational Frequency) and its first three harmonics. There are also spectral components at 200 and 401 Hz, which correspond to the expected BPFO and its second harmonic. The fact that this is slightly higher than the expected 193 may be due to a non zero contact angle, or variation of the shaft speed above the nominal value. At higher frequencies, between 2 kHz and 4 kHz a series of harmonics spaced 163 Hz apart are visible. These are not at any expected passage frequency, and may be caused by the support bearings, the drive motor or the cooling fan (The cooling fan runs during all the tests, but the vibrations generated are usually much lower than those of the damaged bearings). These vibrations are also apparent in the power spectrum of the second fatigue test, Figure 3.51 and the subject is discussed in more detail in that section. There appears to be broad band noise some sort between 4 and 5 kHz.

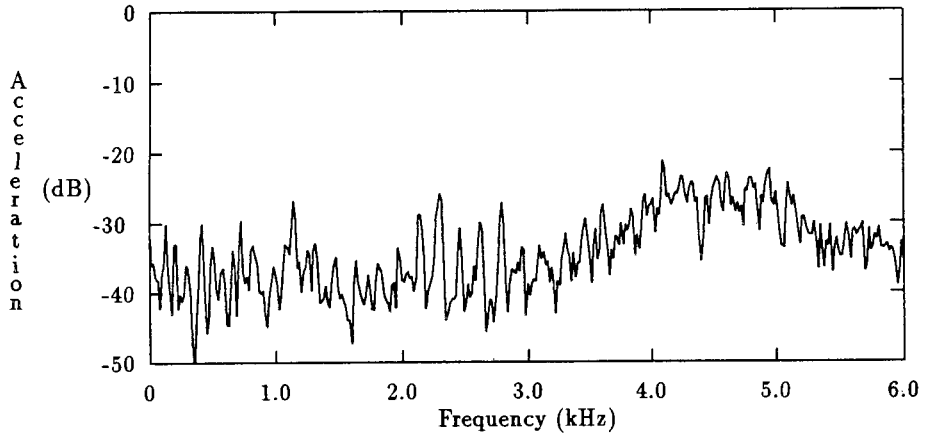


Figure 3.36: Fatigue Damaged Bearing 1u Power Spectrum

There are no obvious harmonic components in Figure 3.37 the envelope spectrum.

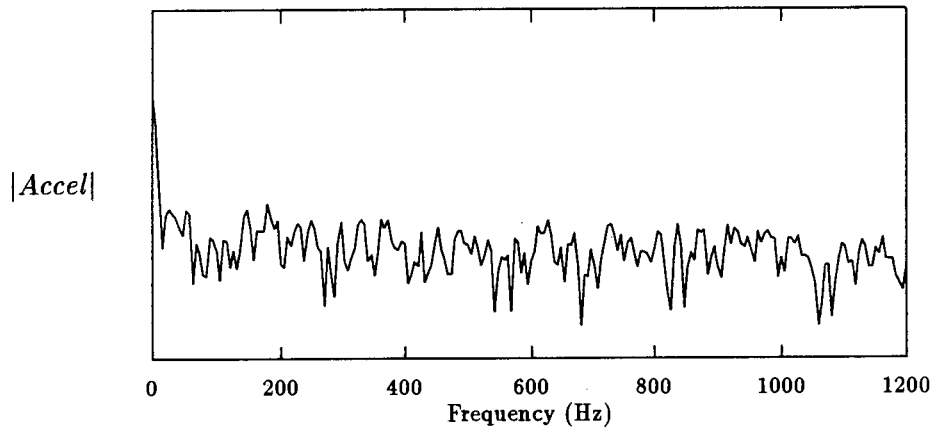


Figure 3.37: Fatigue damaged bearing 1u, Envelope Spectrum

The y^4 spectrum, displayed in figure 3.38, does not show any components near expected ball passage frequencies, but there are lower frequency components of 28 Hz and its harmonics. These may be from the rotational frequency.

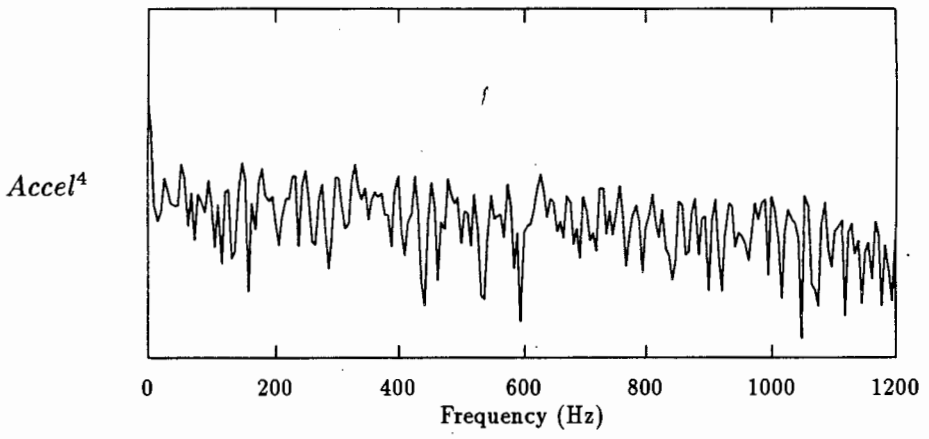


Figure 3.38: Fatigue Damaged Bearing 1u, Spectrum of Acceleration to the Fourth Power

The cepstrum, Fig 3.39 has a component at 6.1 ms, corresponding to the observed 163 Hz spaced peaks at around 2kHz in the power spectrum. The amplitude is low, due to the low energy in the spectrum

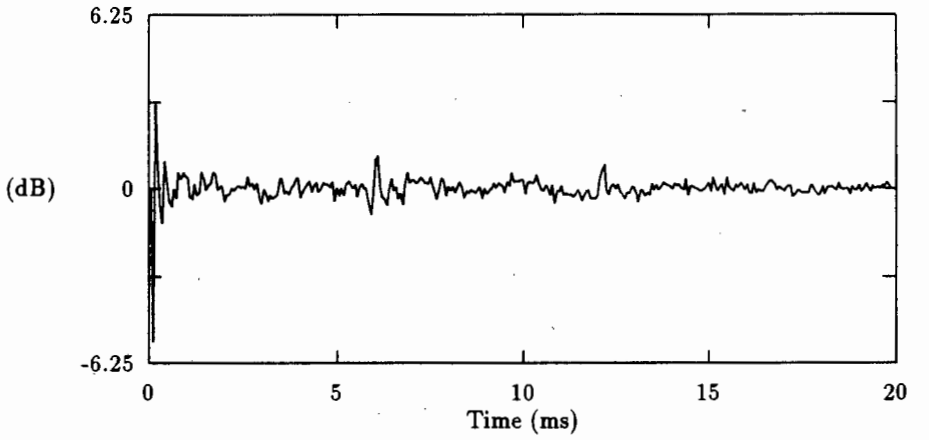


Figure 3.39: Fatigue Damaged Bearing 1u, Cepstrum

3.14.2 Graphs of Fatigue Damaged Bearing 1, Moderate Damage

This capture was made after running for 35 minutes, and the first spall pit had appeared. There may also have been small metal particles in the oil. There are clear impulses in the time signal for this capture. There also appears to be a strong sinusoidal component at *approx*imately 800 Hz. The kurtosis for this signal is 2.6 which is lower than normal, due to the strong sinusoidal component.

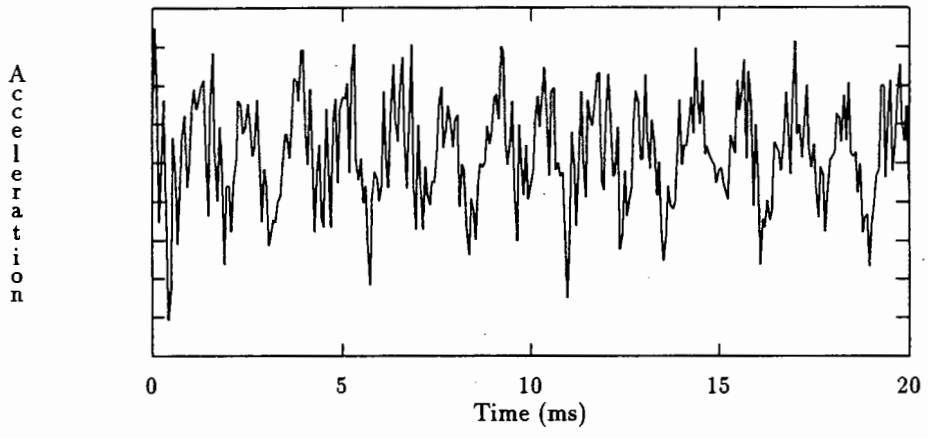


Figure 3.40: Fatigue Damaged Bearing 1m, Time Signal

The power spectrum shows harmonics of 190 Hz (BPFO). The fourth harmonic appears to dominate, causing the sinusoidal appearance of the time signal. Several harmonics of 27 Hertz (Shaft frequency) appear in the lower part of the spectrum, there are also sidebands around the BPFO harmonics. It should be noted that the dominant fourth harmonic is close to 1 kHz, the frequency of the ringing in the spark damaged bearings. This may be a result of the same resonances.

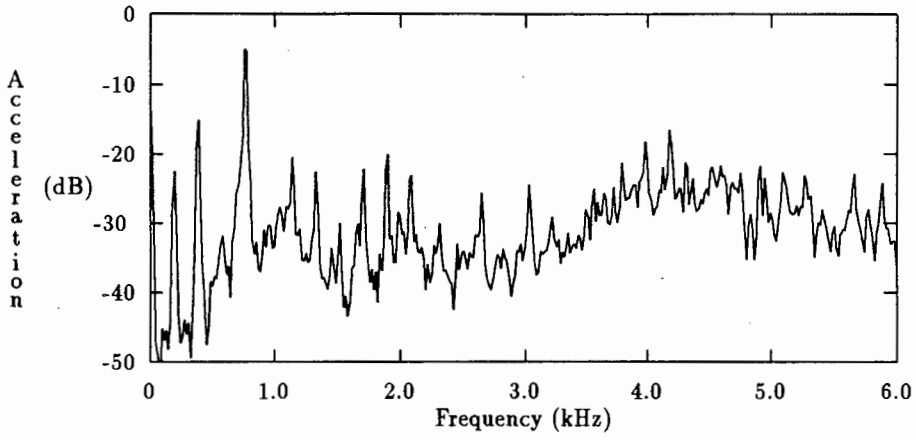


Figure 3.41: Fatigue Damaged Bearing 1m, Power Spectrum

The envelope spectrum (Fig 3.42) has a number of BPFO harmonics, the first harmonic is of lower amplitude than the others.

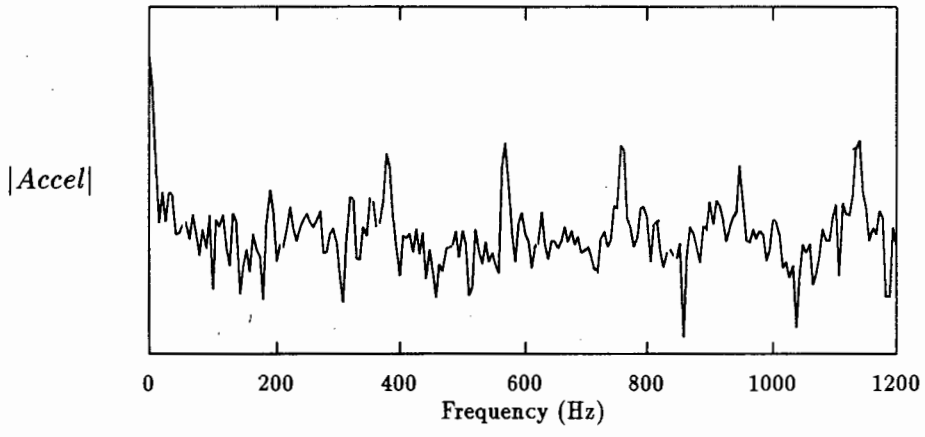


Figure 3.42: Fatigue Damaged Bearing 1m, Envelope Spectrum

The Y^4 spectrum also shows a number of BPFO harmonics, with the first being of slightly lower amplitude than the others, but is still clearly visible. Some sidebands are visible around the BPFO harmonics.

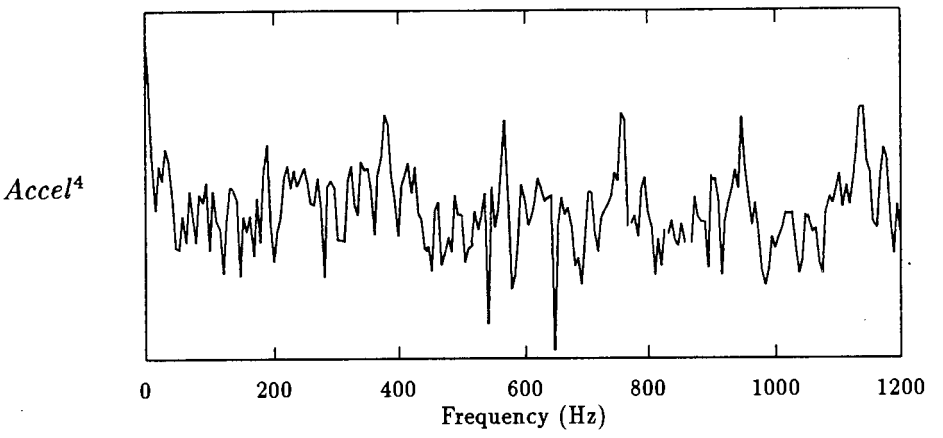


Figure 3.43: Fatigue Damaged Bearing 1m, Spectrum of Acceleration to the Fourth Power

The cepstrum (Fig 3.44) has a component at 5.3 ms, as expected from the 190 Hz and harmonics in the power spectrum.

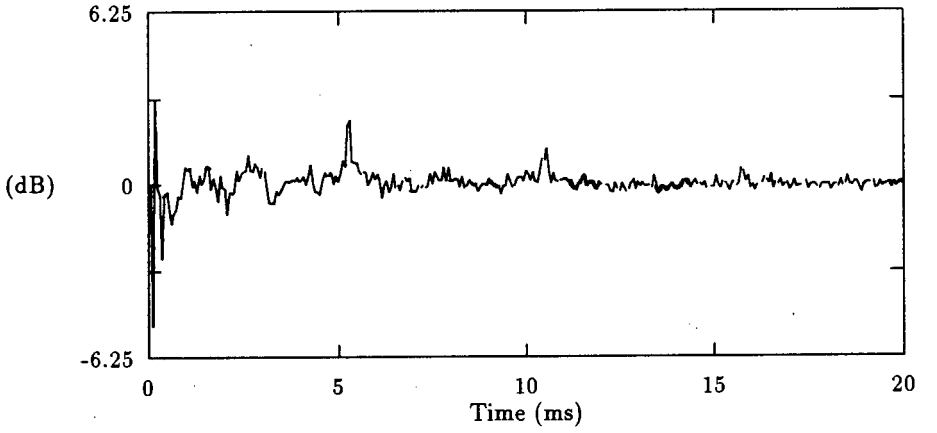


Figure 3.44: Fatigue Damage Bearing 1m, Cepstrum

3.14.3 Graphs of First Fatigue Test, Advanced Damage

The bearing had run for 1 Hr and 49 minutes and was badly damaged at this stage, with severe pitting of both sides of the outer race. There were also a large number of metal particles in the oil. The bearing had been photographed after running for 1 Hr and 23 Min, these are the photographs shown in figures 3.1 and 3.2 The time signal shows a large number of impulses, close together, with approximately 1.4 ms spacing. It is not clear whether they are actually periodic, or if there is another overlying periodicity. There is very little ringing, and Kurtosis is very high, with a value of 7.6.

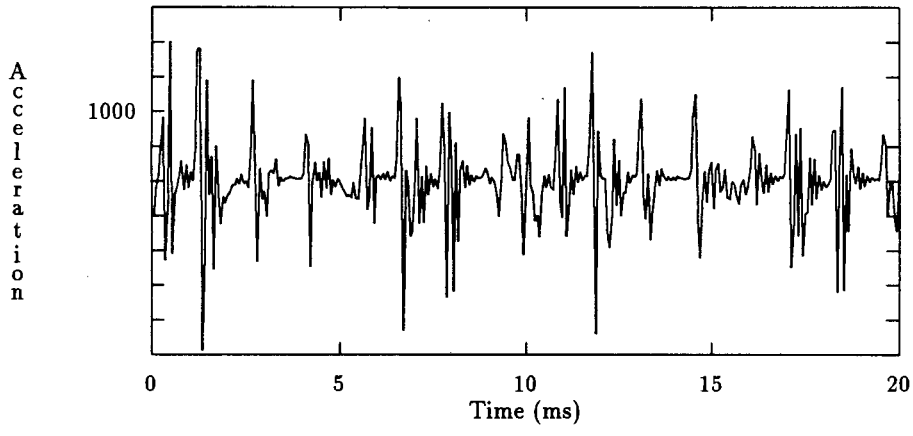


Figure 3.45: Fatigue Damaged Bearing 1a, Time Signal

There is a 190 Hz component in the power spectrum, and its first four harmonics are visible. No other higher harmonics are visible, but there is a great amount of high frequency noise. Some low frequency components at 36 and 72 Hz are apparent these may be multiples of the FTF, or possible interference.

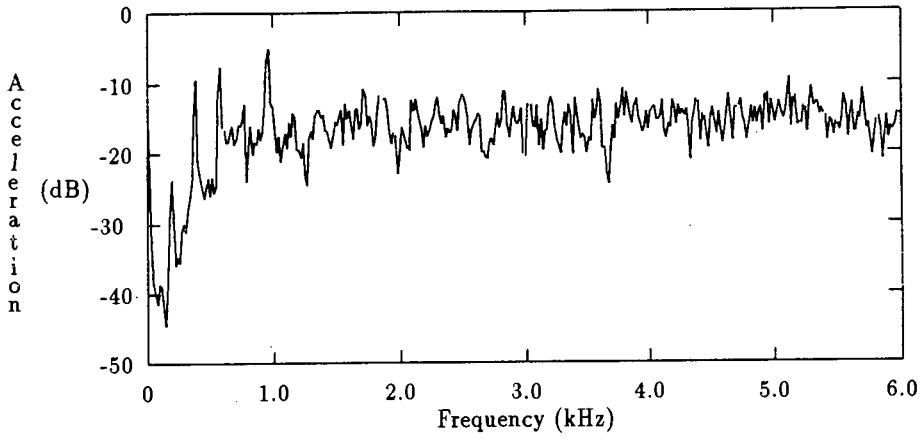


Figure 3.46: Fatigue Damaged Bearing 1a, Power Spectrum

The Envelope spectrum shows a 190 and 380 Hz components, but no other clear frequencies.

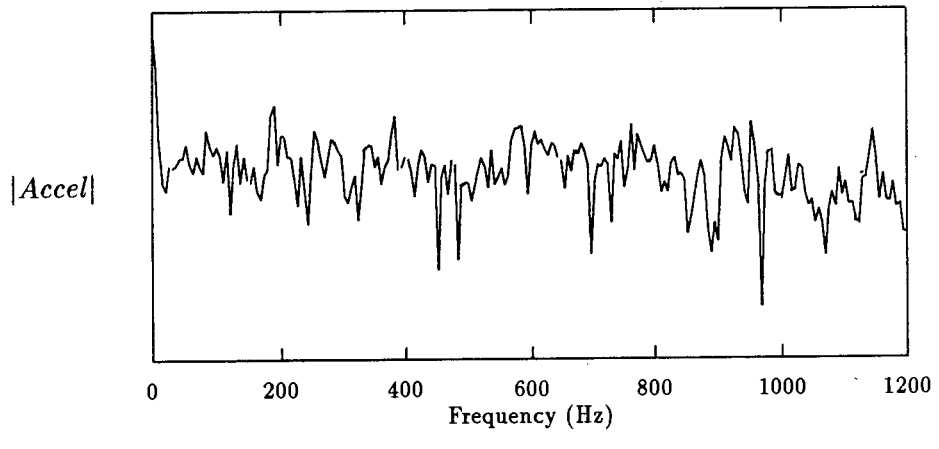


Figure 3.47: Fatigue Damaged Bearing 1a, Envelope Spectrum

The fourth power spectrum has a 190 Hz component, and a large number of harmonics spaced (very approximately) 16 Hz apart .

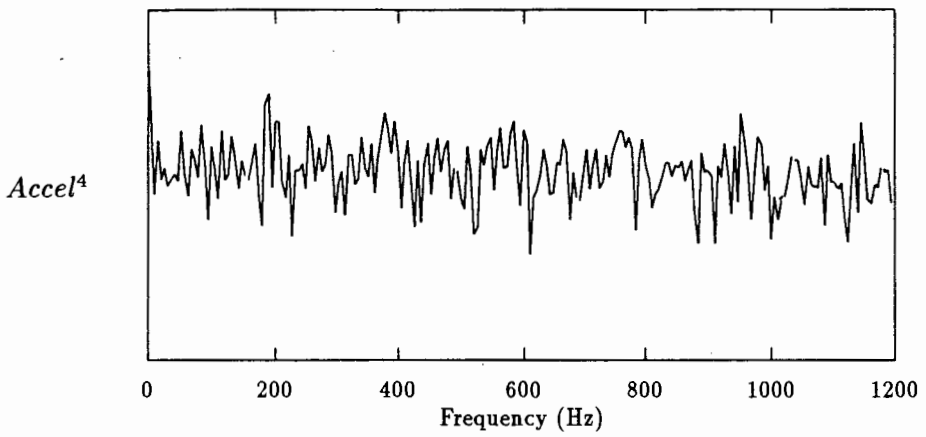


Figure 3.48: Fatigue Damaged Bearing 1a, Spectrum of Acceleration to the Fourth Power

The cepstrum rahmonic at 5.3 ms is not very large. There is also a small component at 8.3 ms (120 Hz) and one at 12 ms (83 Hz). The important fact to note is that while the cepstral component is of similar size to that seen in the case of the undamaged bearing, it is at the expected defect quefrency of 5.3 ms and not at the 6.1 ms in figure 3.39.

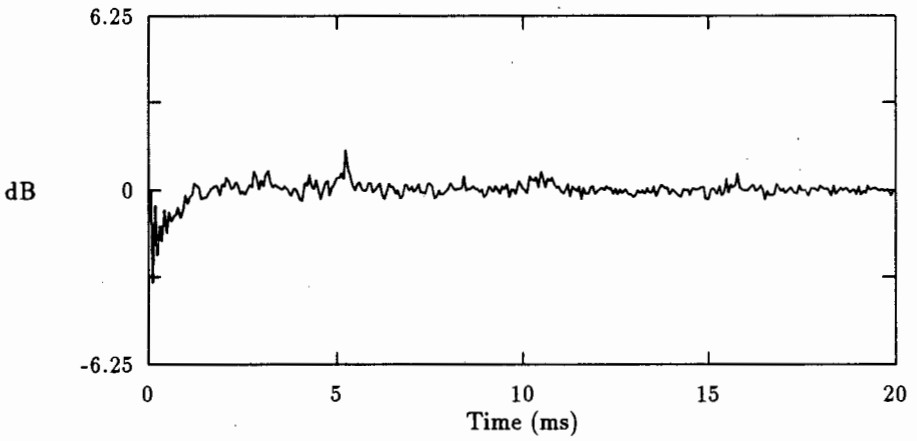


Figure 3.49: Fatigue Damaged Bearing 1a, Cepstrum

3.15 Graphs of the Second Fatigue Test

The second test was run in the same way as test one. The shaft speed was 1700 RPM, the load 22 kN. The expected characteristic frequencies are therefore:

R *28 Hz*

FTF 12 Hz

BPFO 193 Hz

3.15.1 Graphs of Second Fatigue Test, Undamaged Bearing

The first results are for an undamaged bearing, after running for 12 minutes.. The kurtosis is 3.2. and is thus still in the range expected of an undamaged bearing.

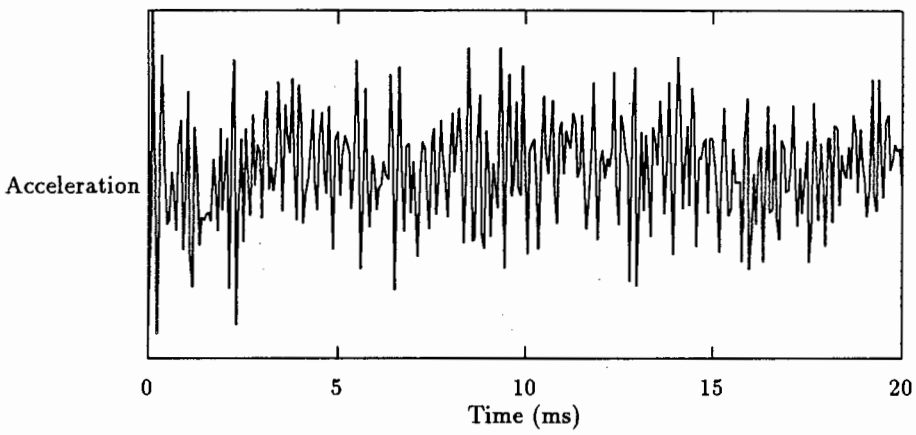


Figure 3.50: Fatigue Damaged Bearing 2u, Time Signal

The power spectrum (Figure 3.51)shows some peaks spaced 166 Hz apart in the 2kHz to 3kHz region. This is lower than the expected ball passage frequency (Peaks at the expected frequency are visible in figure 3.58) and harmonics of the ball passage frequency are not expected from a new bearing. These harmonics are also visible in the spectrum of the adjacent bearing, and in fact are even more prominent in the latter spectrum. For these reasons it is reasonable to suspect that they are not caused by the bearing under test, but come from some other source, possibly the adjacent support bearing, or the drive motor. Some improvement can be gained by adaptive filtering as is shown in subsection 3.15.2 page 74.

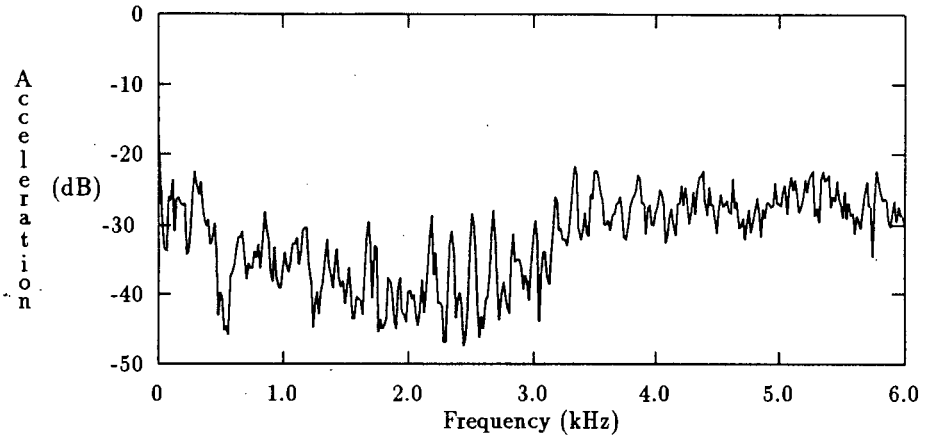


Figure 3.51: Fatigue Damaged Bearing 2u, Power Spectrum

The envelope and fourth power spectra show no obvious frequency components of harmonics, not even those visible in the power spectrum. The cepstrum (Figure 3.54) has a small peak at 6ms, corresponding to the spectral components visible between 2 and 4 kHz.

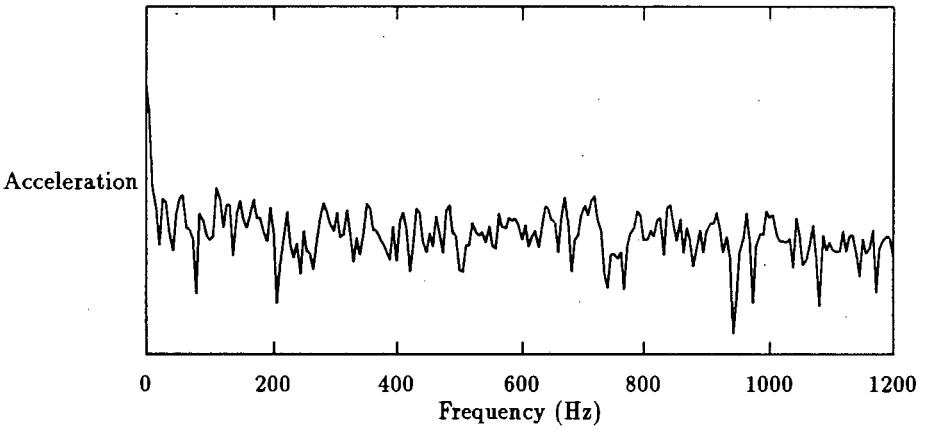


Figure 3.52: Fatigue Damaged Bearing 2u, Envelope Spectrum

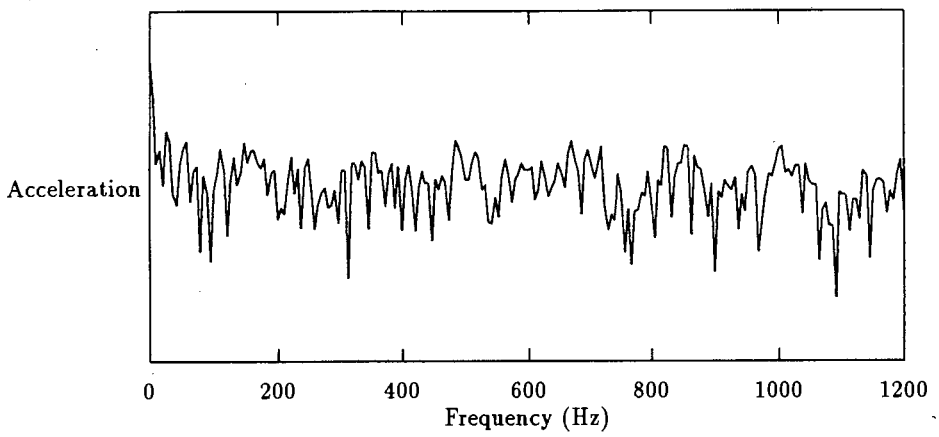


Figure 3.53: Fatigue Damaged Bearing 2u, Spectrum of Acceleration to the Fourth Power

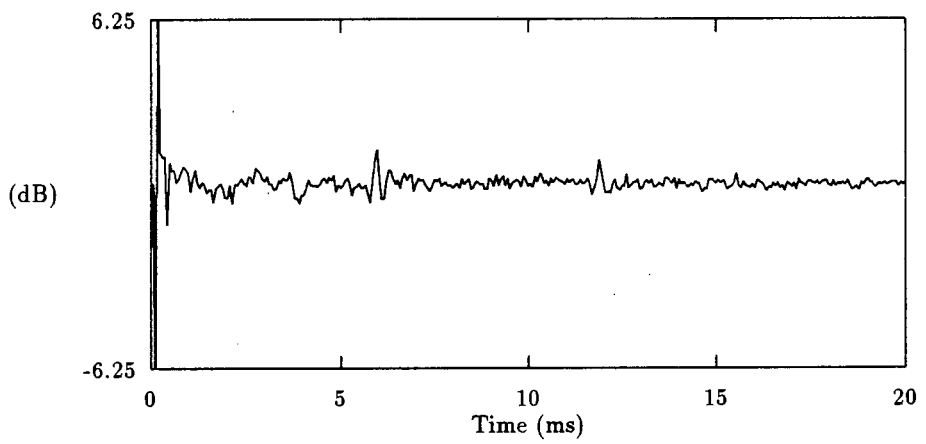


Figure 3.54: Fatigue Damaged Bearing 2u, Cepstrum

3.15.2 Adaptive Filtered Data of 2u

The data was run through an LMS adaptive filter using the vibrations from one of the support bearings as a reference. Some reduction of the interference is visible in the power spectrum (Figure 3.55) when compared with the unfiltered power spectrum, Figure 3.51. The 166 Hz harmonics have been reduced by approximately 5dB. It should be noted that some of the frequency components visible in Figure 3.51 are still present at the same amplitude.

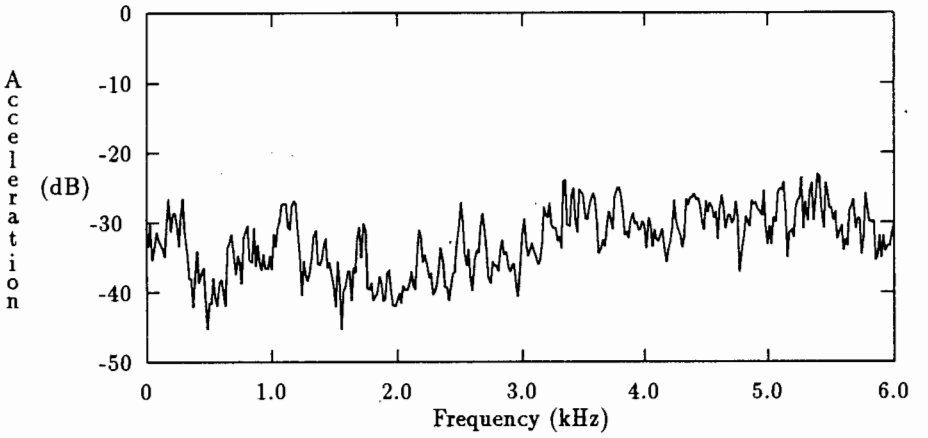


Figure 3.55: Power Spectrum of Adaptive Filtered Data

The peak that was visible in the cepstrum is no longer present, showing the reduction in the level and number of the 166 Hz harmonics.

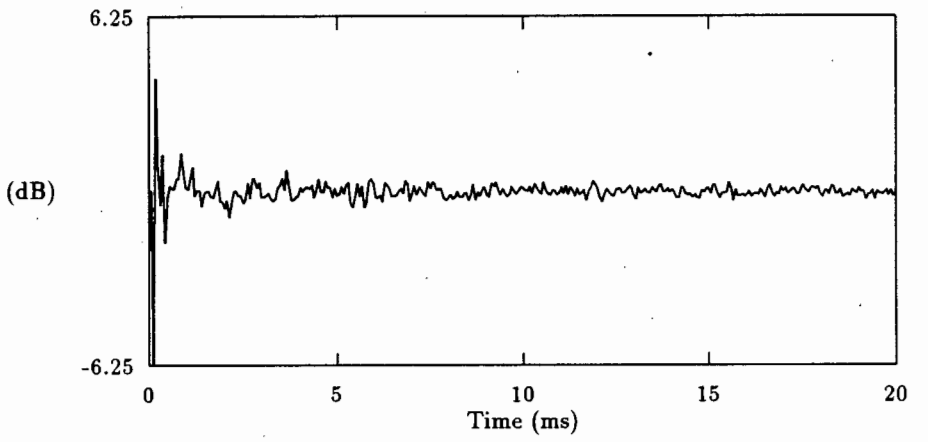


Figure 3.56: Cepstrum of Adaptive Filtered Data

3.15.3 Graphs of Second Fatigue Test, Moderate Damage

The bearing had run for 2 Hrs and 55 Minutes. Some damage has occurred, with kurtosis increased to 4.3, a clear indication of the damage. The running time for the first signs of damage to occur was much longer than for FDB1, and these results are more consistent with *the 2 Hrs* predicted by the bearing life equation.

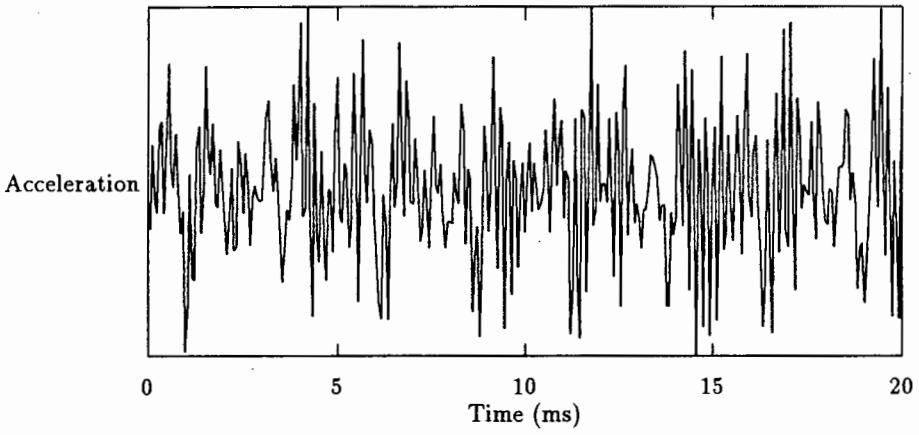


Figure 3.57: Fatigue Damaged Bearing 2m, Time Signal

Harmonics of 190 Hz spacing occur in the 500 Hz to 3kHz region. These are at the expected outer race ball passage frequency.

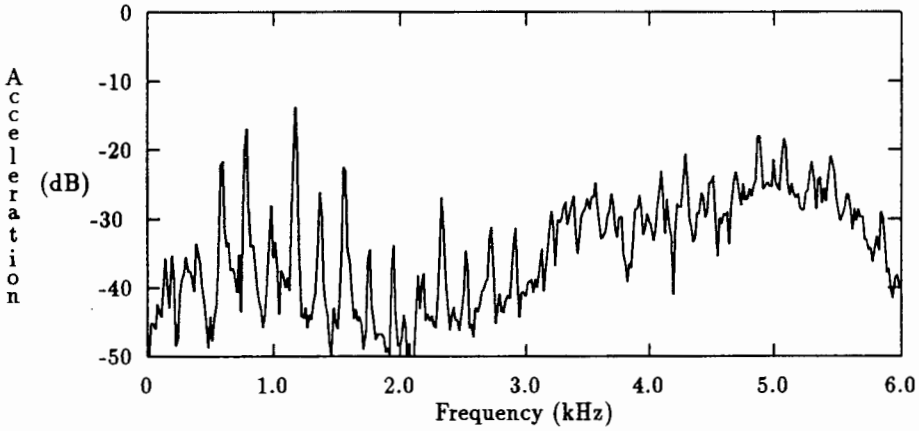


Figure 3.58: Fatigue Damaged Bearing 2m, Power Spectrum

The envelope spectrum shows four harmonics of 190 Hz, the second has the largest amplitude, approximately the same as that of the harmonics in the envelope spectrum of FDB 1m, the others being much lower amplitude.

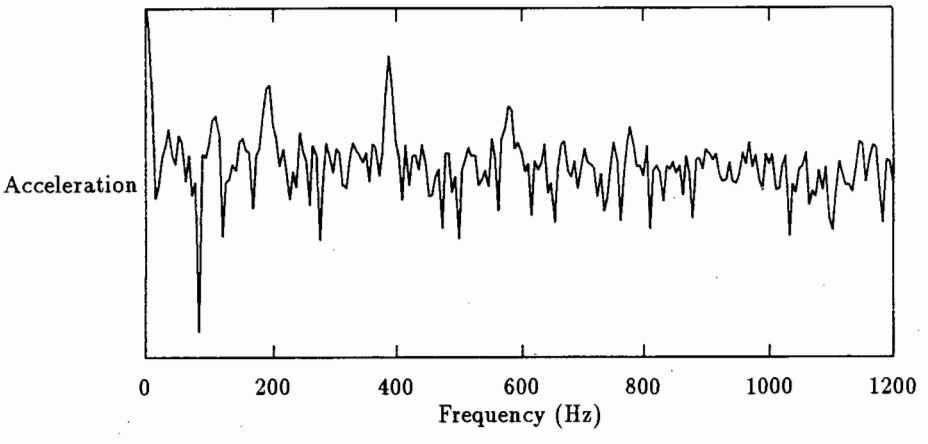


Figure 3.59: Fatigue Damaged Bearing 2m, Envelope Spectrum

The y^4 spectrum shows slightly more harmonics than the envelope spectrum, but they are also of lower amplitude than those in FDB 1m.

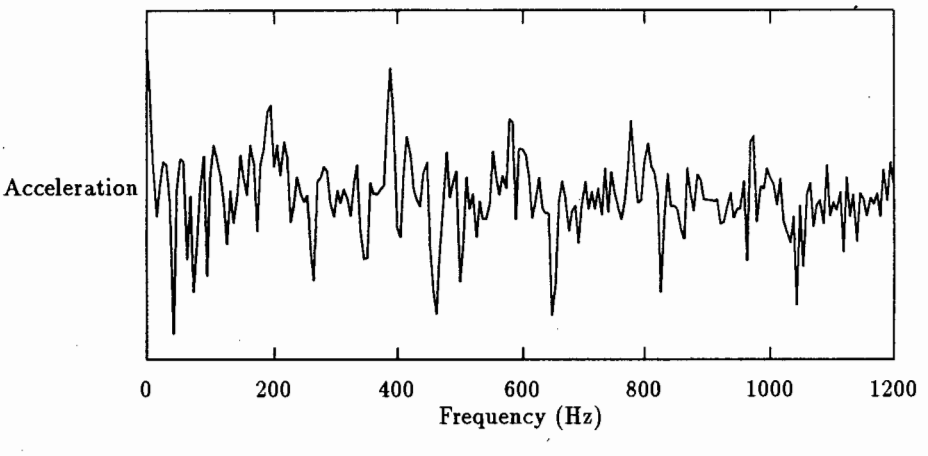


Figure 3.60: Fatigue Damaged Bearing 2m, Y^4 Spectrum

The cepstrum has a prominent 5.1 ms component. These results are very similar to the moderately damaged bearing in fatigue test 1.

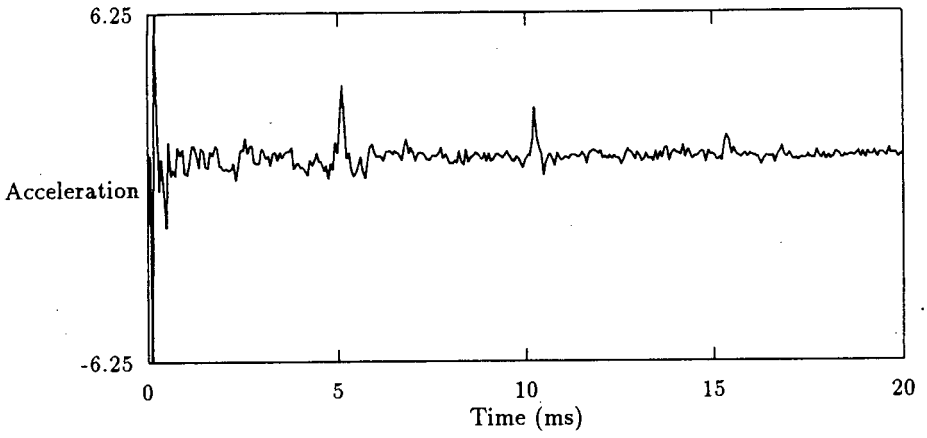


Figure 3.61: Fatigue Damaged Bearing 2m, Cepstrum

3.15.4 Graphs of Second Fatigue Test, Advanced Damage

The bearing had run for 4 Hrs and 12 minutes. The damage is advanced at this stage, but the kurtosis is lower than normal, at 2.7 This may be a result of a dominant harmonic or sinusoidal component.

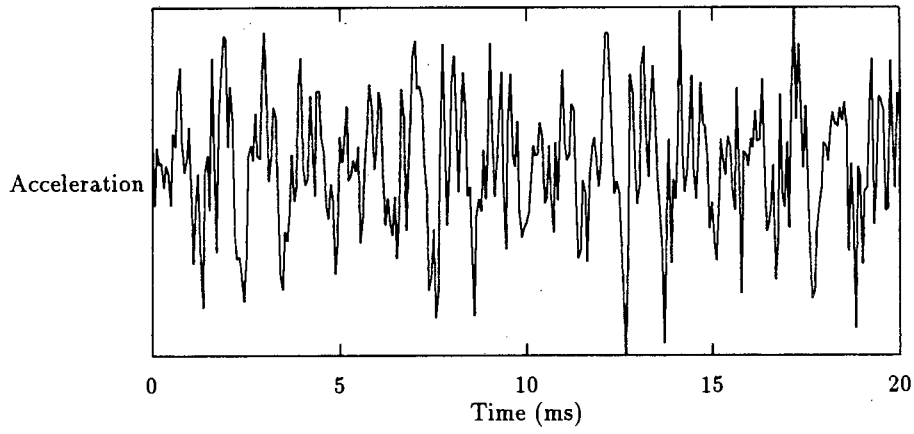


Figure 3.62: Fatigue Damaged Bearing 2a, Time Signal

The damage is advanced at this stage, there are 190 Hz harmonics throughout the spectrum. The components around 1 kHz are particularly prominent and may be responsible for the low kurtosis.

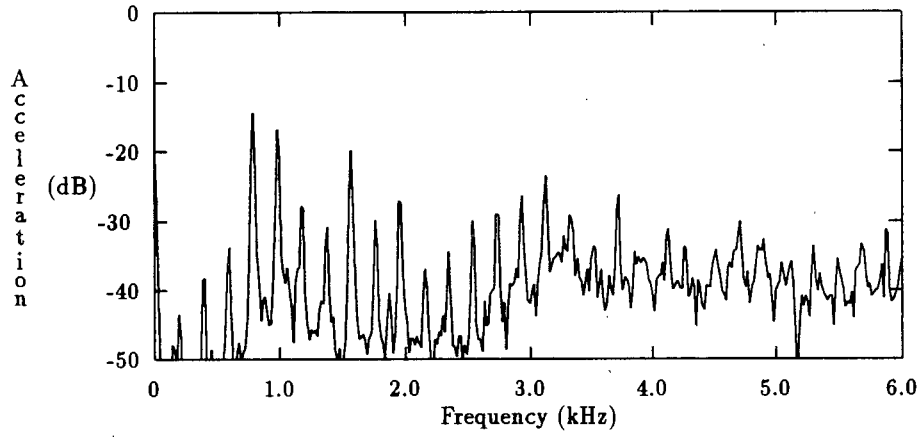


Figure 3.63: Fatigue Damaged Bearing 2a, Power Spectrum

The envelope and y^4 spectra also show prominent 190 Hz harmonics. These are of greater amplitude than those of the moderately damaged bearing, and are of similar amplitude to those in the envelope and y^4 spectra of FDB 1m.

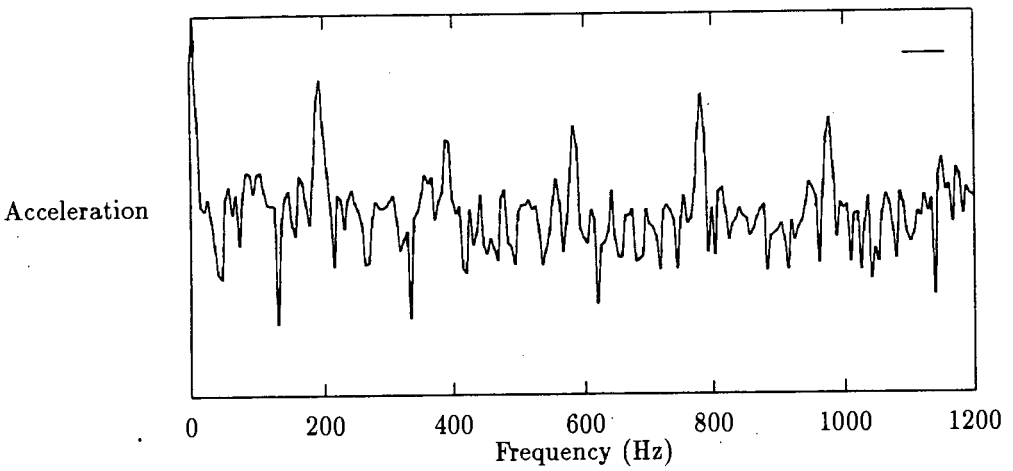


Figure 3.64: Fatigue Damaged Bearing 2a, Envelope Spectrum

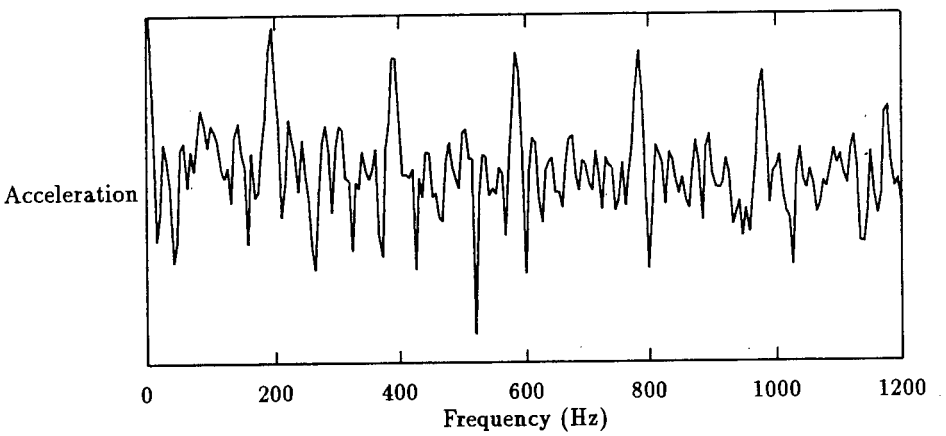


Figure 3.65: Fatigue Damaged Bearing 2a, Y^4 Spectrum

The cepstrum (figure 3.66) has a very prominent rahmonic at 5.1 ms, the cepstrum equivalent of the 190 Hz harmonics in Figure 3.63. The large number of harmonics in the power spectrum being the reason for the large rahmonic.

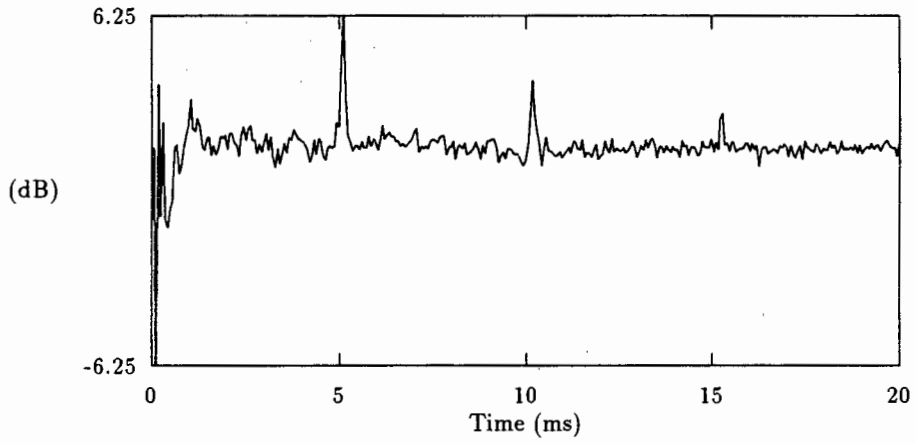


Figure 3.66: Fatigue Damaged Bearing 2a, Cepstrum

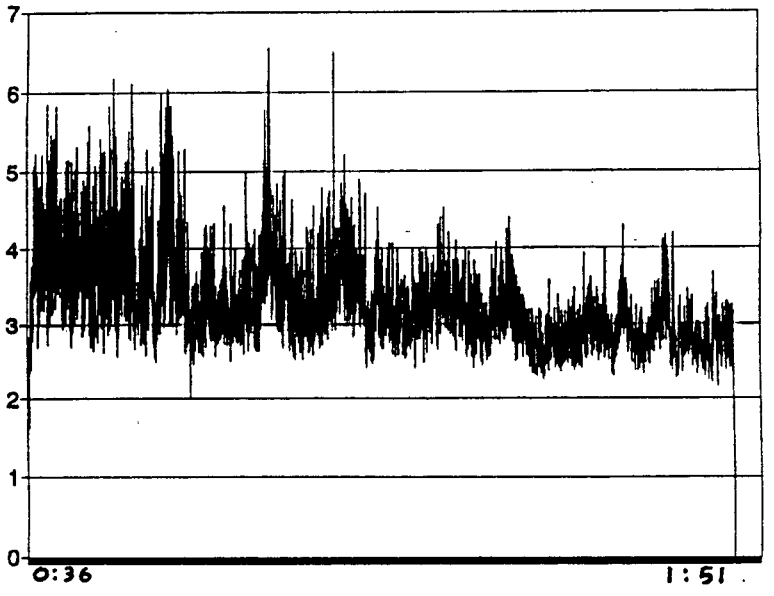


Figure 3.67: Kurtosis of Fatigue Damaged Bearing 2

The kurtosis value was calculated from data captured using the DSP card and saved to a file. Figure 3.67 is a graph of the kurtosis recorded during the last 1Hr 15 min of test 3. The kurtosis value can be seen to decrease as the level of damage increases and the larger number of defects gives a more random vibration. The ADC sampling rate was 50 kHz. This gave a wider bandwidth than the measurements using the spectrum analyser, and for this reason the different kurtosis values are not entirely comparable.

3.16 Comparison of Spectra as Damage Progresses

In the previous sections, the plots of time domain signal, power spectrum cepstrum etc were placed together to illustrate how the different types of analysis highlighted different aspects, of the vibration. In the following section, the power spectrum and cepstrum will be examined to see how these change as damage progresses.

3.16.1 The Progress of Damage as Visible in the Spectrum

If we consider the power spectrum plots of fatigue test 1, the indications of damage appear in the spectrum in the following manner.

Initially, the spectrum is flat, as can be seen in the first part of figure 3.68. The peaks present are those from sources of vibration other than the test bearing. Once damage occurs, a large number of harmonics of the defect frequency are present in the power spectrum. As the damage progresses, the harmonics of the defect frequency are less noticeable. Instead the dominant feature of the spectrum is the high frequency noise. This is due to the vibrations becoming more random as the number of defects increases and the number of metal particles in the lubricant increases.

The power spectrum plots of Fatigue Test 2 shows a similar, but not identical pattern. The undamaged bearing is similar to SDB1u with a flat spectrum, with only sources of interference visible. Once damage occurs, the harmonics of the defect frequency appear in the spectrum, as seen in the power spectrum of the moderately damaged bearing in test two. If the harmonic peaks are ignored, the underlying spectral shape is still similar to that of the undamaged bearing. The final part of the test shows more harmonic peaks, but it is also apparent that the high frequency noise level has increased in comparison with the frequencies below 1 kHz, as it did for the bearing with advanced damage in Fatigue test 1.

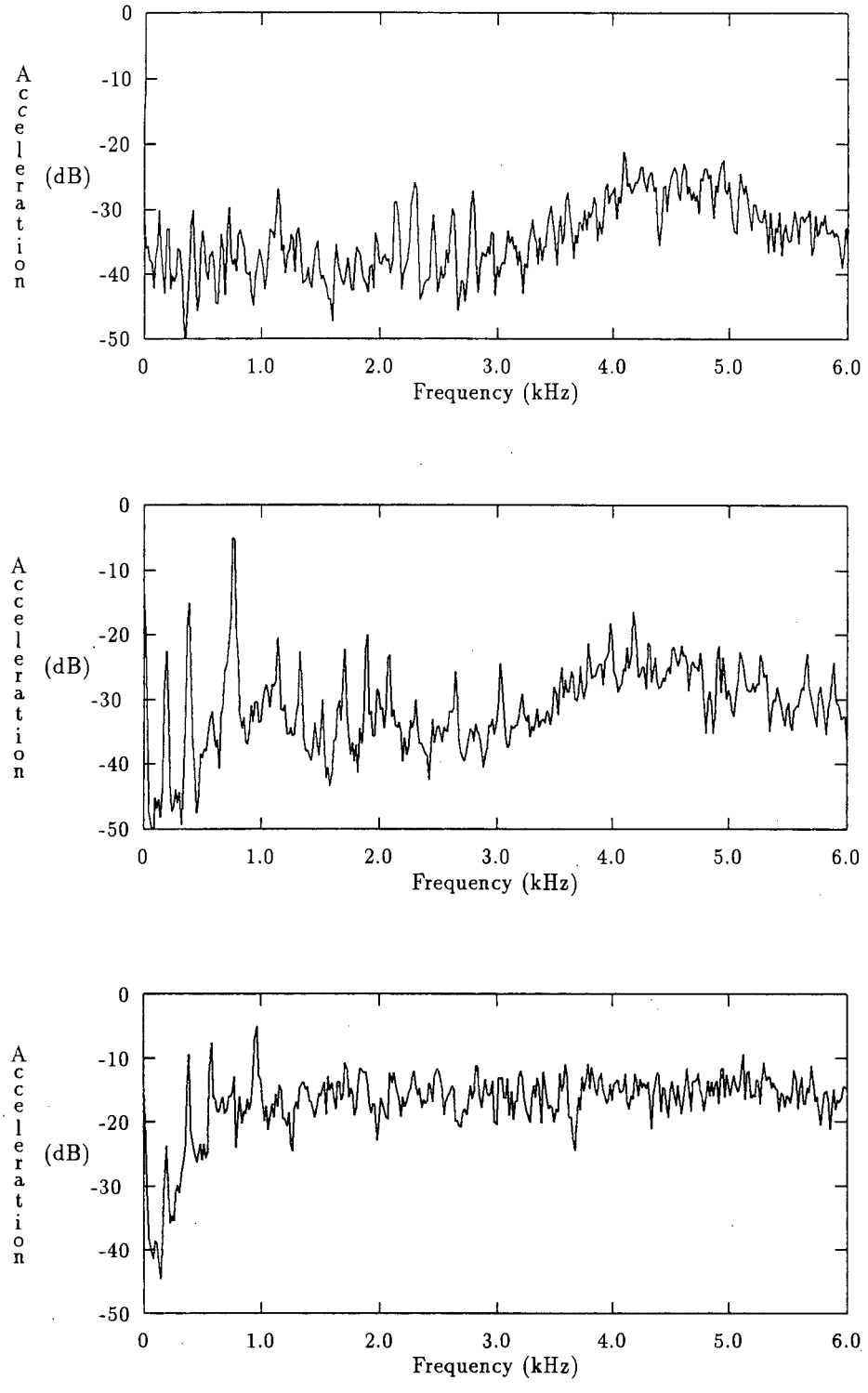


Figure 3.68: Fatigue Test One Power Spectra

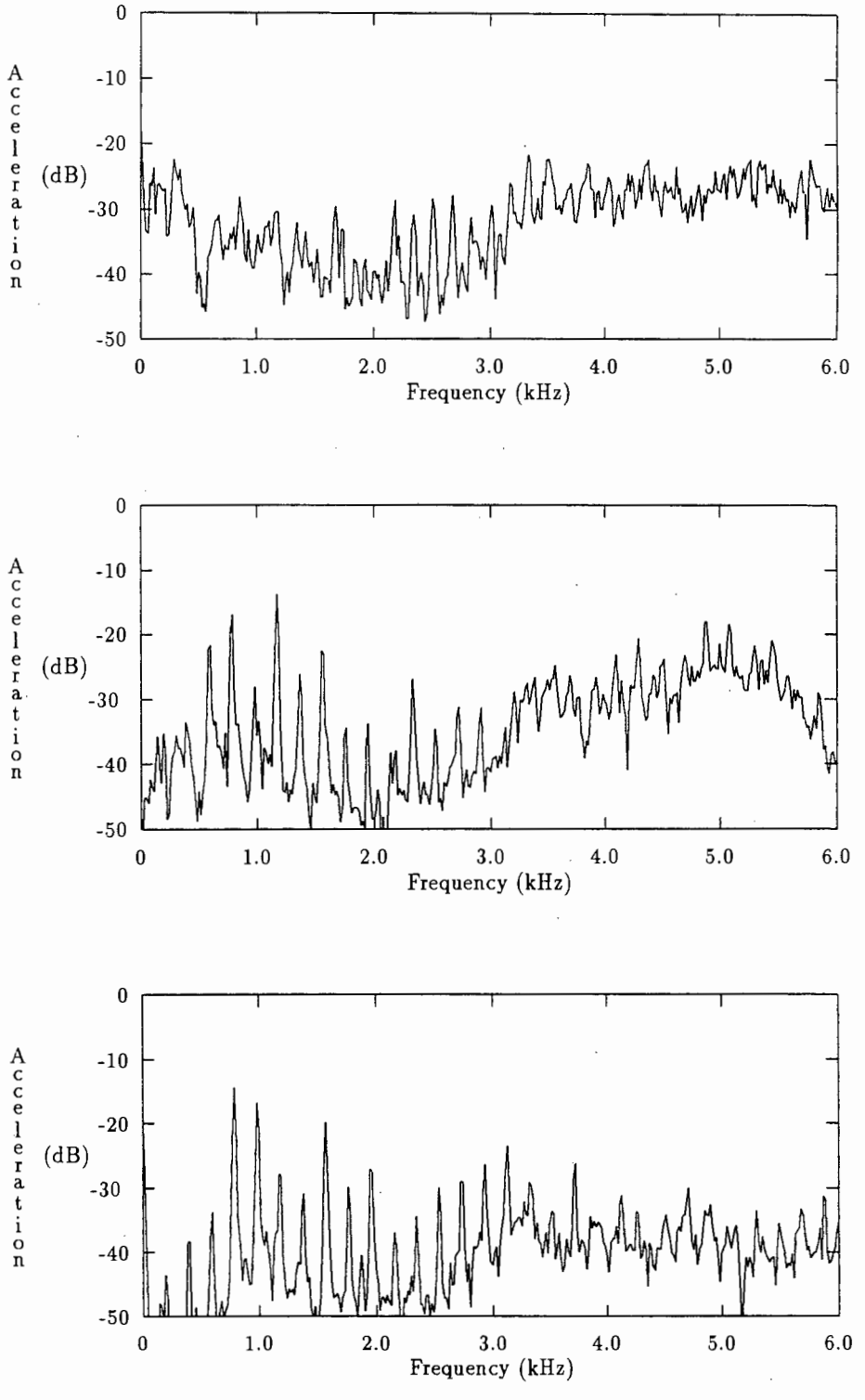


Figure 3.69: Fatigue Test Two Power Spectra

3.17 The Progress of Damage as Visible in the Cepstrum

In figures 3.70 and 3.71 the visible indications of damage proceed quite differently from a similar starting point. In Fatigue test 1, the cepstrum for the undamaged bearing shows a small peak. This peak also appears in the cepstrum of the undamaged bearing in Fatigue test two. It has been mentioned before that this peak is not at the expected ball passage frequency, and it is suspected that some other source of vibration may be the cause of this peak. When the level of damage reaches the moderate stage, it can be seen that the two bearings are starting to behave differently. A cepstral peak is visible in both cases, and is at the expected quefrency, but the peak in the graph of FDB2m is more prominent than that in FDB1m, due to the higher harmonic content. Once the damage has reached the advanced stage, FDB2m has a very pronounced cepstral peak, indicating the rise in the level and number of harmonics in the spectrum. The cepstrum of FDB1m on the other hand shows a decreased peak, of a size similar to that of the undamaged bearing. This indicates a quite different behaviour to FDB2, as there is a definite decrease in harmonic level.

3.17.1 The Cepstrum as an Indicator of Spectral Shape

While the higher quefrency portion of the cepstrum gives information about the harmonic content of the spectrum, the lower portion of the cepstrum, which is approximately 0 to 4 ms in this case, contains information relating to the shape of the spectrum. The cepstrum has been found to convert data relating to harmonic content of the spectrum to a very compact and easy to extract form, so the lower portion may be of use in compressing information about the spectral shape. The cepstra of the undamaged bearings and the moderately damaged bearings show some ringing in the first part of the cepstrum, which may indicate band pass or band stop effects in the spectrum. In order to interpret this, magnification of this section of the cepstrum would be necessary. The cepstra of the bearings with advanced damage show a dip a marked dip in the first part of the cepstrum. This is also visible to a lesser extent in the cepstrum of FDB1u. This is similar to the cepstrum of a high passes signal, and reflects the increase in high frequency random noise in the power spectrum. Further analysis of this portion of the cepstrum would be necessary to determine whether this information can be easily extracted from the cepstrum.

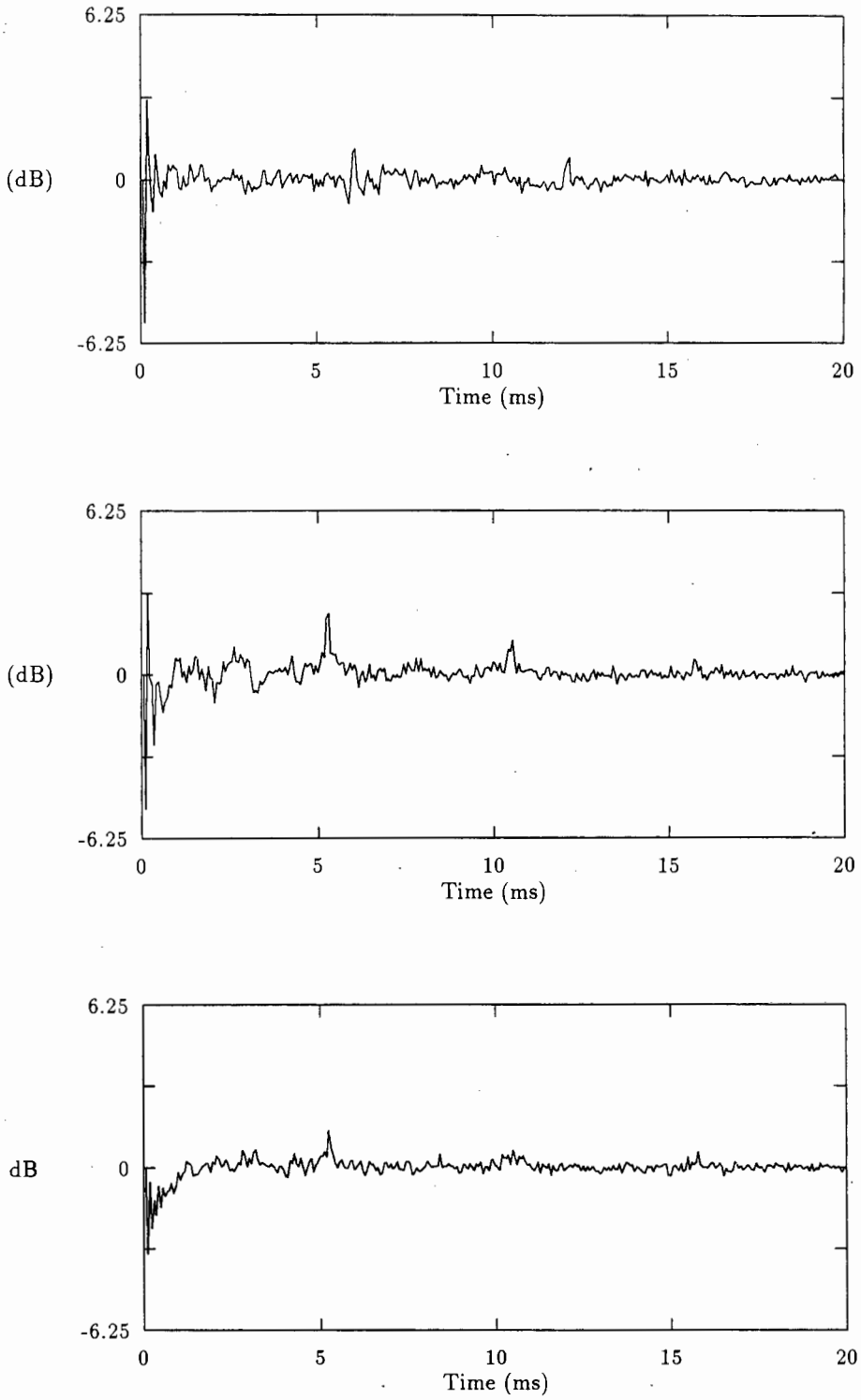


Figure 3.70: Fatigue Test One Cepstra

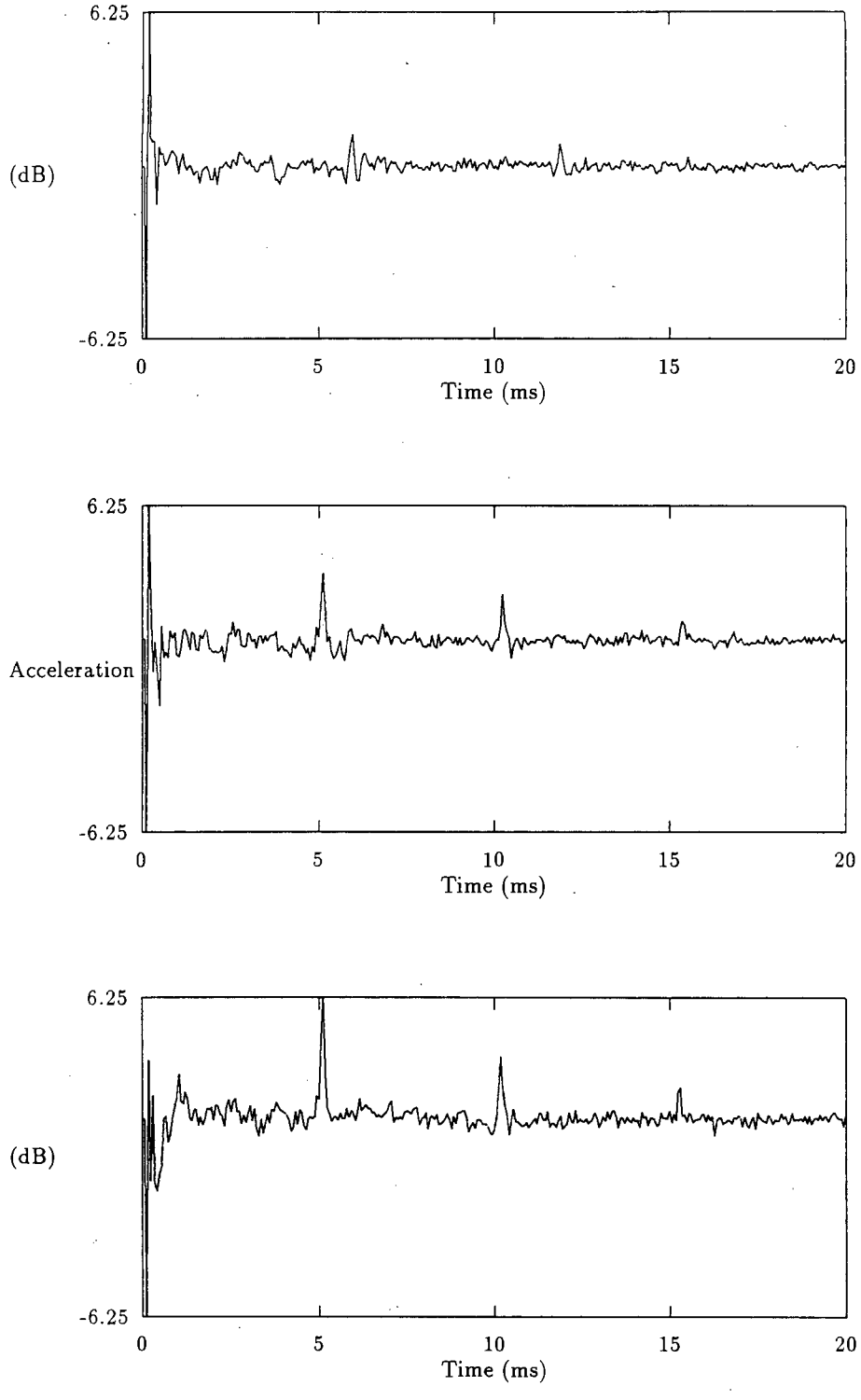


Figure 3.71: Fatigue Test Two Cepstra

3.18 Variation of Kurtosis with Increasing Damage

The kurtosis of Fatigue test 2 was shown in Figure 3.67, which is repeated below. The general trend of kurtosis as damage progresses is clearly visible. The kurtosis is initially close to 3.0, then once pitting occurs, the kurtosis rises, reaching values as high as 10.0 in this case. Actual values measured being dependent on the type of defect and the signal bandwidth. As the pitting spreads, the kurtosis value drops, until it may be close to 3.0, the value of a good bearing. This does not however rule out the possibility of a badly pitted bearing having a high kurtosis value.

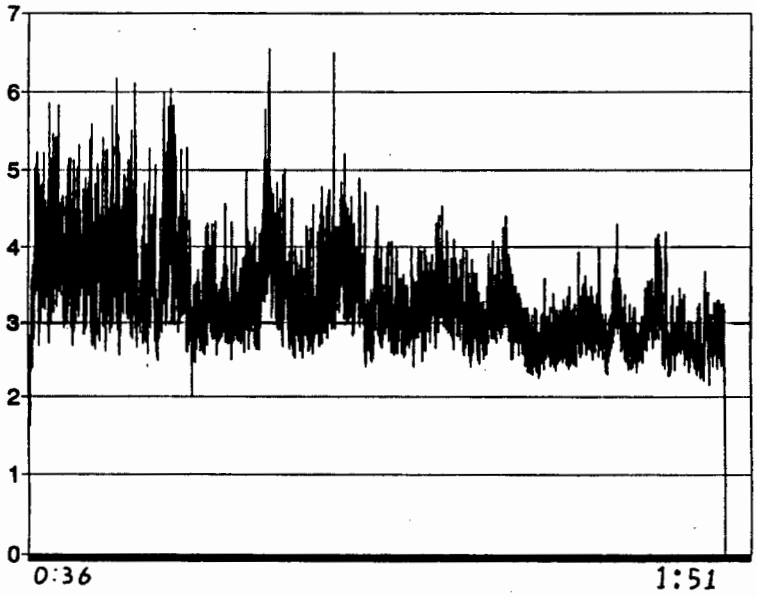


Figure 3.72: Kurtosis of Fatigue Test 2

3.19 Variation of RMS vibration with Increasing Damage

Little attention has been paid to the use of RMS vibration levels in this thesis, largely due to its lack of sensitivity as a detector of early damage in bearings, and also due to the large variations in RMS vibration with load and speed. in addition, there is variation of RMS vibration amongst machines of different types and amongst similar machines, even if operating under the same conditions. This problem is normally overcome using some sort of trending or baseline measurement for each particular machine. One useful aspect of RMS vibration levels is that they continue to increase as the level of damage increases, and in the case of FDB 1 and 2, the increase in RMS level from the start of the test to the end was a factor of 50. If tables of normal vibrations for a machine of that type are consulted, a change of this magnitude in the RMS level will be noticeable, despite the variations in RMS level that occur from machine to machine. While this sort of increase may not occur in all tests,

and the increase may only be apparent for severely damaged bearings, it can serve as a useful method of separating the severely damaged bearings from undamaged bearings, which is not always possible on the basis of kurtosis or crest factor measurements.

3.20 Discussion of the Tests on Spark Damaged Bearings

The data from these tests was an ideal test case, as there was a good signal to noise ratio and a single point defect. The kurtosis (Table 3.1) was above 4.4 for all the cases except SDB 8 and SDB 2. The vibrations in these cases were similar to those of good bearings, and are discussed at the end of this section. The kurtosis showed no obvious variation with load, but considerable variation of kurtosis with shaft speed and spectral shape was observed. This is due to the impulses becoming broader in relation to their spacing and is discussed further in Section 3.22. The power spectra of the test data showed a large number of BPFO harmonics, as well as a less pronounced component at BPFO. Two obvious resonances are present, one at approximately one kilohertz, the other at approximately three kilohertz. The exact cause of these resonances is uncertain, however the measured resonant frequency of the outer race was close to three kilohertz (Appendix E). Narrow band analysis of the lower frequencies shows peaks at R and 2R. There are also components at BPFO and 2 $\times$ BPFO. These are approximately 30 dB below the components at 1 kHz. Sidebands at the rotational and twice the rotational frequency appear around the harmonics of BPFO, at approximately 20 dB below the level of the harmonics. The BPFO components appeared in the envelope spectra and the Y^4 spectra. Sidebands at the rotational frequency appeared in some of the envelope spectra and in most of the Y^4 spectra. The cepstrum, like the other techniques, shows a clear quefrency component corresponding to BPFO.

In the SDB 2 and the SDB 8 tests, the defect is outside the load zone, and the vibrations are similar to those of undamaged bearings. The kurtosis values are low, at 2.9 and 3.0. The spectrum is nearly flat, with a component at BPFO but none of its lower harmonics (2nd 3rd etc). There are however some peaks at around 1kHz which are spaced BPFO apart. The envelope and Y^4 spectra did not give any indications of damage. The cepstrum did show a small but clear component at 10 ms, indication the presence of BPFO harmonics in the spectrum.

To summarise the observations, the outer race damage is indicated in the power spectrum by the presence of the harmonics (and not necessarily of the base frequency) of the ball pass frequency. The cepstrum, envelope spectrum and Y^4 also show the presence of this type of damage by a component at the BPFO base frequency (or quefrency), with the cepstrum being one of the more sensitive indicators. Kurtosis values higher than 3.5 indicated damage, but the actual value showed a large variation with shaft speed and the presence or absence of resonances.

3.21 Discussion of the Tests on Fatigue Damaged Bearings

The data from these three tests was analysed in the same way as the previous data. While setting up the test rig for the first test BDB 1, the bearing was brinelled. This caused a series of

small indentations around the lower half of the outer race. The kurtosis value of 3.4 indicates this damage, although it is still within the range possible from an undamaged bearing. The other indicators of damage appear in the power spectrum, with spectral components at BPFO and its harmonics. An interesting point to note is that there are also spectral components *at half the* ball pass frequency. This may be a result of interaction of signals from the rather unusual defect. The presence of BPFO harmonics is confirmed by a corresponding component in the cepstrum. The autocorrelation, included for comparison purposes, shows the same component, with more ripple either side of it. There are no obvious frequency components in the envelope or Y^4 spectra.

The FDB 1u time data showed nothing unusual, with a kurtosis value of 3.0. The spectrum did show some discrete components, at 200 and 400 Hz. Higher up, at around 2.5 kHz, were harmonics of 164 Hz. These are believed to be caused by sources of interference. The envelope and fourth power spectra showed no obvious components of BPFO or any other frequency, the cepstrum had one component at 6.06 ms, the equivalent of the 164 Hz harmonics in the power spectrum. The overall vibration level was low.

The FDB 1m data showed some time domain periodicity but low kurtosis of 2.6, with 189 Hz and its harmonics visible in the power spectrum, and also 189 Hz harmonics in the fourth power and envelope spectra. Harmonics of the rotational frequency were also visible. The fourth power spectrum showed some sidebands at 72 Hz and 16 Hz (The FTF Frequency). A component was visible in the cepstrum at 5.3 ms (1/189). The strong seventh harmonic component may be the cause of the low kurtosis value (below 3.0) The signal becomes almost sinusoidal, and as a result the kurtosis value becomes closer to the 1.5 of a sine wave.

The FDB 1a data had obvious time domain impulses and high kurtosis of 7.6 with the first four corresponding harmonics of 193 Hz. 36 and 72 Hz components were also visible. The envelope and fourth power spectrum had harmonics of 190 Hz, with some 16 Hz (FTF) sidebands in the fourth power spectrum. The cepstrum showed rahmonics at 5.3 ms and 10.6 ms as well as smaller components at 8.3 ms and 12 ms.

The results of fatigue damaged bearing two were very similar to those of fatigue damaged bearing one. Test FDB 2u, like FDB 1u had power spectral components at 164 Hz, which was lower than the BPFO. Some reduction in these was obtained by adaptive filtering, which suggests that they were caused by interference. The FDB 2m and 2a test were almost identical to the 1m and 1a tests.

The overall conclusions from these results is that the damage is indicated in the power spectrum by harmonics of the ball pass frequency. The non-BPFO frequencies in the power spectra of the undamaged bearings shows that care must be taken in analysing the power spectra, in order not to be mislead by sources of interference. The envelope and Y^4 spectra indicate the damage with components at the BPFO, sometimes these also display sidebands. The cepstrum indication is a clear quefrency component corresponding to the ball passage

frequency, without any obvious signs of other components. The kurtosis values were not reliable indicators of damage, with damaged bearings often producing near normal kurtosis.

3.22 The Observed Dependence of Kurtosis on Bandwidth.

In the tests run by Etienne Kruger, some of the vibrations showed more pronounced ringing than others, and a corresponding dominant narrow band resonance. The kurtosis for these cases is lower than the others. The conclusion was made that the impulses ring due to the narrow band resonance and are thus of longer duration, resulting in a lower kurtosis value. The two highest kurtosis values measured (6.9 and 10.0) are for the two data sets where the shaft speed was 600 RPM and not 1000 RPM. The impulses still ring in much the same way as the other cases, but are spaced further apart, in relation to their width, hence the high kurtosis values. In cases where the pulse rate is close to the time constant of the filter, the signal might become nearly sinusoidal. The kurtosis would be very low and damage would not be obvious from kurtosis measurement. In some cases the data could be inverse filtered or whitened to make an improvement. The highest kurtosis recorded (10.0) is from test C8-0. The accelerometer output was high pass filtered with a cut off of three kilohertz. This reduced the dominance of the one kilohertz ringing, resulting in a higher kurtosis.

In the case of a signal swamped by white noise, the resonance might not be visible in the spectrum. In cases like this we might use:

- A high frequency range, or multiple bands, as in Dyers original paper, to get more information from the kurtosis measurement.
- The frequency domain analysis techniques, as there might still be visible peaks in the spectrum.
- A different accelerometer mounting position, as this may provide a cleaner, less modified signal.

3.23 Conclusions

The test results all showed the same general trends. The power spectrum was a good indicator of damage, and in all the cases of damage showed spectral components at BPFO and harmonics much higher in the spectrum. In the two cases where the bearings were undamaged, there were discrete power spectral components, but these were not at a spacing corresponding to BPFO, showing that when applying these techniques, care must be taken that the observed frequencies do actually correspond to the defect frequencies. The envelope and Y^4 spectra were also effective indicators of damage, with components at BPFO. The fact that the components are at the base frequency, and not higher up in the spectra, makes them

easier to analyse. It should be noted that in the case of undamaged bearings, and bearings with defects outside the load zone, these two types of spectra showed no indication of damage, perhaps a sign that they are less likely to give false alarms. The cepstra tended to display the same information as the spectrum, namely a component corresponding to the spacing of *the harm*onics in the power spectrum. This means that when using the cepstrum, one should work with the same care necessary when using the power spectrum. The cepstrum is however easier to use than the power spectrum, as the defect signals can be observed easily at the base quefrency. Kurtosis proved to be a simple to use indicator of damage, but will tend to miss indications of damage, particularly in situations of prominent narrow band resonances, or of harmonic interference.

3.24 The Use of Multiple Parameters in Estimating the Condition of a Bearing

It has been seen so far that if only one statistic is examined, there are cases where a damaged bearing appears to be undamaged, or an undamaged bearing may appear to show signs of damage. We therefore consider the application of two statistics to the problem of classifying the condition of a bearing. A method commonly used in pattern classification is to plot two statistics or features of the test cases against one another. The data appear as points on the graph. The resulting scatter plots can then be examined for any clusters of like points. Various techniques exist for grouping the points [6], and for determining in which group a particular test case belongs. In this case, the graphs will be examined visually. The plots shown are:

1. Crest Factor vs Kurtosis
2. Cepstral Peak level vs Crest Factor
3. Cepstral Peak level vs Kurtosis

The plot of Crest factor vs Kurtosis (Figure 3.73) has most of the points on a diagonal line, indicating that these two statistics are sensitive to the same factors. Several clusters of points are visible. The undamaged bearings, and those with defects outside the load zone, shown as $\diamond$, form the cluster at the lower left corner of the graph. These are all within the region bounded by kurtosis of 3.5 and crest factor of 4.0. The tests on the bearings with spark induced defects (+) form a separate group on a diagonal line. Brinelled bearing 1 ($\times$), has a borderline kurtosis vale of 3.4, it is however clearly visible on the graph as a damaged bearing, due to the high crest factor value. There were however two points that lie very close to the cluster of undamaged bearings. The two points representing Fatigue Damaged bearing 1m($\triangle$) and 2a ($\star$) have low kurtosis and crest factor values and are thus within the region of the undamaged bearings. Possible reasons for the low crest factor and kurtosis values have previously been suggested, including dominant pure tones and insufficient signal bandwidth. We would nevertheless wish to be able to identify these cases as damaged bearings. For this reason, we will consider the use of parameters representing spectral information.

[6] The K-Means algorithm is one such technique. Using an initial set of cluster centres, the points are assigned to a cluster using a least squares algorithm. Once all points have been assigned to a cluster, improved cluster centres can be computed. This algorithm can be iterated to find better centres, but is not guaranteed to converge

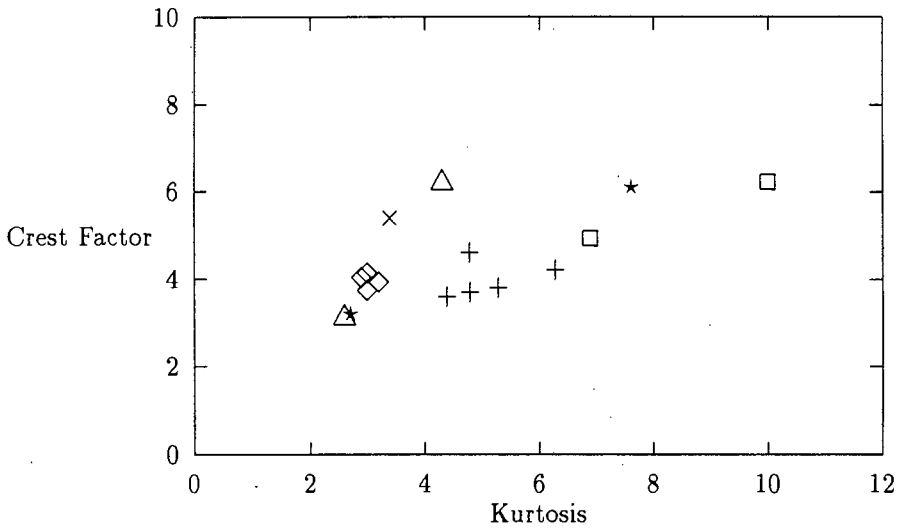


Figure 3.73: Crest Factor vs Kurtosis

Key:		
SDB 8,2 FDB 1u,2u	◇	Undamaged or Defects outside load zone
SDB 4,9,3,1,7	+	Spark Damaged
SDB 5,6	□	Spark Damaged, Low speed
BDB 1	×	Brinelled
FDB 1m,2m	△	Fatigue Damaged, Moderate
FDB 1a,2a	⋆	Fatigue Damage, Advanced

The level of the cepstral peak was chosen as a measure that compresses a large amount of spectral information into a single parameter. When the graph of Cepstral peak versus Kurtosis (Figure 3.74) is considered, the undamaged bearings ($\diamond$) and the spark damaged bearings ($+$) again form two distinct clusters. In addition, by grouping all bearings with kurtosis higher than 3.5 and cepstral peaks greater than 2dB as damaged, all of the damaged bearings can be distinguished from the undamaged bearings, despite the fact that fatigue damaged bearings do not form any obvious clusters. In this regard, there are two examples of moderately damaged bearings (FDB 1m and 2m) ($\triangle$), and two examples of bearings with advanced damage (FDB 1a and 2a) ($\star$) yet it is FDB 1m and FDB 2a that exhibit similar behaviour (high kurtosis, high cepstral peak) while FDB 1a and FDB 2m have low kurtosis. With more examples, a general trend might be observed, but in the case of only two examples, with both cases exhibiting different behaviour any attempt at generalisation is misleading. In order to correctly classify any type of bearing damage, a large number of test cases of each type of damage would be required. In addition research into which combinations of parameters yield the most useful information would be indicated.

This analysis has shown that consideration of more than one statistic can help in determining whether or not a bearing is damaged. At present there is too little data available to indicate which statistics should be considered, and how these should be interpreted. To proceed further from this point, research would be needed to determine:

- Which statistics to use.
- Which combinations of statistics we want.
- How we make allowances for variations in conditions, such as different speeds, different loads and varying bearing types.

The suggested research methodology is as follows:

Using bearings at a known stage of damage, run them under a variety of speeds and loads, recording the vibrations and vibration parameters. This was done to some extent by Brad Leggat but is of limited applicability as only RMS and Kurtosis were monitored. From this data, work to establish:

- The relationship of the statistic to load, speed, bearing condition.
- The variance of the statistic when these parameters are kept constant.
- The degree of covariance between statistics and how thus which statistics to use to reduce the amount of redundant information.

This information can then be used to develop a general condition monitoring system.

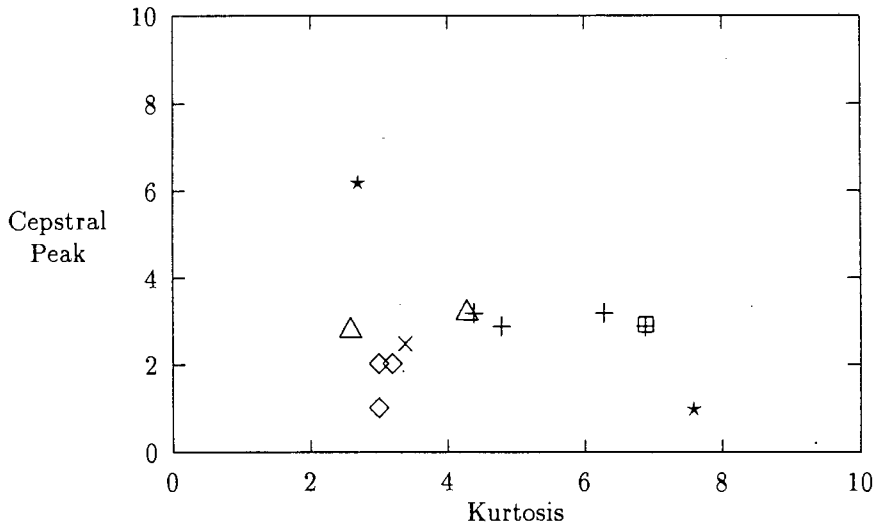


Figure 3.74: Cepstral Peak vs Kurtosis

Key:

SDB 2, FDB 1u,2u	◇	Undamaged
SDB 1,3,4,5	+	Spark Damaged
SDB 5	□	Spark damaged, Low Speed
BDB 1	×	Brinelled
FDB 1m,2m	△	Fatigue Damage, Moderate
FDB 1a,2a	⋆	Fatigue Damage, Advanced

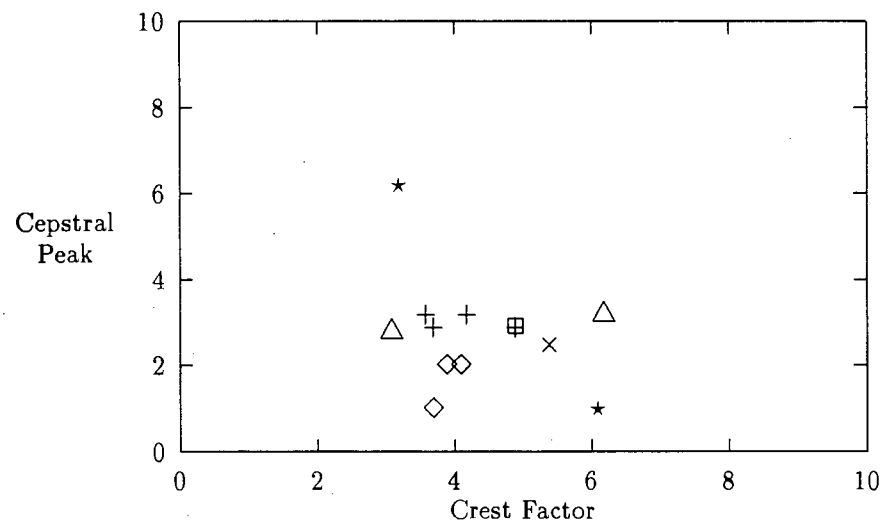


Figure 3.75: Cepstral Peak vs Crest Factor

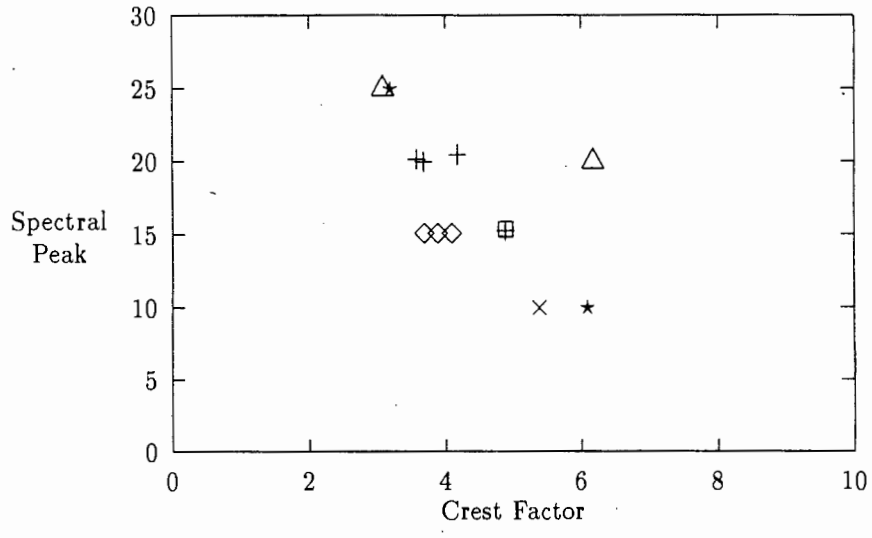


Figure 3.76: Cluster plot spectrum peak vs crest factor

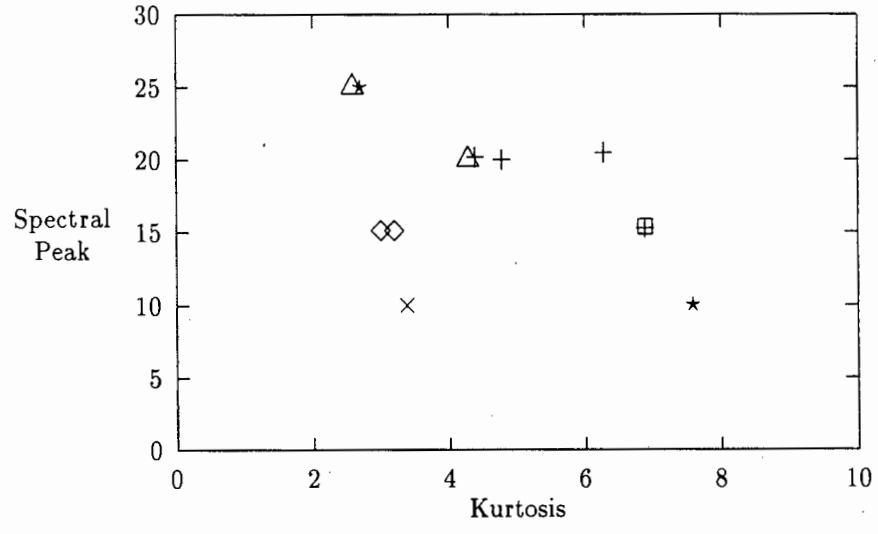


Figure 3.77: Cluster plot spectrum peak versus kurtosis

Key:

SDB 2, FDB 1u,2u	◇	Undamaged
SDB 1,3,4,5	+	Spark Damaged
SDB 5	□	Spark Damaged, Low Speed
BDB 1	×	Brinelled
FDB 1m,2m	△	Fatigue Damage, Moderate
FDB 1a,2a	⋆	Fatigue Damage, Advanced

3.25 Making use of Condition Estimates

There are three things we may wish to determine from condition monitoring:

- Does some form of bearing damage exist?
- What stage of damage has been reached?
- What life does the bearing have remaining.

Previous sections have been concerned with determining whether damage exists, and what stage of damage has been reached. The third item listed will be considered, that of determining the remaining life of the bearing. To attempt this several factors should be taken into consideration. The first is what should be considered the end of the usable life of the bearing. One criterion would be the first detectable signs of damage. Another may be the point at which catastrophic failure of the bearing is imminent. The actual criterion used would depend on considerations such as the safety margins required, and the likelihood of the damaged bearing affecting other machinery. In each case, a level of damage must be chosen that represents the point at which that particular machine must be shut down. Then we wish to estimate time that will take for this stage to be reached. Ideally, this would be in hours running time, but initial estimates would be in terms of % of L_{10} bearing life. From the general life span of bearings on that machine, or estimates based on the operating conditions, the number of hours running time remaining can then be estimated.

3.26 General Procedure of Analysis

The suggested process follows finding the three general types of information:

1. Determine whether or not there is a reason to suspect damage.
2. Verify that damage exists by using signal restoration techniques, and further analysis. Then determine extent and type of damage present.
3. From the type and extent of damage present, as well as knowledge of the machine type, estimate the remaining life.

3.26.1 Determining Whether some Form of Damage Exists

To obtain an initial assessment a variety of basic parameters would be measured:

- Temperature [7]

[7] This is not a particularly sensitive measure of damage, but this and other non-vibration measurements should also be considered

- Kurtosis
- Crest Factor
- RMS Acceleration and Velocity
- Acceleration Power spectrum
- Acceleration Power Cepstrum

If all of these parameters are within the range suggested for a normal bearing, then the bearing may be regarded as being undamaged. If any of these parameters indicates the bearing is damaged, then further analysis is indicated. In addition, if it is possible that evidence of damage is being masked by unusual resonances or interference, then again further analysis is indicated. (It has been seen that if there is some effect masking the damage, it may be apparent when other signal processing techniques are considered. For example the ringing that produced low kurtosis values in SDB 5 was apparent in the power spectrum) This initial analysis is intended to be something that can be performed quickly at the measurement site, and catch all possible types of damage, thus allowing further measurement if necessary.

3.26.2 Pinpointing the Type and Extent of Damage

If the signal has any interferences or resonances or for some other reason is not adequate to accurately determine the type and extent of damage, then signal restoration techniques should be applied. Filtering and adaptive filtering can be used to remove interference, while inverse filtering can help suppress resonance. Other appropriate techniques such as those discussed in Chapter 2, Section 2.5 can be applied. Once the improved signal has been obtained, the vibrations parameters can once more be computed. If the indicators of damage are still present, or have become apparent, then the analysis can proceed to determining the type of damage (By this we mean rolling element damage, inner/outer race damage, cage damage or some other type of damage). The various signatures produced are discussed in Chapter 1, Section 1.8. These signatures are usually found using the spectral analysis techniques discussed in Chapter 2, Section 2.3. Once the type of damage has been determined, the extent of the damage can then be estimated, by examining which combinations of damage symptoms are present. Once the extent of the damage has been estimated, then an attempt may be made at estimating the remaining life of the bearing. Methods of doing so have not been examined in any detail in this thesis, but some ideas are presented in the next section.

3.27 Estimating the Remaining Life of a bearing

The estimate of of the remaining bearing life will be a statistical one based on a large number (hundreds) of test cases. For example, if we tested 100 bearings to destruction, we might

find that the time it takes for 10 % of the bearings to reach the initial stages of damage is $1.0 \times L_{10}$ as per the definition of L_{10} life. Then, from the time initial damage occurs to the state of advanced damage is $k_1 \times L_{10}$, where k_1 is perhaps 0.1 again this is for 10 % of the bearings. The time for 10 % of bearings to reach advanced damage from moderate damage *would* be $k_2 \times L_{10}$ where k_2 is probably lower than k_1, perhaps 0.05.

3.28 Summary of the Method used by C. Cempel

In the paper by C. Cempel [29], only RMS vibration was considered. This is the parameter S in the paper. What is of interest however is not the parameter used, but the method. If we consider a symptom S, which has two main value of interest to us, S_a the alarm level and S_b, the break down level. We wish to determine appropriate values for these. S is dependent on the machine running time θ, but is also dependent on a number of other parameters, ω_i. To give :

$$S = S(\theta, \omega_1, \omega_2, \omega_3, \omega_4) \tag{3.3}$$

This gives a certain random nature to measurements of $S(\theta)$. If we work with a large number of machines, we can obtain statistical estimates of $S(\theta)$ and $p(S)$. Cempel has made the assumptions that the machine life θ_b is defined by a Weibull distribution, that $p(S)$ and $p(S(\theta_b)$ are also defined by a Weibull distribution. If one then assumes that $S(\theta)$ increases monotonically with increasing θ and that the measured $p(S)$ is measured at increments of constant $\delta\theta$, one can estimate $S(\theta)$ from the parameters of the Weibull distribution. Then by using the Nayman – Pearson observer the optimal θ_a can be derived for a given rate of unnecessary repairs and minimum breakdowns.

Chapter 4

Condition Classification and Artificial Intelligence

It has been stated that part of the objective is to make use of artificial intelligence techniques to analyse and classify the condition of the machine component. This may be as an aid to the user of the system, or as part of an automated condition monitoring system. This chapter provides a review of two basic artificial intelligence techniques and illustrate how they can be used in condition monitoring.

4.1 Feature Vectors

The result of the signal processing is a series of attributes of the vibration and possibly any other features concerning the operation of the machine. This list of features forms what is often known as a feature vector. On the basis of this vector, we wish to classify the condition of the bearing in some way. This may be a classification into two classes, damaged and undamaged, or classification according to the type of damage. The feature vector defines a point in an N dimensional space. Thus for a N = 1 the vector defines a point on a line. For N = 2 a point on a plane, for N = 3 a point in space. For N > 3 these ideas cannot be depicted using geometry, but the same techniques apply. For initial explanations of these ideas, we will assume that we wish to classify the cases into two distinct sets, good and bad. In the case of a single parameter or one dimensional feature vector, this is done by defining a set of points that form the boundaries of the range of that feature for good bearings. If the feature used was kurtosis, the boundary points might be selected as 2.7 and 3.4 . Any bearings with kurtosis outside this range would be classified as damaged. For N=2 the boundaries would be lines and curves, for N=3 a planes and surfaces .... The benefit of a higher order feature vector was demonstrated in Chapter 3. The Brinell damaged bearing had a kurtosis of ...if classification were performed on this basis alone, the bearing would be classified as undamaged. If an additional parameter is used, such as crest factor then the b4bearing is

easily correctly classified as damaged.

The information represented by the feature vector should be carefully chosen in order to make optimal use of the classification algorithms available. The data should provide as much useful information as possible. The information should be reliable and there should be a minimum of redundant or irrelevant information. The signal processing techniques of Chapter 2 provide the basic data for inclusion into the feature vector. The time domain statistics are obvious candidates, as the information is in a concise form. Use of RMS vibration and one or two other factors (kurtosis, crest factor) would represent the bulk of the information available from these statistics. The experimental data has shown that these alone do not provide sufficient information for all cases. The use of frequency domain information is necessary to provide a complete picture of the machine condition. The frequency domain techniques provide a large quantity of information, The power spectrum generally provides 400 data points. Inclusion of this data into the feature vector would provide the necessary information, However it has been seen that analysis of the power spectrum may be complicated, which further complicates the analysis procedure. Alternatively, the data can be reduced by making use of one of the other frequency domain techniques. The cepstrum, metacepstrum and the envelope spectrum have all been shown to simplify the data present in the power spectrum. The position and amplitude of the largest peaks can be included into the feature vector. Alternatively, the amplitude at the defect frequencies can be included. A simpler technique that can be used is the spectral alarm level technique. The information from this technique is already reduced to a few values, and easily evaluated. The complexity of the system should vary according to the importance of an accurate evaluation, and the time available for it.

4.2 Artificial Intelligence and Machine Condition Determination

Two techniques for classifying machine condition from using artificial intelligence will be discussed. The first type is a heuristic or rule based classification system. The most commonly type of heuristic classifier used in condition monitoring is an expert system. The second type is a trained classification system, which in this case is a neural network. Expert systems have been in use in condition monitoring for some time, while neural networks are only beginning to make an appearance.

4.2.1 Expert Systems

Expert systems operate using a set of rules of an **if this is true... then that is true...** nature. Some of the well known rules of thumb can then be programmed into the computer and applied automatically, for instance a simple set of rules might be:
If the Kurtosis > 3.5 and vibration at 120 Hz > 10mm/s

then DAMAGED = TRUE

True expert systems differ from simple decisions making programs in a number of ways:

- Expert Systems are designed for change. The rules, or rule base is not hard coded into the program, but can be altered without altering or recompiling the program.
- The process of making conclusions using the rules is handled by the shell, and does not have to be handled by the user or incorporated into the rules.
- An expert system can be interrogated to allow the user to follow the decision process that led to the output. In some systems the user may force the system to make a particular decision, and then follow the reasoning from then onwards.

Expert systems are of little use by themselves, as whatever conclusion the expert system comes to, a decision must be made by the user as to what action should be taken and when. They do however reduce the amount of data to be analysed and allow relatively unskilled personnel to perform the condition assessment. The rules used by an expert system can perform any classification and decision making process that a person can perform, provided at set of rules form performing the decision making process can be produced. It is the creation of this set of rules that forms the major task in producing an expert system.

An expert system can utilise the data and rules using two distinct decision making techniques, forward chaining, and backward chaining. In forward chaining, the expert system starts with the available data and uses the rules given to make a conclusion. In backward chaining, the process would start with an assumption, and the expert system would attempt to prove that assumption with the available rules and data.

An expert system for condition monitoring was the topic of a thesis in the Mechanical Engineering Department. The author, Brad Leggat [27] found that kurtosis or RMS measurements alone were not sufficient for determining the condition of a bearing. For this reason, the three parameters Temperature, RMS acceleration and kurtosis were used as the basis of the decision making. While this is better than a single parameter, it may not always be sufficient data to make the correct classification. Any future system should make use of frequency domain information, to produce a more reliable classification.

4.2.2 Neural Networks

Neural networks are becoming very popular and deserve some consideration in any situation where data classification and decision making takes place. There are many types of neural network, (back propagation, self organising maps etc) only one of the simplest, the back propagation network will be considered here. The back-propagation network consists of several layers of interconnected nodes. Each node is constructed as follows: There are several inputs, usually a number in the range -1 to 1, representing some normalised datum. Each of the

inputs is multiplied by a weighting factor and they are then added together. This sum is then passed through a non-linear limiting function, often a sigmoid or an arc-tan function. This output is then connected to the inputs of the nodes in the next layer as in Fig 4.1.

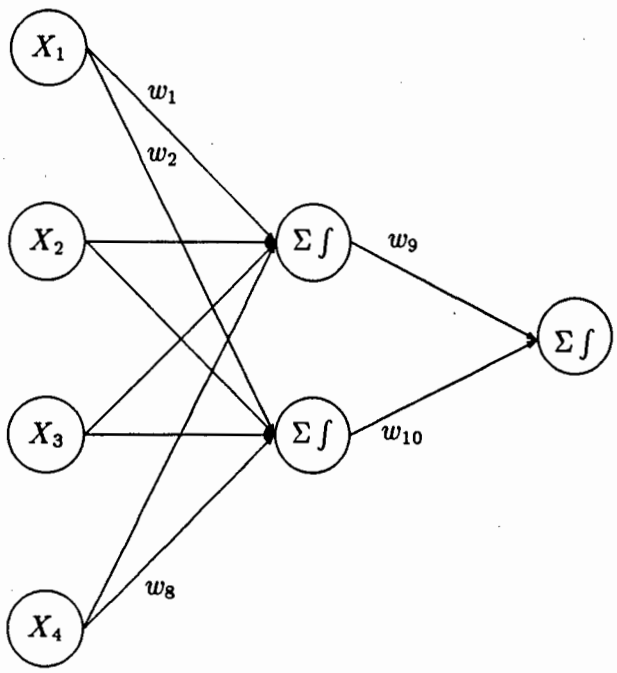


Figure 4.1: Diagram of a Neural Network

To be of any use, the network must be trained. This is accomplished by applying a set of data to the inputs of the network, and then adjusting the weights of the nodes to make the output closer to the desired value for that input. This is repeated several times for the whole set of training data until the output of the network is within an acceptable range of the desired output. The network is then tested on data that has not been presented to it before. If it performs satisfactorily on this data, then it is ready for use. If not then it must be retrained using a larger training set, with data more representative of the different cases the network must learn.

Neural Networks can be used in two ways, either as classification tools, or as computation tools. When used as classification tools, the input to the network would be some feature vector made up of a variety of parameters concerning the machine running conditions and vibrations. For example the inputs could be shaft speed, temperature of the bearing housing,

RMS vibration amplitude, kurtosis etc. The output/s could represent good, bad or indeterminate bearing condition. The network would be trained to give the desired output for the input parameters, and would then be tested to see if it will generalise to other similar cases. If used as a computational tool, the input data would consist of partially processed or unprocessed vibration data, and the output would be some sort of statistic representing the bearing condition. It has been proven (Kolmogorov) found in [30] that a three layer network with N(N+1) nodes can map any continuous function of N variables. However this does not give an indication of how the weights and limiting functions in the network can be found. In general Neural networks have been found to be better suited to problems of classification [31]. Neural networks have been successfully used in a variety of applications, but do have some important drawbacks.

- A neural network provides no information on how the answer was arrived at, merely a set of numbers representing the answer.
- The behaviour of a neural network may be unpredictable for unusual data. Part of the power of a neural network is that from the training data it will generalise to cope with similar but previously unseen cases. However, if the data presented to the network is unlike any in the training set, then the output may not be reliable and therefore it is risky to base decisions on the output of the neural network.
- When used in computation, neural networks are not usually efficient, and algorithms that can be implemented using very simple order N processes may be converted to algorithms using order $l_0 \times l_1 + l_1 \times l_2 + ...l_{n-1} \times l_n$ where l_k is the number of nodes in the Kth layer and l_0 is equal to N.

This should be weighed against the major advantages of a neural network:

- Training a neural network to perform a classification process does not require any knowledge of how the classification can be performed, merely a large set of examples of data correctly classified. This may be obtained in practice by measuring the appropriate parameters and physically examining the bearing to determine the level of damage.
- Once the neural network is trained, it may be able to generalise from the training examples to similar cases which it has not as yet seen.

To date several projects using neural networks for pattern recognition and classification have been undertaken within the Electrical Engineering Department. There have not however been any such projects using neural networks in condition monitoring.

4.3 Conclusions

From the preceding sections the following conclusions can be made:

1. Expert systems can perform classification using a set of rules.
2. The major task in producing an expert system is in formulating the rule base.
3. An expert system using rules based on bearing temperature, RMS vibration and kurtosis was part of a previous thesis. Frequency domain analysis was not used in this expert system.
4. Neural networks can perform classification based on training examples.
5. The use of neural networks in condition monitoring has not yet been studied at UCT.

4.4 Recommendations

1. It is recommended that further work on expert systems makes use of frequency domain data as well as time domain information for classification.
2. The suitability of neural networks for condition determination should be studied.

Chapter 5

DSP Tools for Real Time Condition monitoring

The previous sections have concentrated on signal processing techniques, their strong points and weak points. This section is concerned with a real time signal processing system, and how these techniques can be incorporated into it.

5.1 An Ideal Analysis System

Earlier in this thesis the overall objective was stated, namely the ability to determine machine condition on site, using real time tools. The system used to perform this real time analysis would incorporate many of the features of a spectrum analyser, notably power spectrum and cepstrum. In addition to this other tools would be needed, as some of the signal processing techniques discussed are not normally available on spectrum analysers. Kurtosis, crest factor and meta cepstrum would all be useful additions. Combined with this we would like to make use of other computer aided analysis tools:

1. Data bases of bearing dimensions, machine parameters and suggested vibration limits for various type of machines.
2. Analysis aids, such as programs for calculations of bearing characteristic frequencies, gear mesh frequencies, fan blade pass frequencies or other frequencies we would expect to observe on a machine.
3. Artificial intelligence programs to automate or guide the condition monitoring process.

5.1.1 The Importance of Real Time Analysis

The term real time means that data is sampled, processed and displayed as the event being analysed is occurring. The question arises as to why real time analysis is necessary. The other

options are:

1. Record the data using a DAT recorder or a vibration snapshot recorder. This data can then be analysed using a Personal Computer with a signal processing package, or using condition monitoring software.

2. Use a portable computer equipped with an analogue to digital converter card. A section of data can be captured, then analysed and displayed.

While many companies do perform their condition monitoring by collecting data using a recorder and analyse it later on a computer, occasionally a difficult to diagnose problem arises. An initial measurement and partial analysis is necessary as a guide to making further measurements. Once some initial clues to the type of fault are available, the appropriate measurement type and transducer positions can be chosen. The second question then arises, why is dedicated DSP processing necessary? The signal processing could be performed by the host computer. The reason is that the display of data should be fast and responsive. Then one can observe the machine vibrations vary as changes are made to the system, for instance as the motor speed changes or the load varies. The problem areas can then be quickly isolated, and further analysis performed.

5.1.2 System Architecture

The question of what type of system architecture to use is still unanswered. As initially planned, the system was to be built around a portable IBM compatible PC with a DSP card. However, this may not be the best choice. Options that should be considered in more detail include:

1. A spectrum analyser. Spectrum analysers are available that perform most of the signal processing tasks that we require, and can be programmed to perform others. In addition, the basic signal processing, display and user interface routines are provided. This, combined with the excellent analogue front end amplifiers supplied in spectrum analysers make this an attractive proposition. However, the system lacks one of the aspects that one would want, the inclusion of applications such as data bases and expert systems on the same platform.

2. A spectrum analyser, combined with a PC could overcome the drawbacks mentioned. There would however be a problem that the system is now bulkier and less portable, and puts the cost of a personal computer on top of the high costs of a spectrum analyser.

3. A computer of some other type, with a DSP card. There may be some other type of computer better suited to the application, however IBM compatibles are a virtual standard in South Africa, and are easily and cheaply available.

While these are all viable options, a PC with a DSP card has a large number of advantages:

- A PC is a cheap option. Fast portable computers are available for under R 20 000.
- IBM compatibles are a virtual standard in South Africa and are therefore easily available.
- Use of a plug in DSP card makes the design modular. One part of the system can be upgraded without affecting other parts.
- A PC provides a familiar environment for programming and developing software.

For these reasons, it was decided that use of an IBM PC and DSP card was the option to use.

5.1.3 Details of System Modules

The system can be considered to consist of a series of modules. The modules must be integrated into a system, but for our current purposes we will consider them separately. The basic modules for the system would be:

- Analogue Front End
- Anti-alias filter and Analogue to Digital converter.
- Signal processing system
- System processor and hardware
- System software

The analogue input section conditions the signal so that it is at the optimum level for conversion by the ADC. The input must also provide overload protection for the sensitive ADC circuitry. The input amplifiers should have variable gain, to allow inputs from millivolts to tens of volts, with bandwidth sufficient to cover the frequencies of interest, up to 50 or even 100 kHz. If possible, the gain of the input section should be digitally controlled. This allows auto ranging and prevents overloaded input data giving incorrect results. Careful consideration should be given to the input section and a decision made either to build or buy the appropriate equipment.

Anti-alias filtering and analogue to digital conversion. Once the signal has been amplified to the right level, it is anti-alias filtered and then sampled, usually at 12 or 16 bits resolution. The sampling rate of the analogue to digital converter will depend on the frequency range of interest. The required sampling rates may vary from 100 Hz to 100 kHz. For each sampling rate, the appropriate anti-aliasing filter must be used. There are several

ways that the variable rate sampling can be accomplished. The first option is to use a tunable low pass filter, or several low pass filters, and vary the sampling rate of the ADC. Filters that are sufficiently flat in the pass band, and then drop sharply enough in the transition band are costly to construct and so the use of multiple filters is problematic. Alternatively the sampling could be performed at a constant rate.The data could then be digitally filtered and decimated to produce samples at the required sampling rate. This reduces the filtering requirements, as only one anti aliasing filter is needed, but needs either extra signal processing software, or dedicated decimation chips. The compromise option is to use a ADC chip like the DSP56ADC16, which requires a very simple anti aliasing filter. The input filter for a device like this is a simple RC filter. Several of these could be constructed to cover all the signal ranges necessary, or a tunable filter could be constructed using a single resistor, and adding different capacitors using switches.

The signal processing chip, as well as the way it is implemented, is a complex choice that was considered in a previous project [32]. For this reason it will only be touched upon briefly. The signal processor must be capable of handling sixteen bit data, without loss of precision, and must be capable of performing the signal processing techniques described in Chapter 2.

System processor: With the low cost of PCs, and considering the speed with which technology becomes outdated, the fastest technology available should be considered. If the system processor is powerful, then complex algorithms can be handled by a combination of the DSP processor and the system processor. The current top of the range is a 486 based machine. This would probably be a good choice for the target machine as the processor can make use of 32 bit software, and has a built in numeric co-processor. These machines are currently very expensive, but the system could be developed using a slower machine.

System Software: The system software must interface with DSP card, by transferring data and downloading programs as well as provide user interface. A Graphical user interface, such as Microsoft Windows would provide most of the interface routines, but the display is very slow, and unresponsive even on a fast computer. Considering this, a simple shell written in C may be preferable.

5.2 Capabilities Required of the DSP Subsystem

From the previous sections we have learnt that a DSP system will need to perform the following functions:

- Time domain statistics: This is a fairly simple procedure - by calculating the mean, peak, sum of the squares and sum of the fourth powers, all of the basic time domain statistics can be calculated. For example these four values can then be used to calculate kurtosis, crest factor and standard deviation.

- Fourier transform based calculations. The DFT forms the basis of a range of calculations. With minor variations, the DFT can be used to calculate the power spectrum, envelope spectrum, cepstrum and autocorrelation. The major difference between these being the way that the data is processed before and after the DFT. The power spectrum in many analysers is a four hundred line spectrum, with an eight hundred or sixteen hundred line spectrum used for narrow band analysis. This suggests that a 4096 point DFT (needed for 1600 line spectra) is a suitable mark to aim for in Fourier transform capability.

- Adaptive filtering. This has been shown to be an effective way of reducing interference, and should be an option for use in combination with the other techniques.

- Decimation and zooming. Decimation and zooming are used to perform narrow band analysis. Decimation is a software technique for lowering the sampling rate of sampled data, thus reducing the bandwidth. Zooming is used with the power spectrum, to produce power spectra of narrow frequency bands.

These signal processing functions make up the basis of most of the analysis that will be performed. Other functions tend to be variations and adaptations of these capabilities. The sections that follow will discuss the DSP hardware and software, and conclude on the suitability of the hardware available, and what should be done to improve it.

5.3 Existing DSP Hardware

The Central Acoustics Laboratory purchased a Proto56 DSP board, which is built around a Motorola DSP 56000 CPU. During 1990, Mark Levy [32] added two DSP56ADC16 analog to digital converters and the relevant decoding and interface circuitry. This circuit was built on vero-board and attached to the Proto56 board. The author rebuilt this circuit on a printed circuit board, and placed it in a separate enclosure. A simple analog front end buffer was added and the schematics were updated. The new schematics and circuit board plots are shown in Appendix F.

5.4 Analogue to Digital Converter Circuit Description

5.4.1 Address Mapping and Decoding

The analog to digital converters are mapped into the top area of processor Y-memory. ADC1 is situated at address Y:$FFFE; ADC2 is situated at Y:$FFFF. The data from the ADC is 16 bit, twos complement data. When the ADC is read, the data is placed into the top 16 bits of the processor word, thus the data does not have to be sign extended. The End of Conversion flag is read with ADC1 and is placed in bit 0 of the processor word. A register

used for controlling the sampling rates of the ADC's is mapped to address Y:$FFFD. Bits 0 and 1 of this register connect to the FSEL pins of the ADC chips. Writing 0 to this port will select the lower sampling rate, $0002 the higher. The address decoding is performed by a circuit constructed from ordinary CMOS logic gates. There are 3 outputs, one for each of the three addresses used. This address decoding forms the bulk of the circuit. Replacing the decode circuitry with a PAL would reduce the size and complexity of the circuit considerably.

5.4.2 Analog Front End

The front end amplifiers consist of LF 351 op-amps with limiting diodes. The signal is AC-coupled and level shifted to 2.5 volts. A lowpass filter is used to cut out high frequency signals. This front end is merely intended as a basic protection circuit for the ADCs. Variable gain amplifiers will still be needed to amplify the signal to the required two volt peak to peak input level.

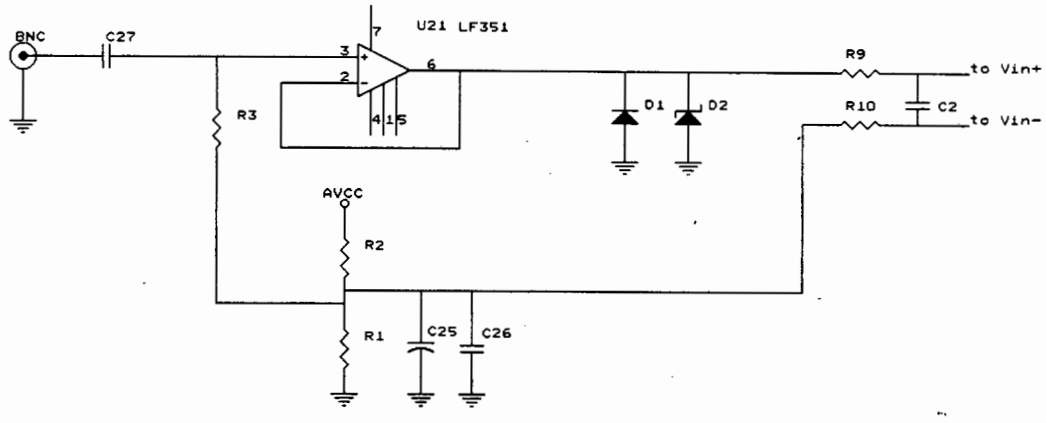


Figure 5.1: Analog Front End

5.5 The ADC16 Analog to Digital Converter

5.5.1 Operating Principle

The converter consists of two main stages. The first stage is a 12 bit Delta-Sigma analog to digital converter that can operate at frequencies up to 400 kHz. The basic sigma delta

principle is as follows. A switchable voltage reference is subtracted from the input signal. The result is integrated. The voltage reference is switched in or out each clock cycle to maintain the output of the integrator at zero volts. At the end of the measurement cycle the integrator will have been switched in for N out of the total M clock cycles. Since the output of the integrator will be zero or close to zero, the integral of the input voltage will equal the integral of the reference giving the result $Vin_{avg} = Vref \times \frac{N}{M}$, Since the signal is integrated over the entire measurement period, the converter provides rejection of all frequencies which are integer multiples of the sampling frequency, resulting in the nulls visible in the frequency response shown in fig 5.2. The second stage consists of a digital filter and 4 to 1 decimation section. This section provides 16 bit output at 1/4 the frequency of the first stage. The two internal filters mean that when the converter is operated in 16 bit mode, the only anti-alias filter needed is a simple one pole RC filter.

5.5.2 Anti Alias Filtering

The delta-sigma converter has an inherent filter, shown in Figure 5.2. This is combined with an RC filter with cut off frequency of 50 kHz to remove high frequency signals. The digital filter of the second stage compensates for the soft roll-off of the first stage filter and gives a sharp cut-off. There will then be only a small amount of aliasing, in the region of 45 to 50kHz. Further information on the analog to digital converters can be found in [32, 33, 34, 35].

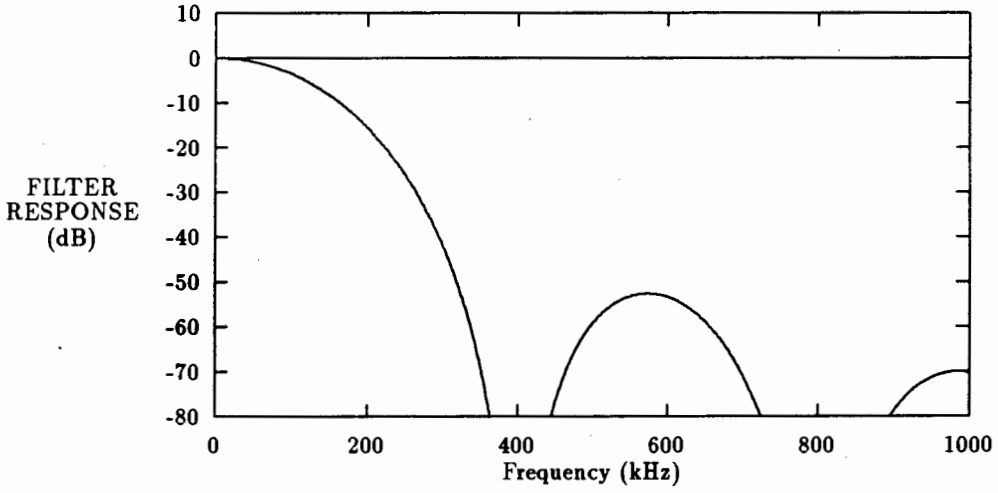


Figure 5.2: Inherent Comb Filter of ADC chip

5.6 The DSP Processor

The Motorola 56000 is a 24 bit micro processor for DSP applications. It is based on a Harvard architecture, with separate data and program memory. The data memory is also subdivided into two spaces, the x and y memory spaces. There are four general purpose 24 bit registers, x1, x0, y1 and y0 which can also be used as two 48 bit registers. The arithmetic is accumulator based, with the destination of arithmetic operations being one of the two 56 bit accumulators. The data is represented as 24 bit binary fractions. This means the data is in the range -0.99999 to 0.99998. If the data is to be considered integer data, then appropriate shifting and control of the binary point is needed.

5.7 The Proto56 DSP Board

The proto56 board has been mentioned in previous sections. From a programming point of view, it consists of an M 56000 processor with 4k words of x data memory, 4k words of y data memory and 512 words of program memory. The limited program memory allows only short programs to be used, although other boards are available that have more program memory.

5.8 Programming Tools

There are several tools available for programming the 56000, namely assemblers and C compilers. Motorola produce an assembler for the 56000 which is commonly used. There are also C compilers available:

- The C compiler produced by Motorola
- The Public Domain C compiler GCC produced by GNU and modified for the 56000 by Motorola.

The GCC compiler version 1.37 was tested, and although easy to use, the code was large and inefficient. Considering the limited memory available, short efficient code is desirable. The version of GCC for the 56000 that is available, does not allow the easy inclusion of assembler code that later versions of GCC allow, thus combining C code and assembler code is difficult.

5.9 Program Format and Conventions

Figure 5.3 shows the general format of a DSP program, consisting of data acquisition, processing, data transfer and servicing host requests. The processing section will consist of such routines as FFT, kurtosis or cepstrum. The data acquisition would in general consist of collecting samples from the ADC's but can also be transfer of data from the host. Preprocessing

of the data can also be performed as part of the capturing routines. Some common types of preprocessing would be scaling, filtering, rectification or adaptive filtering.

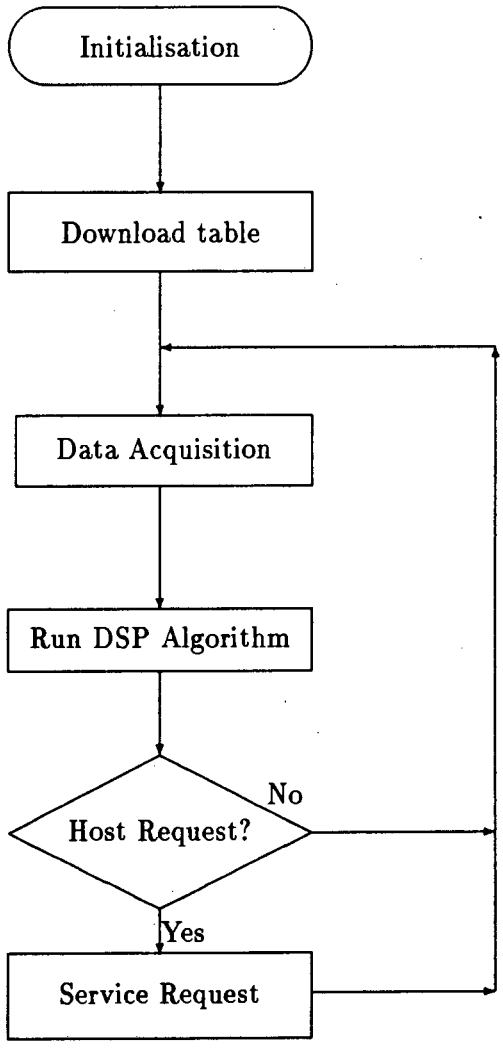


Figure 5.3: Flowchart of a DSP Program

5.10 Public Domain DSP Routines

Certain DSP routines are available in the public domain and can be downloaded from various archives via the internet. Amongst them are FFT routines, sin and cosine routines and log routines. Where these have been used, the code has been labelled as such. FTP sites for these routines and other DSP tools were:

- nic.funet.fi in pub/ham/dsp
- evans.ee.adfa.oz in pub/micros/56k

5.11 The DSP Shell

A simple shell program was written to simplify the use of DSP routines. The shell is a C program that runs on the PC. This program provides routines of several types:

- Display routines for the data are provided. These provide fairly simple ways of displaying the data in gray shades or as a graph. Page swapping can be used to produce flicker free fast moving displays.
- Menu and mouse support. A menu system is provided. Some menu functions have been written, such as Print Screen or Exit Program. Others can be added fairly simply.

The code for the routines and the details of using the shell are presented in the appendix H.

5.11.1 Running the Shell

Several of the DSP routines have been included in the program as a demonstration. To run this program first run DSP.COM, then run DSPSHELL.EXE. The program will run and is menu driven from then onwards. The disk included with this thesis combines these two programs in a batch file. This can be run by typing DEMO at the A:> prompt.

5.12 DSP Routines

Some DSP ROUTINES were written to create a sample real time system. There were four of these routines, a power spectrum routine, a cepstrum routine, a kurtosis routine and an adaptive filter. The source code for these routines can be found in Appendix G.

5.12.1 The Power Spectrum

The power spectrum routine was in most respects similar to that written by Mark Levy. The first major difference is the sine lookup table is not downloaded from the host, but generated

using a macro. The second is the logarithm lookup table has been replaced by a function using a polynomial approximation. The program operates as a series of steps:

1. The window is downloaded from the host.
2. The data is read from the ADC.
3. The data is windowed.
4. An FFT is performed.
5. The log of the magnitude is calculated.
6. The data is transmitted to the host. At the same time it is converted from bit reversed order.
7. Steps 2 – 6 are repeated indefinitely.

A plot of the output of the program for a triangular wave input is shown in Figure 5.4. A Grey scale plot of a varying frequency triangular wave is shown in Figure 5.5. The power spectrum is based on a 1024 point FFT. There is however sufficient memory to allow 4096 point FFTs. Thus up to 1600 line power spectra are possible.

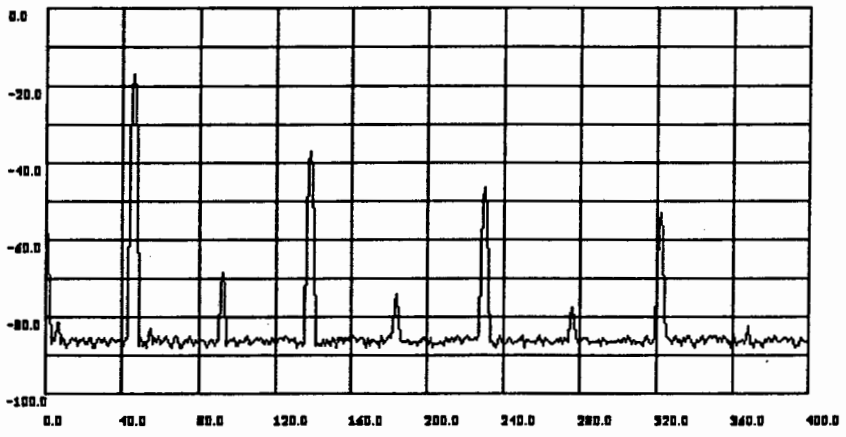


Figure 5.4: Power Spectrum of a Triangular Wave

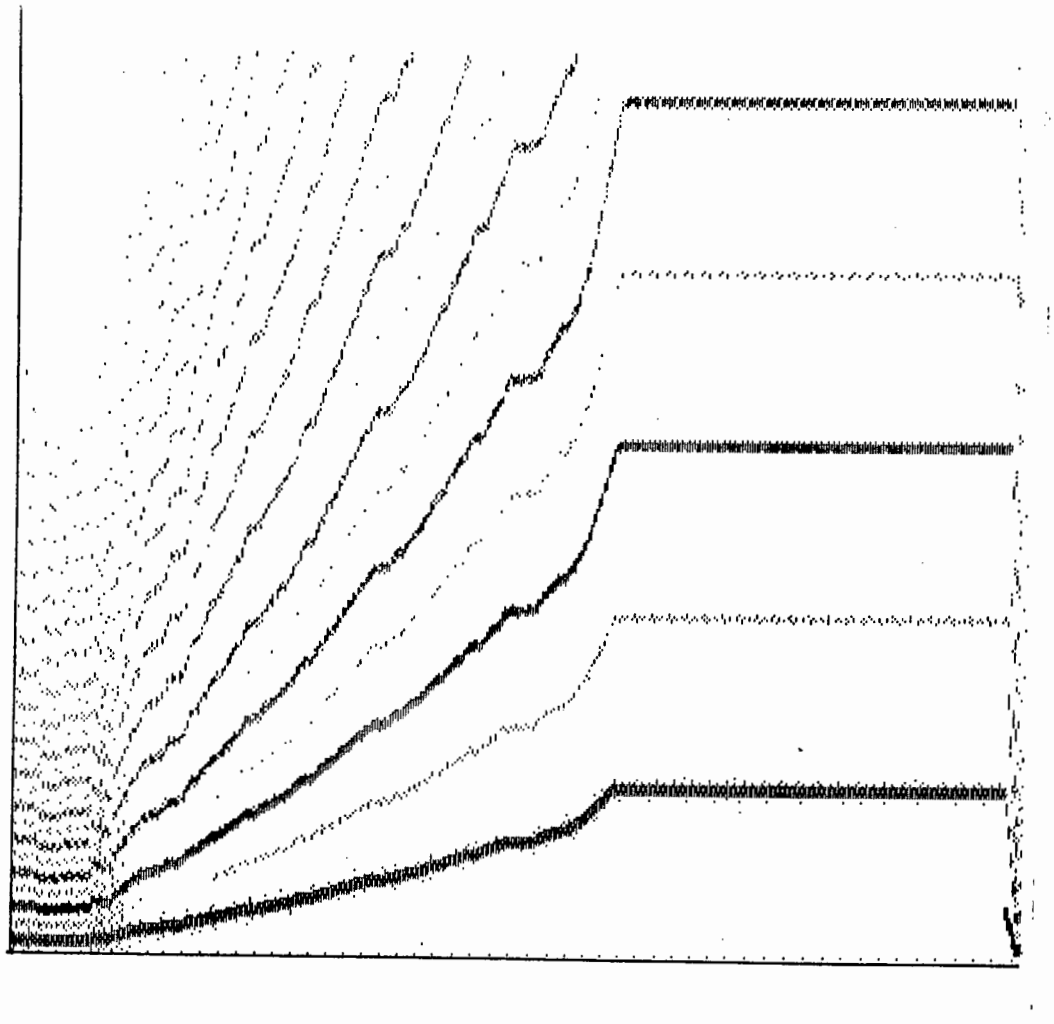


Figure 5.5: Grey scale Waterfall Plot

5.12.2 The Power Cepstrum

The power cepstrum routine is similar to the power spectrum routine, with the addition of two extra steps. After the FFT has been performed and the log of the magnitude calculated, the data is then scaled and converted from bit reversed order to linear order. The complex part is zero filled at the same time. A second FFT is then performed before the data is transmitted to the host.

5.12.3 The Kurtosis Routine

This routine does not actually calculate kurtosis, instead the sum of the squares and the sum of the data to the fourth power are calculated. These are then transmitted to the host where division is performed to obtain the kurtosis. Since the data is to the fourth power, there is a large variation in the magnitude of the results. For this reason, the results are scaled at several stages, and stored as 48 bit rather than 24 bit numbers. A plot of the output of the program for a signal switched from square wave to sine and then triangular wave input is shown in Figure 5.6.

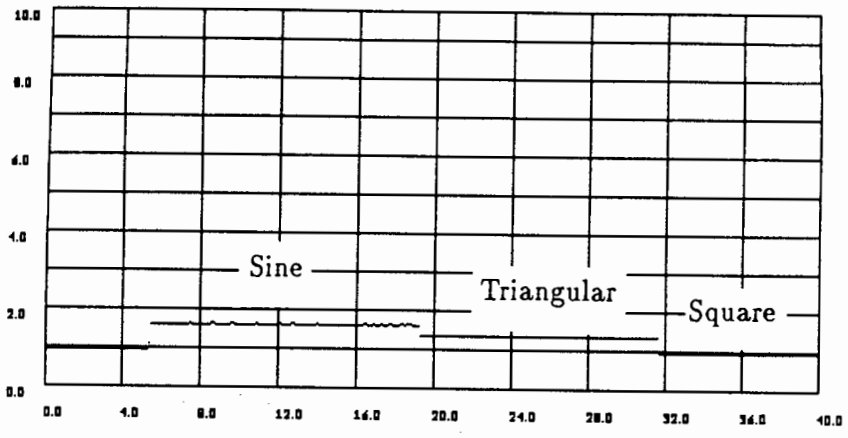


Figure 5.6: Kurtosis of Square Wave, Sine Wave and Triangular Wave

5.12.4 Adaptive Filter Routine

The adaptive filter is a 128 point LMS filter, which operates on a block of 1024 data points. The output is a set of 1024 -128 = 896 adaptively filtered points. The coefficients are stored from block to block, and are not reinitialised with each block of data. This allows the filter to continuously adapt, without having to start from the beginning each time. This, however, is a compromise solution. The filter would be more effective if it could run continuously, and not on batches of data. This would require the capability for the filter to run between sampling intervals. Using the current polled IO system, there is the possibility that a sample would be missed, thus for this to operate, a buffered or interrupt driven IO system would be preferable. An adaptive filter would normally be used as a preprocessor for another routine. The adaptive filter would be allowed to run for some time to allow the filter weights to settle, then the data would be used by some other processing routine. Ways that this can be accomplished are discussed in Section 5.14.

5.13 Decimation and Zooming

Decimation and FFT zooming are not analysis techniques, but rather techniques to aid in narrow band frequency domain analysis. Decimation is a technique for lowering the sampling rate of a signal. The incoming data is lowpass filtered, then subsampled at a lower rate. The two processes of filtering and subsampling are usually combined into one, to avoid filtering data that will be discarded. This technique can be used to reduce the sampling rate by any integer factor. (Non integer decimation is possible, but not as easily accomplished.) This allows varying sampling rates with an ADC operating at a constant frequency.

Zooming is used in combination with decimation to achieve narrow band resolution with an FFT. It is not always desirable to have a power spectrum display that starts at zero

frequency. Yet displaying only the desired portion of a power spectrum does not have the required resolution. The zooming process shifts the desired lowest frequency to the base band using modulation. The process works a follows.

The data is multiplied by a complex exponential of the form:

$$e^{j\omega n} = \cos(\omega n) + j\sin(\omega n) \tag{5.1}$$

This has the effect of shifting the frequencies to be centred around the carrier, as shown by the following equation from [36].

$$DFT(x(n)) = X(k) \tag{5.2}$$

Once the signal has been modulated by the exponential:

$$DFT\left(x(n)e^{j\left(\frac{2\pi}{N}Mn\right)}\right) = X(k-M) \tag{5.3}$$

Due to the wrap around properties of the DFT, the low frequencies move into the high frequency portion of the spectrum. By combining the modulation with decimation, a zoom capability is achieved. It should be noted that after multiplication by the complex exponential, the signal is complex, and the results of the FFT are no longer symmetrical. The problem of zooming was discussed by Marc Levy as part of his undergraduate thesis. It was found that a zoom of a factor of 64 would be necessary to achieve a 1 kHz power spectrum bandwidth when the data was being sampled at a rate of 100 kHz. This meant that to get 1024 points to perform the FFT, 65536 samples would have to be processed. In addition to this, the storage needed for the data and intermediate results would be 144 k words of memory. This amount of memory is greater than the 128 k bytes of data memory that can be addressed by the DSP 56000. Suggestions were made as to how the memory range could be expanded using paged memory. However extending the addressable memory is not strictly necessary. If one bears in mind two things:

- The only data of interest is the resulting 1024 points for the FFT. The rest of the original data is discarded.
- The 64 k of data would take only 50 ms to process, but 0.66 seconds to gather.

It is apparent from these two points that the data can be modulated and decimated while it is being recorded, and the intermediate data need not be stored. This can be accomplished using a buffered ADC input system.

5.14 The Need for Buffered Sampling

From the discussion of the data preprocessing (adaptive filtering and zooming) it is apparent that preprocessing the data as is is sampled is a very useful capability. Processing time is

reduced, and less memory is used. The current ADC IO system is based on polled sampling. Under program control the data ready flag is repeatedly checked until the data is ready. Provided the time spent processing is always shorter than the sampling period, the data can be processed and the next sample input once it is ready. As the time taken to process one sample of the data becomes close to the sampling interval, there is the possibility that the polled sampling system will miss a sample. The solution to this problem is to buffer the output of the ADC with a FIFO (First In, First Out) buffer. The data would be transferred from the ADC to the buffer at regular sampling intervals. The processing routine can then use the samples at a varying rate. It is also possible for the data to be transferred out of the buffer at a lower rate than it is input. The buffer would slowly fill up, acting as temporary storage for the data. These buffers are available in lengths of up to 32 kilobytes. For most purposes, a buffer one kilobyte in length would be sufficient.

Consider the case of a routine that performs an eight to one decimation followed by a Fourier transform:

```
loop 1024 times:
....Read in 8 samples from the FIFO, place them in the filter buffer
....loop 256 times
........multiply filter coefficient N by sample N
........add the result to the total
........increment N
....place the total in the output buffer
Perform FFT
```

From this it is can be seen that although 8192 samples are processed, the total memory space used is 1024 words for output buffer, 256 words for the filter coefficients and 256 words for the data in the FIR filter, making a total of 1536 words of memory needed. The input buffer need only have space for storage of nine samples,as the data can be processed within the time taken for the eight samples to be received.

5.15 Conclusions

1. A portable 486 based computer with a DSP card would make a suitable platform for a real time condition monitoring system.

2. Analogue front end amplifiers will be required for the ADC subsection, in order to protect the analogue to digital converters and amplify the input signals to suitable levels.

3. The Motorola 56000 processor is a suitable signal processing chip, able to perform the required signal processing algorithms.

4. The proto56 card is not adequate for our purpose. The program memory is insufficient, and the data memory cannot be expanded above 32 k words.

5. A buffered ADC system is necessary to allow on the fly preprocessing of the incoming data.

5.16 Recommendations

From these conclusions the following recommendations can be made:

1. A portable computer to use as a basis for the analysis system should be purchased.

2. A new DSP card suiting our requirements should be purchased. DSP cards are available with sufficient data and program memory for our needs. In addition these cards are available with built in ADCs. A card fitting all these requirements should be purchased as the main signal processing card.

3. If input amplifiers are not provided with the card, a variable gain front end amplifier should be built or purchased.

Chapter 6

Overall Conclusions and Recommendations

6.1 Restatement of Objectives

In Chapter 1 the objectives were stated as follows:

The Central Acoustics Laboratory at UCT is researching condition monitoring techniques. We aim to achieve the assessment of machine condition and remaining machine life using a one–off measurement of the vibrations, without a need for detailed knowledge of the machine in question and with minimal reliance on data trending. While a great many techniques for condition monitoring exist, they each have their own strong points and weaknesses. It is hoped that by utilising a combination of these techniques, a system of analysis can be developed that will be suited to a wide range of conditions. Condition monitoring techniques can be developed for analysis of a variety of machine elements, for example gears, fans, rolling element bearings and journal bearings. In this way most rotating machinery could be assessed. By making use of fast portable computers and signal processing equipment, the necessary tests can be carried out on site, in the shortest possible time, to provide immediate answers on machine condition.

The objectives of this thesis are to review the signal processing techniques available, to determine their effectiveness, ease of computation and ease of use. Different methods of combining these techniques for greater effectiveness will also be investigated. An assessment of the suitability of these techniques for incorporation into a computer based system for the determination of the condition of rolling element bearings will be made, as well as an assessment of the hardware requirements.

6.2 The Signal Processing Techniques

Chapters two and three reviewed and evaluated signal processing techniques for use in a condition monitoring system. It was found that:

- The time domain statistics were the simplest techniques to use and involved the least computation, typically of the order of N calculations. These techniques are not always reliable. RMS measurements may fail to detect the early signs of damage, while techniques which measure the impulsiveness of a signal may fail to detect signs of advanced damage.
- The power spectrum and other frequency domain techniques are more complex to use and evaluate than the time domain statistics. They also require slightly more computation time, typically N Log(N) calculations, with N = 1024 or 2048.
- The transpectral coherence and bicoherence (bispectrum) based techniques were also evaluated. It was found that too little is known about their use in condition monitoring to form any definite conclusions about their usefulness. The computation and memory requirements for these techniques is beyond the capabilities of most real time analysis systems.

It is recommended that further research into the bispectrum and coherence techniques be under taken, to determine the effectiveness of these techniques in analysing machine vibrations.

6.3 Hardware Requirements

The hardware requirements for a real time analysis system were reviewed. It was found that:

- A portable IBM compatible compatible PC and DSP card make a suitable signal analysis platform for real time condition monitoring..
- The Motorola 56000 processor is suitable for implementing the commonly used signal processing algorithms.
- The currently available implementation of the M 56000 processor, a Proto 56 card is not suitable for the proposed system. There is only 512 words of program memory available, at least 1024 words of memory are needed for our application.
- The current polled ADC system is not sufficient for our needs. A buffered sampling system is necessary to allow on the fly preprocessing of the incoming data.

From these conclusions it is recommended that a portable PC be purchased for use in the project and that a new DSP card with increased program memory and buffered ADC systems should be purchased.

6.4 Artificial Intelligence and Condition Monitoring

Artificial intelligence is used in a wide range of analysis and classification applications. A previous thesis [27] has concentrated on an expert system for condition monitoring. This however did not make use of any frequency domain analysis techniques. A practical expert system should make use of some aspects of frequency domain analysis for increased reliability. Neural networks are being used in a wide range of processing and classification applications, and may be suitable for use in condition monitoring applications, but have as yet not been tested in this application at UCT.

It is recommended that further research into artificial intelligence be undertaken, both in expert systems and neural networks.

6.5 Suggestions for Continuation of this Project

There are several areas of this project that will benefit from further work. The more unusual signal processing techniques should be tested using vibrations from a range of bearings, in order to determine whether these techniques can be useful in a condition monitoring system. It has also been suggested that an improved signal processing card be purchased. Programming the signal processing techniques for this card is another aspect of this project that should be continued.

6.6 Other Aspects of the Overall Objective

This project has concentrated on rolling element bearings. Other thesis work at UCT has dealt with aspects of condition monitoring of rolling element bearings. Studies of condition monitoring of other machine components should be undertaken, to build up a range of tools for analysis of all the common machine components. This will then help reach the overall objective which is condition monitoring of all types of machines.

Appendix A

Details of the First Bearing Test

This is a description of the first bearing test on the 17 December 1991. The bearing load was 22 kN and the shaft speed 1800 rpm. The lubricant used was grease. Vibrations were recorded using the Proto56 card and the Hewlett Packard spectrum analyser. Vibrations recorded with the Proto56 card have a VIB suffix, while those recorded with the spectrum analyser have a DAT suffix.

	Day 1, 17 Dec
Time	Action
10:34	Test Started, The amplifier gain set to 1*10
10:50	The amplifier gain reduced to 5*10
10:55	The test was halted
11:54	The test was restarted
12:46	The test was halted and the amplifier gain set to 2*10
13:34	The test was restarted
14:14	Test stopped and amplifier set to 5*10
14:38	Test started
15:05	Test stopped The bearing load was dropped to 18.5 kN
15:23	Test started
15:53	Test stopped

File Names and Creation Times		
File	DATE	Time
TESDAT0.VIB	12-17-91	10:32
TESDAT1.VIB	12-17-91	10:52
TESDAT2.VIB	12-17-91	12:13
TESDAT3.VIB	12-17-91	12:33
TESDAT4.VIB	12-17-91	13:41
TESDAT5.VIB	12-17-91	13:51
TESDAT6.VIB	12-17-91	14:01
TESDAT7.VIB	12-17-91	14:11
TESDAT8.VIB	12-17-91	14:45
TESDAT9.VIB	12-17-91	14:55
TESDAT1.VIB	12-17-91	15:30
TIME1.DAT	12-17-91	10:43
TIME2.DAT	12-17-91	10:49
TIME3.DAT	12-17-91	10:55
TIME4.DAT	12-17-91	11:58
TIME5.DAT	12-17-91	12:12
TIME6.DAT	12-17-91	12:23
TIME7.DAT	12-17-91	12:34
TIME8.DAT	12-17-91	12:45
TIME9.DAT	12-17-91	13:38
TIME10.DAT	12-17-91	13:50
TIME11.DAT	12-17-91	14:03
TIME13.DAT	12-17-91	14:11
TIME14.DAT	12-17-91	14:48
TIME15.DAT	12-17-91	15:03
TIME16.DAT	12-17-91	15:28
TIME17.DAT	12-17-91	15:41
TIME18.DAT	12-17-91	15:51

CAPT1.DAT	12-18-91	15:00
CAPT2.DAT	12-18-91	15:01
CAPT3.DAT	12-18-91	15:04
BEAR2HI	12-18-91	15:19

CAPT1 = b2 low
CAPT2 = b2 hi
CAPT3 = b3
CAPT4 = deleted
CAPT5 = frame near bearing
CAPT6 = frame near base

The bearing, when removed from the rig showed the following marks and damage:

1. The balls of the bearing showed three darker bands, one around the circumference of the ball and one either side of this center band. This indicates that the balls were rolling about one axis. The ball diameter was 7.94 mm across the largest diameter. There was little or out of roundness in the balls.

2. The outer race had two grooves worn into it from the action of the balls. these grooves were deepest on one side of the race, probably the side bearing the load. There were also a number of small indentations evenly spaced on the race, visible where the groove was shallower. These would appear to have been caused by the pressure of the balls on the race when the shaft was stationary.

3. The inner race is constructed with two deep grooves for the balls to run in. The entire inner race had a slightly yellow discolouring, except for a slightly bluish band worn towards the center of the inner race in each of these grooves. Within this band there was slight colour variation, there was a darker central stripe, on either side of it another dark stripe with lighter stripes separating them.

Appendix B

Details of the Second Bearing Test

Description of the test of the second bearing. The test was started on Tuesday 18th Feb 1992. The Accelerometer calibration is in g's. Initial amplifier gain 1 mechanical unit per volt. Chan 1 = reference bearing 1, Chan 2 = test bearing . The motor speed was 1700 rpm, load 22 kN. The lubricant used was grease, which was later changed to oil.

Time	Action Taken
11:31	Test started
11:41	stopped
12:30	Test restarted. CAPT 1 made
	Amplifier set to 5 mechanical units per volt
	kurtosis approximately 3
	Bearing housing temperature 55 deg
	CAPT 4, channel two was set on the other ref bearing
	Test paused after CAPT 4, rig now very noisy
	damage has probably occurred.
13:18	test restarted
	Amplifier gain reduced to 20 mechanical unit per volt,
	CAPT 5 made
	end of the days test

Day Two

11:05	Start of testing on day 2
	CAPT 1 made – single channel high frequency capture;
	amplifier gain 0.1 mv/g
	CAPT 2 made – single channel high frequency capture;
	amplifier gain 0.01 mv/g

	Vrms = 500mv
11:24	grease melted and ran out: test stopped.

At this point the bearing was removed and photographed, one side of the outer race was badly pitted.

Day three, lubricant changed to oil.

Time	Action Taken
13:52	test started
	CAP1 made, amplifier gain 20 mechanical units per volt
	Test bearing housing temperature 115 deg
	support bearing housing 85 deg
14:02	Test stopped. The bearing seems to have shifted slightly on shaft
14:40	Test restart, cap 4 made with amplifier gain 50 mechanical units per volt
	CAP 5 made.
	amplifier gain reduced to 100 mechanical units per volt
14:55	The test was stopped

The bearing was removed and photographed:

- Both sides of the outer race were badly pitted in load zone, through an angle of approximately 120 degrees. The non-load side showed slight wear but no damage.
- The balls showed no visible signs of wear or damage. The inner race showed a polished area from the passage of the balls but no visible damage.

List of Files and Creation Times		
Name	Date	Time
BEAR000.vib		
BEAR1.VIB		
BEARB1.vib		
CAPT1	2-18-92	12:34
CAPT2	2-28-92	12:51
CAPT3	2-28-92	12:51
CAPT4	2-28-92	12:53
CAPT5	2-28-92	13:20
CAPT1	2-19-92	11:06
CAPT2	2-19-92	11:09
CAPT3	2-19-92	11:22
CAP1	2-20-92	13:55
CAP2	2-20-92	14:03
CAP3	2-20-92	14:42
CAP4	2-20-92	14:47
CAP5	2-20-92	14:51

Appendix C

Details of the Third Bearing Test

Discription of the third bearing test. Date: Mon 2 march 1992. Lubricant used OIL. The vib files are called bearcx vib. Motor speed 1700 rpm.

Time	Action
11:44	Start of Test. Amplifier gain 2 mech unit per volt
	CAPT 1
	CAPT 2 gain set to 5 mech unit per volt
	CAPT 3 temp 85 deg
	CAPT 4
	CAPT 5 hi freq, 1mV/mech unit temp 105 deg
	CAPT 6 temp 118 deg
	CAPT 7 temp stable at 125 deg
12:38	Amplifier gain changed to 10 mech unit per volt
	CAPT 8 hi fr 1mV/mech unit
	CAPT 9 temp 130 deg
	CAPT 10 temp 138 deg
	CAPT 11 (high frequency capture) temp 135 deg
	CAPT 13 machine at a slower speed the speed was 1613 rpm
14:19	Capture 4 deleted to make room for capture 16
14:20	rig stopped
	CAPT 17 made with speed 767 rpm (note charge amp may have saturated) also 18
	CAPT 19 speed 767 rpm load 14 kN

The test was stopped at this point. There was a single spalling spot on the outer race.

Second day of test three

Time	Action

10:45 Test started at 10:45
CAPT 20 load 767 kN
CAPT 20 repeated amp fault
CAPT 22 same load as 20
10:52 load put up to 22kN
1100sec Note charge amp fault, temp 106 deg
CAPT 24 temp 112 deg
11:19 load dropped to 14 kN
11:21 back up to 22 kN
Note are there 23Hz side bands of bpfo?
CAPT 26 temp 127 deg
CAPT 27 hi fi
CAPT 28 high peak at 780 hz
Time trace shows near sinusoid at this fr.
temp = 137 deg
11:45 gain changed to 20 mech unit per volt
CAPT 28 caught something strange :
possibly bearing moving in plummer block
11:51 temp 155 deg
gaim to 50 mupv
CAPT 31 temp 166 deg
CAPT 32
12:00 test stopped

Note: kurtosis seems to vary with load, kurtosis is higher with the lower loads

The bearing at this point showed spalling at several points along one side of the outer race, through an angle of approximately 90 degrees.

Spectrum Analyser Files from Day 1		
Name	date	Time
CAPT1.DAT	3-02-92	
CAPT2.DAT	3-02-92	11:52
CAPT3.DAT	3-02-92	12:00
CAPT4.DAT	3-02-92	erased
CAPT5.DAT	3-02-92	12:11
CAPT6.DAT	3-02-92	12:26
CAPT7.DAT	3-02-92	12:35
CAPT8.DAT	3-02-92	12:43
CAPT9.DAT	3-02-92	13:06
CAPT10.DAT	3-02-92	13:22
CAPT11.DAT	3-02-92	13:42
CAPT12.DAT	3-02-92	13:48
CAPT13.DAT	3-02-92	13:54
CAPT14.DAT	3-02-92	14:02
CAPT15.DAT	3-02-92	
CAPT16.DAT	3-02-92	14:23
CAPT16a.DAT	3-02-92	14:33
CAPT17.DAT	3-02-92	14:34
CAPT18.DAT	3-02-92	14:35
CAPT19.DAT	3-02-92	14:37

Spectrum Analyser files from Day 2		
Name	Date	Time
CAPT20.DAT	3-04-92	10:43
CAPT21.DAT	3-04-92	10:49
CAPT22.DAT	3-04-92	10:51
CAPT23.DAT	3-04-92	11:06
CAPT24.DAT	3-04-92	11:16
CAPT25.DAT	3-04-92	11:19
CAPT26.DAT	3-04-92	11:31
CAPT27.DAT	3-04-92	11:35
CAPT28.DAT	3-04-92	11:41
CAPT29.DAT	3-04-92	11:48
CAPT30.DAT	3-04-92	11:53
CAPT31.DAT	3-04-92	11:57
CAPT32.DAT	3-04-92	12:00

VIB Files from Day 1 and 2		
BEARC0.VIB	3-02-92	11:23
BEARC1.VIB	3-02-92	11:43
BEARC2.VIB	3-02-92	12:03
BEARC3.VIB	3-02-92	12:23
BEARC4.VIB	3-02-92	12:43
BEARC5.VIB	3-02-92	1:03
BEARC6.VIB	3-02-92	1:23
BEARC7.VIB	3-02-92	1:43
BEARD0.VIB	23-04-9	10:18
BEARD1.VIB	23-04-9	10:38
BEARD2.VIB	23-04-9	10:58
BEARD3.VIB	23-04-9	11:18

Appendix D

Characteristic Frequencies for Steyr 1207 ball bearing

Number of balls per race	16
Ball diameter	7.95 mm
Pitch diameter	53mm
contact angle	1 deg

Table D.1: Bearing Dimensions

RPM	R (hz)	FTF	BPFI	BPFO	BSF
60	1.00	0.43	9.20	6.80	3.26
600	10.00	4.30	92.00	68.00	32.60
1000	16.70	7.08	153.33	113.30	54.31
1700	28.33	12.04	260.66	192.67	92.32
1800	30.00	12.75	276.00	204.00	97.75
1900	31.67	13.46	291.33	215.34	103.18

Table D.2: Characteristic Frequencies in Hertz

It should be noted that these dimensions are not the same as those of the bearings used by [27] even though they have the same type number and load ratings.

Appendix E

Bearing Component Resonance Frequencies

The vibrations of the components of a bearing can be calculated using the following equations[1] from [17]:

For the resonant frequencies of the balls:

$$F_b = \frac{0.424}{r} \times \sqrt{\frac{E}{2\rho}} \tag{E.1}$$

For the resonant frequencies of the races:

$$F_r = \frac{n(n^2-1)}{2\pi\sqrt{n^2+1}} \times \frac{1}{a^2} \times \sqrt{\frac{EI}{m}} \tag{E.2}$$

- ρ = density of ball
- r = radius of ball
- m = mass of ring per metre
- a = race radius to the neutral axis
- E = modulus of elasticity (Youngs modulus)
- I = Moment of inertia of race cross section about neutral axis
- n = Number of waves around the race n = 2,3,4...

Various other formulations exist, depending on the shape of the components, the assumptions made, and the type and mode of vibrations expected. Several different equations can be found in [37].

[1]These are similar to those from [37], but certain assumptions and changes have been made to suit this particular case.

For a circular ring, with displacements in the plane of the ring:

$$F_r = \frac{n(n^2-1)}{\sqrt{n^2+1}} \frac{1}{4\sqrt{\pi}} \times \frac{c^2}{a^2} \times \sqrt{\frac{E}{m}} \tag{E.3}$$

or, for a thin ring [2]:

$$F_r = \frac{n(n^2-1)}{\sqrt{n^2+1}} \frac{1}{2\pi} \times \frac{1}{a^2} \times \sqrt{\frac{D}{2\rho h}} \tag{E.4}$$

Where D is given by:

$$D = \frac{2}{3} \frac{Eh^3}{1-\sigma^2} \tag{E.5}$$

- c = radius of ring cross section.
- h = half the ring thickness

Calculations for 2207 bearings:

- $\rho = 7000 kg/m^3$
- $r = 3.98$ mm
- $m = 137 \times 10^{-3}$ kg/m (outer race) 87.9×10^{-3} kg/m (inner race)
- $a = 32$ mm (outer race) or 21 mm (inner race)
- $E = 207 \times 10^9$
- $I = 1$
- $c = 2.5$ mm or 2mm

The ball resonant frequency using equation E.1 is : $F_b = 410$ kHz

For the inner and outer races, the frequencies for the different modes, calculated using equation E.3 are:

n	2	3	4
Outer	2.8 kHz	8.0 kHz	15.4 kHz
Inner	5.3 kHz	14.8 kHz	28.5 kHz

Table E.1: Resonant Frequency of Bearing Components

[2]Due to the assumptions made in deriving equation, it is not an accurate formulation

These calculations are not necessarily accurate, but they do give the correct order of magnitude. From these we can deduce that for most bearings, the resonant frequency of the balls is outside the measured range. The outer race resonant frequency is the lowest, in this case 2.8 kHz. Figure E.1 shows the measured outer race resonances. The race was suspended and struck with a plastic rod. The power spectrum of the resulting sound shows peaks at 3.2 kHz, 5.8 kHz and 8.9 kHz. The two dominant peaks, 8.9 kHz and 3.2 kHz have the frequency ratio of 2.8. This is the expected ration of the first and second modes of vibration. The other frequencies present may be caused by other modes, such as vibration normal to the plane of the ring.

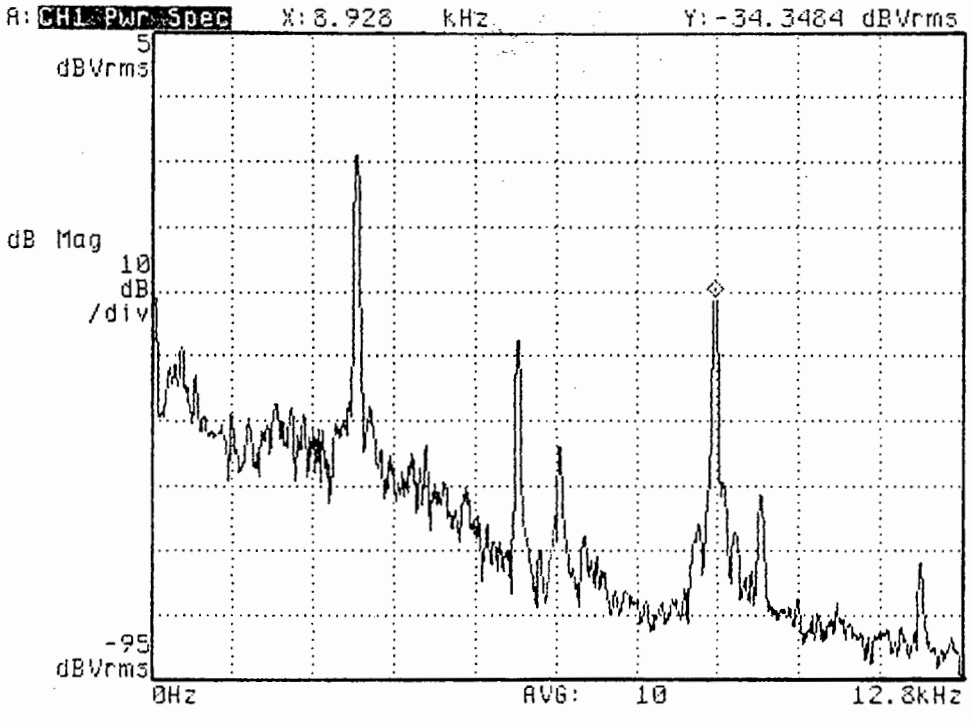


Figure E.1: Power Spectrum of Outer Race Resonance

Appendix F

Schematics and PCB for ADC Circuit

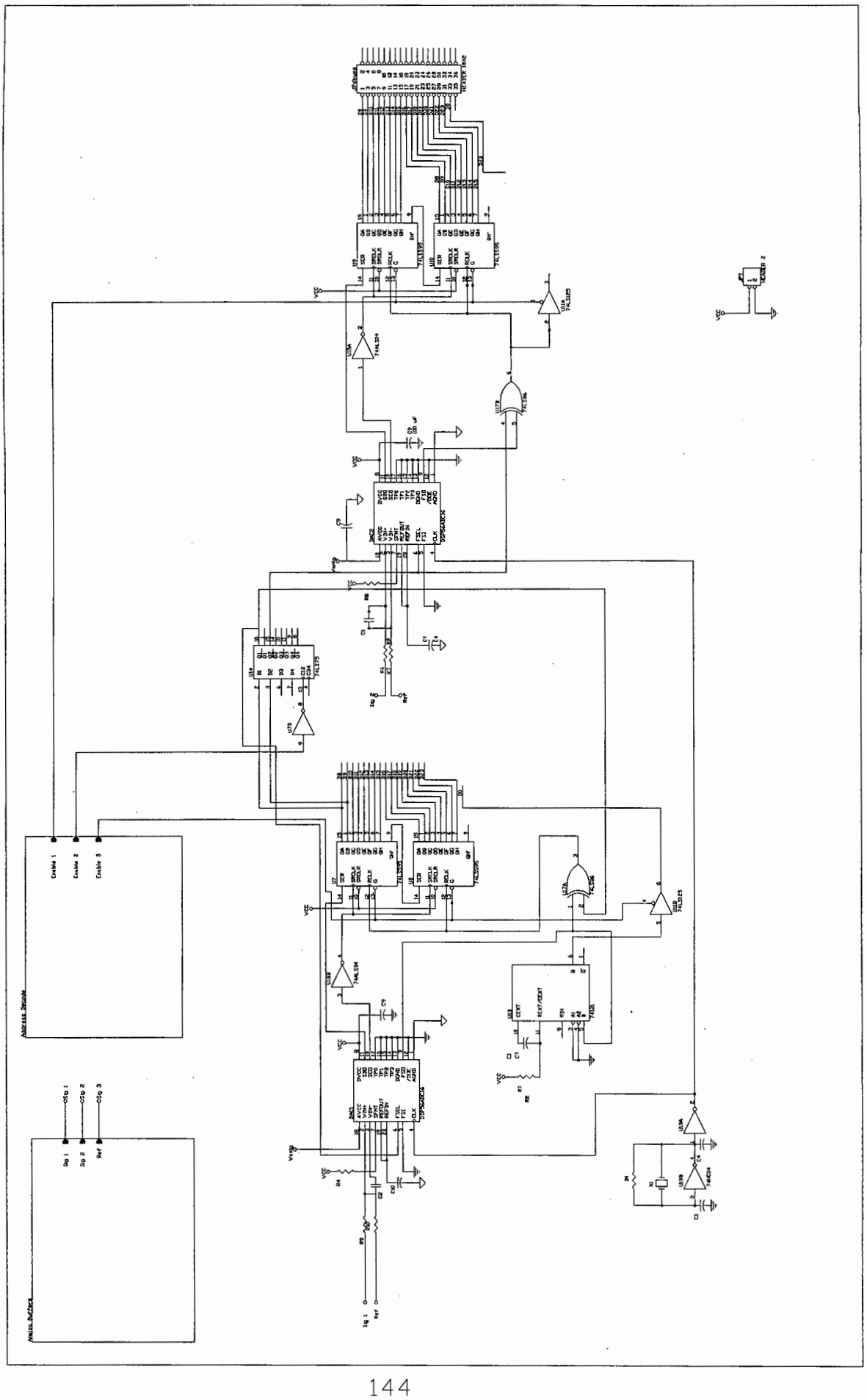


Figure F.1: Circuit Diagram of ADC Circuit

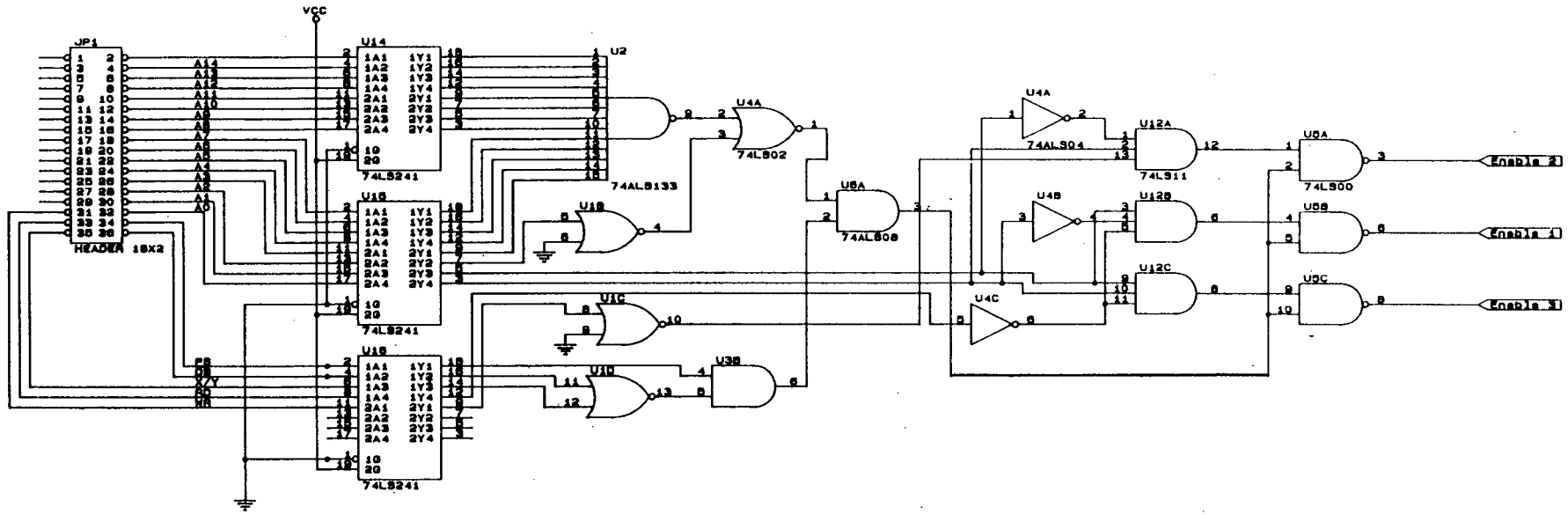
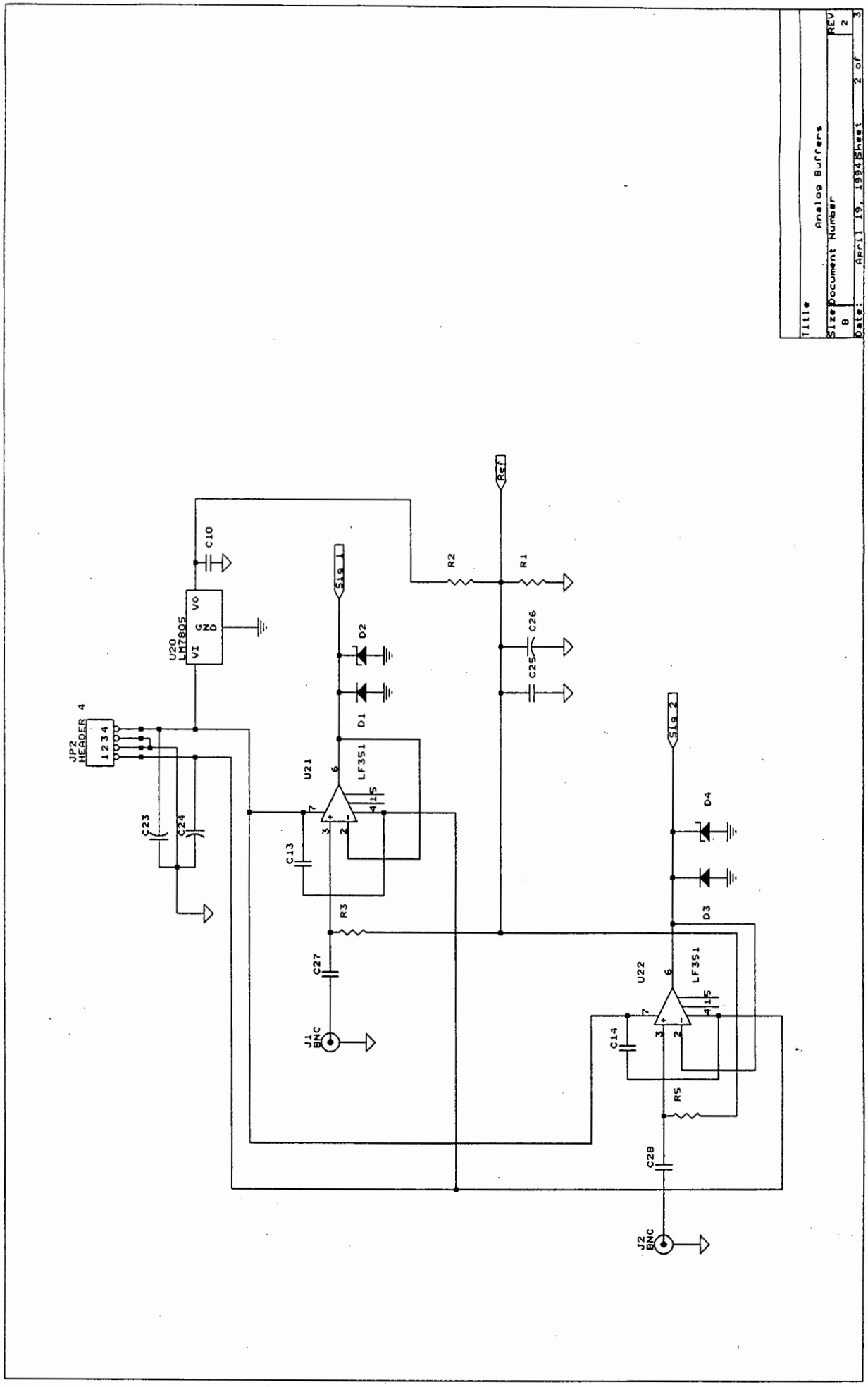


Figure F.2: Circuit Diagram of ADC Address Decode

Figure F.3: Circuit Diagram of ADC Analog Input



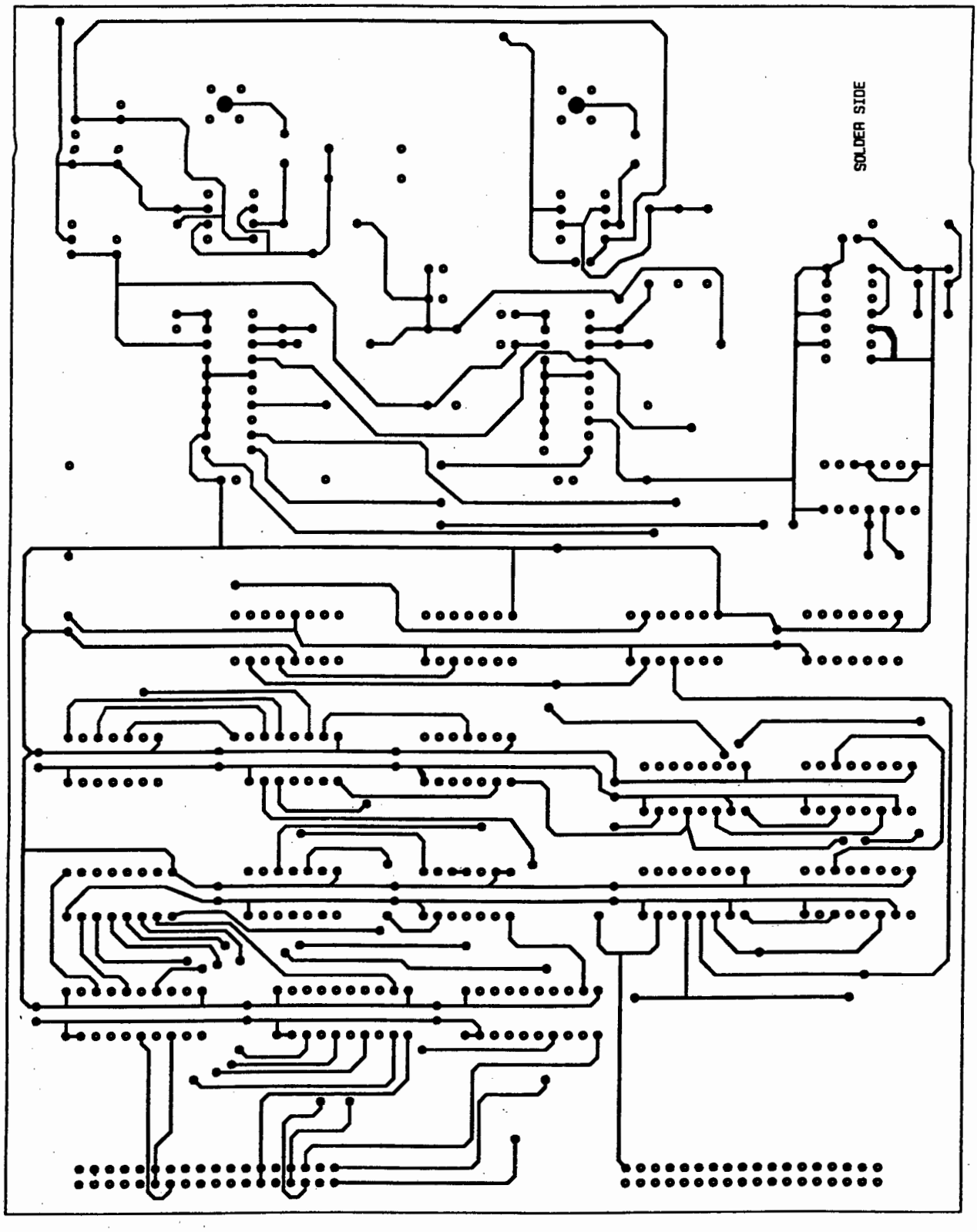


Figure F.4: PCB solder side

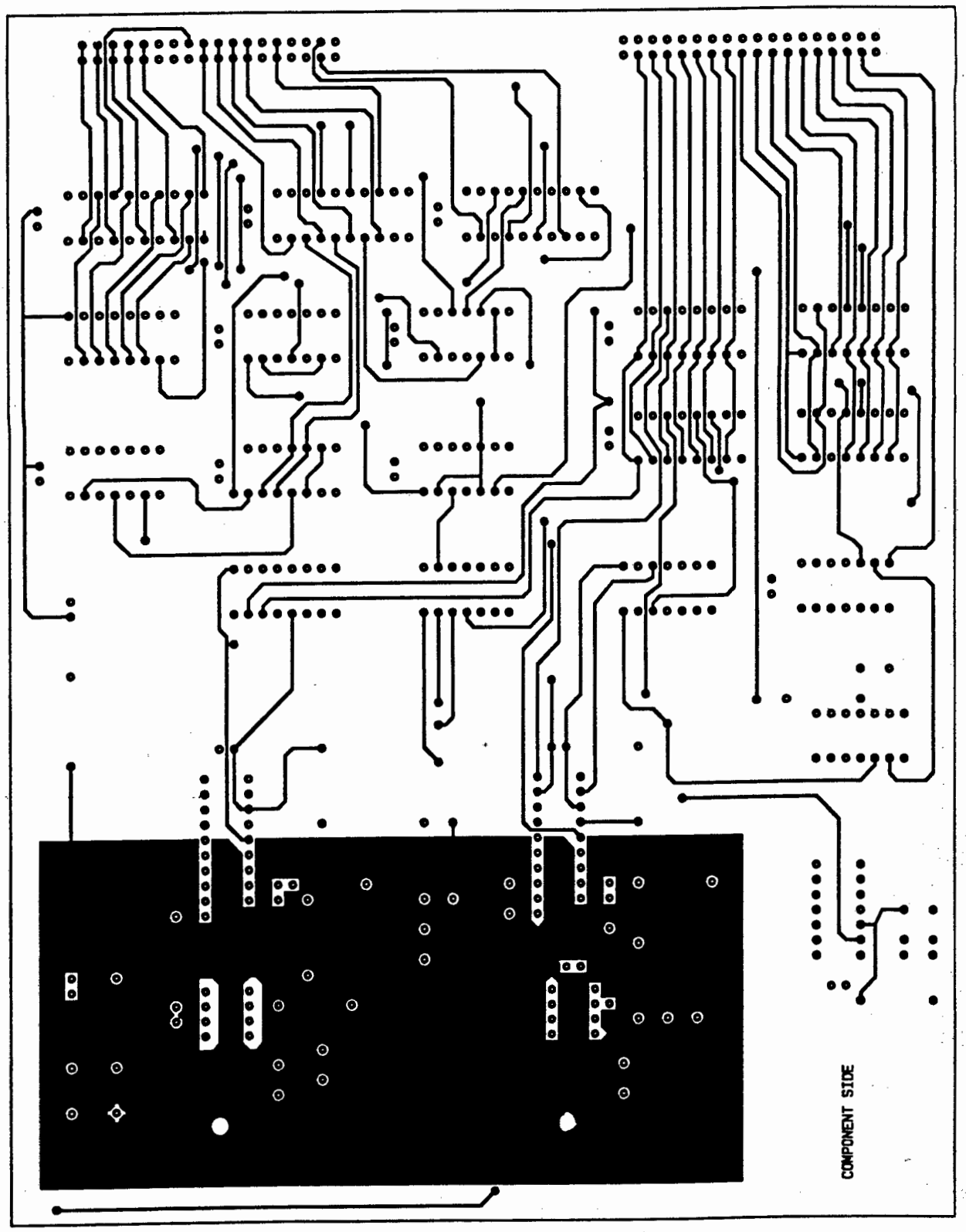


Figure F.5: PCB Component side

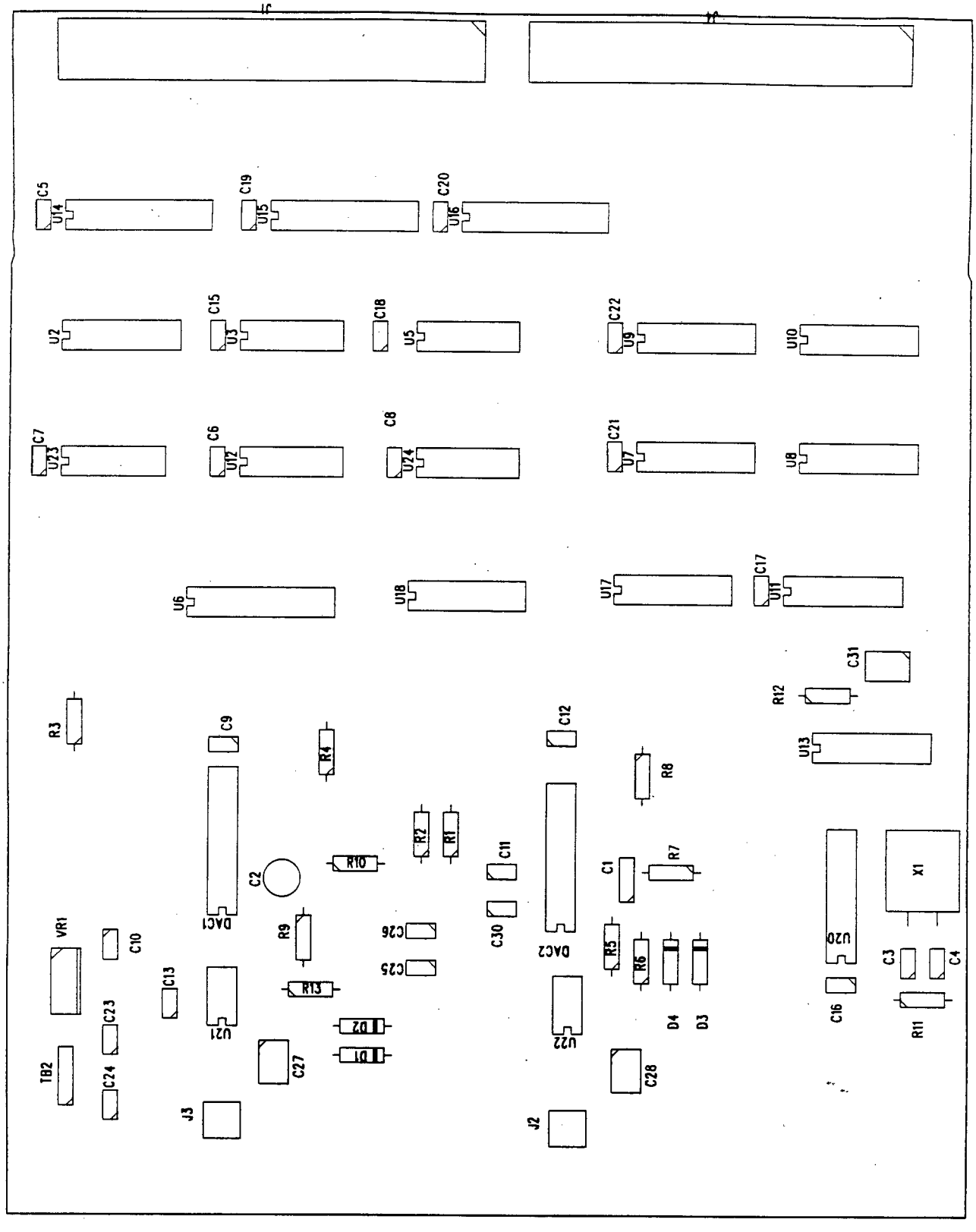


Figure F.6: PCB Component Placement

Appendix G

Source Code Listings

G.1 Shell Program Listing

```
#include "c:\davbir\c\graphru.c"        /* graph and mouse routines */
#include "c:\davbir\c\menuinc.c"        /* menu routines */
#include "c:\davbir\c\gsplot.c"         /* grayshade mapping and display*/
#include "c:\davbir\c\lasdumpb.c"       /* print routine */
#include "c:\davbir\c\asm\window.c"     /* FFT window calculation */
#include <stdlib.h>
#include <string.h>

/*#define                NOCARD  1*/   /*this should be defined if */
                                        /*the card is not present   */
                                        /* eg for test purposes     */


/*General function declarations */

void swap(void);


/*declarations of menu functions*/
/*all  take an int parameter */
/* and return  void */
void print(int itemno);
void exprg(int itemno);
void wind(int itemno);
void dispt(int itemno);
void start(int itemno);
void ret(int itemno);
void gp(int itemno);

/*declarations of processing functions*/
/*parameter and return of type void */
void donothing(void);
void prfunc1(void);
void prfunc2(void);
void prfunc3(void);
void prfunc4(void);
void prfunc5(void);
```

```
/* global variables */
void (*proutine)(void) = donothing;
char progpath[80];
int currentmenu = 0;
int visualpage =0,activepage=0;
int displaytype = 0;
long win[1024];
struct menu_elem menblock[5][8]={
"1 disp1", dispt,"2 disp2",dispt,"3 kurtosis",dispt,"4 cepstrum",dispt,
"5 lms",dispt,"6 print",print,"7 start",start,"8 exit",exprg,
"1 rectangle",wind,"2 Hanning",wind,"3 Hamming",wind,"4 Blackman",wind,
"5 return",ret,"6",ret,"7",ret,"8 return",ret,
"1 B&W Print",gp,"2 Grey",gp,"3",NULL,"4",NULL,"5",NULL,"6",NULL,
"7",NULL,"8 return",ret,
"1 ",NULL,"2 ",NULL,"3",NULL,"4",NULL,"5",NULL,"6",NULL,"7",NULL,
"8 return",ret,};

void main(void)
{
int mousex,mousey,mousebut,status=1;
int choice;

initi();                                /*initialise graphics*/
drawmenu(menblock[currentmenu]);
blackman(win,1024);
resetmouse(&mousebut);                  /* reset the mouse then display it*/
showmouse();
dispt(0);                               /* set default dispaly */
fflush(stdin);                          /* clear keyboard buffer */

loopforever:
status =1;
getpresstate(&mousex,&mousey,&mousebut,&status);        /* check mouse*/


if(mousebut != 0)
        {
        choice = (mousey -10)/20;                       /* call the relevant */
        menblock[currentmenu][choice].func(choice);     /* menu function */
        delay(100);
        }

if(kbhit())
        {
        /* Keyboard alternative to menu */
        choice = getch();
        choice = choice - '1' ;
        if((choice>=0)&&(choice<8))                     /*valid keys 1 to 8*/
        menblock[currentmenu][choice].func(choice);
        fflush(stdin);
        delay(10);
        }
proutine();         /* call the relevant processing routine*/
goto loopforever;
}
```

```
/*-------------------------------------------------------------------*/
void swap(void)
{
int temp;

temp = activepage;
activepage = visualpage;
visualpage = temp;

setactivepage(activepage);
setvisualpage(visualpage);

}

/*-------------------------------------------------------------------*/

/* puts labels on both pages*/

void axesboth(float xstart,float xstep, float ystart,float ystep)
{
cleardevice();
Xaxes(xstart,xstep);
Yaxes(ystart,ystep);
swap();
cleardevice();
Xaxes(xstart,xstep);
Yaxes(ystart,ystep);
}


/*-------------------------------------------------------------------*/


void print(int itemno)          /* calls the print routine */
{
currentmenu = 2;
drawmenu(menblock[currentmenu]);
swap();
drawmenu(menblock[currentmenu]);
swap();
}

void gp(int itemno)
{
FILE *outy;
outy = fopen("c:temp","wb");
if (itemno == 0)
dumpbw(outy);
if(itemno == 1)
dump(outy);
fclose(outy);
}

void exprg(int itemno)              /* exit routine */
{
hidemouse();
```

```
closegraph();
restorecrtmode();                    /* close graphics*/
exit(0);                             /* return to dos */
}


/*-------------------------------------------------------------------*/
                                                                      160

void wind(int itemno)                /* calculates the FFT window */
{

 switch(itemno){
        case 0:
                rectang(win,1024);
         break;

         case 1:                                                      170
                hanning(win,1024);
         break ;

         case 2:
                hamming(win,1024);
         break;

         case 3:
                blackman(win,1024);
         break;                                                       180
         }
#ifndef NOCARD
dsp_load(progpath);
dsp_go();
dsp_writelong(win,1024);
#endif

}
/*-------------------------------------------------------------------*/
                                                                      190
/* A very important routine, It loads the dsp program and sets up
graphics and diplay functions*/

void start(int itemno)                    /* Start the selected process */
{                                         /* and display routines */

int menu;

hidemouse();
cleardevice();                                                        200
if(displaytype==0)
        {
        normcol();                        /* set colour mapping */
        visualpage = 0;
        activepage = 1;
        setvisualpage(visualpage);
        setactivepage(activepage);
        axesboth(0,40,-100,10);
```

```
        mousesensitivity(32,32);
        proutine = prfunc1;                    /* set the function pointer */
        strcpy(progpath,"c:\\davbir\\c\\asm\\fftex2");
        #ifndef NOCARD
                dsp_load("c:\\davbir\\c\\asm\\fftex2");
                dsp_go();
                dsp_writelong(win,1024);
        #endif
        swap();
        delay(100);
        menu = 1;
        }

if(displaytype ==1)
        {
        grayshade();                           /* set greyscale mapping*/
        visualpage =0;
        activepage = 0;
        setvisualpage(visualpage);
        setactivepage(activepage);
        proutine =  prfunc2;                   /* set the function pointer*/
        strcpy(progpath,"c:\\davbir\\c\\asm\\fftex2");
        #ifndef NOCARD
                dsp_load("c:\\davbir\\c\\asm\\fftex2");
                dsp_go();
                dsp_writelong(win,1024);
        #endif

        menu = 1;
        }
if (displaytype == 2)
        {
        normcol();
        visualpage =0;
        activepage =1;
        setvisualpage(visualpage);
        setactivepage(activepage);
        proutine = prfunc3;
        strcpy(progpath,"c:\\davbir\\c\\asm\\shkurt" );
        #ifndef NOCARD
                dsp_reset();
                dsp_load("c:\\davbir\\c\\asm\\shkurt");
                dsp_go();
        #endif
        axesboth(0,4,0,1);
        menu =3;

        }

if(displaytype==3)
        {
        normcol();                             /* set colour mapping */
        visualpage = 0;
        activepage = 1;
        setvisualpage(visualpage);
        setactivepage(activepage);
        mousesensitivity(4,4);
```

```
        proutine = prfunc4;                    /* set the function pointer */
        strcpy(progpath,"c:\\davbir\\c\\asm\\cepst");
        #ifndef NOCARD
                dsp_load("c:\\davbir\\c\\asm\\cepst");
                dsp_go();                                                    270
                dsp_writelong(win,1024);
        #endif
        menu = 1;
        axesboth(0,40,-5,1);
        }

        if(displaytype==4)
        {
        normcol();                             /* set colour mapping */
        visualpage = 0;                                                      280
        activepage = 1;
        setvisualpage(visualpage);
        setactivepage(activepage);
        mousesensitivity(4,4);
        proutine = prfunc5;                    /* set the function pointer */
        strcpy(progpath,"c:\\davbir\\c\\asm\\shlms");
        #ifndef NOCARD
                dsp_load("c:\\davbir\\c\\asm\\shlms");
                dsp_go();
                dsp_writelong(win,1024);                                     290
        #endif

        axesboth(0,40,-5,1);
        menu = 3;
        }


drawmenu(menblock[menu]);                      /* draw the menu*/
if (activepage != visualpage)                  /*on all pages used*/          300
        {
        setactivepage(visualpage);
        drawmenu(menblock[menu]);
        setactivepage(activepage);
        }

mousepage(visualpage);
showmouse();
currentmenu = menu;
                                                                             310
}

/*-------------------------------------------------------------------*/

void dispt(int itemno)
{
displaytype = itemno;
}

/*-------------------------------------------------------------------*/       320

void ret(int itemno)                   /* return to previous menu*/
```

```
{
int lastmenu[5]={0,0,0,0,0};

hidemouse();
currentmenu = lastmenu[currentmenu];
drawmenu(menblock[currentmenu]);
if (activepage != visualpage)
        {
        setactivepage(visualpage);
        drawmenu(menblock[currentmenu]);
        setactivepage(activepage);
        }
mousepage(visualpage);
showmouse();
}

/*-------------------------------------------------------------------*/
/* The following routines are data acuisition and display routines      */
/* A pointer is set to point to one of these routines and it is then     */
/* called in the main program loop                                       */


/*.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-*/

void donothing(void)
{
/*A dummy routine that does nothing.
Used at the start before user has selected
The appropriate option*/
}

/*.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-*/

/* Displays spectrum */

void prfuncl(void)
{
long data[512];
int temp;

#ifdef NOCARD
data[0] = -3000000;
        for(temp =1;temp<401;temp++)
                data[temp] = data[temp-1]  - 100000 +
                (long)10 * random(20000) ;

#else
dsp_readlong(data,512);
#endif

drawaxes();
/*Yaxes(-80,10);
Xaxes(0,40);*/
displaylin(data,41008,77,14);          /* display the points*/

hidemouse();
swap();
```

```
mousepage(visualpage);
showmouse();
}

/*.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.- */

/*Display the spectrum as gray shades */


void prfunc2(void)
{
long data[512];
int temp;

#ifdef NOCARD
        data[0] = -3000000;
        for(temp=1;temp<401;temp++)
                data[temp] = data[temp-1] - 100000 + (long)10*random(20000);
#else
dsp_readlong(data,512);                    /* Get data */
#endif

hidemouse();
displaygs(data,500000,6000000);            /* display it */
showmouse();
}

/*.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.- */

/*Displays kurtosis of the wave */

void prfunc3(void)
{
static long data[400];
long tarray[5];
static int place =0;
int temp;
double squar,four;

#ifdef NOCARD
                data[place] = random(100) + 100;
#else
/*read data as 48 bit nos, avg, sigma x^2 and sigma x^4*/

dsp_readlong(tarray,5);
squar = 0;          four = 0;
tarray[1] <<= 8;           tarray[3] <<= 8;
squar = tarray[1];         four = tarray[3];
tarray[2] = tarray[2] & 0x00ffffff;          /*convert to floats*/
tarray[4] = tarray[4] & 0x00ffffff;
squar += tarray[2] / 65536.0;
four  += tarray[4] / 65536.0;

data[place] = 150000 * 1024 *  1024.0 * (four)/(squar*squar);


#endif
```

```
place++;
place %= 400;

drawaxes();
displayp(data,1,-100,13);
hidemouse();
swap();
showmouse();

}

/*.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-*/

/* Display the Cepstrum */

void prfunc4(void)
{
long data[512];
int temp;

#ifdef NOCARD
data[0] = -3000000;
        for(temp =1;temp<401;temp++)
                data[temp] = data[temp-1]  - 100000 +
                (long)10 * random(20000) ;

#else
dsp_readlong(data,512);
#endif

drawaxes();
displaylin(data,100000,0,9);

hidemouse();
swap();
showmouse();
}

/*.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-.-*/

/*Display adaptive filtered data, and the filter*/

void prfunc5(void)
{
long signal[1024],filter[400];
int count;
int temp;

#ifdef NOCARD
signal[0] = 0;
        for(temp =1;temp<401;temp++)
        {
                signal[temp] = -400000 +  (long) 80 * random(20000) ;
                filter[temp] = -400000 + (long) 80 * random(20000);
        }
```

```
#else
dsp_readlong(signal,1024);                    /*get data*/
dsp_readlong(filter,128);
for(count=128;count<400;count++)
        filter[count]=0;
#endif

drawaxes();
displaylin(signal,5000,0,5);                 /*display both*/
displaylin(filter,10000,0,6);

hidemouse();
swap();
showmouse();
}
```

G.2 Graph Routines Listing

```
#include <graphics.h>
#include <stdlib.h>
#include <conio.h>
#include <stdio.h>
#include <dos.h>
#include "c:\davbir\c\asm\dsp.c"
#include "c:\davbir\c\minc.c"


                                                                        10
void drawaxes(void);
void displayp(long points[400],long scale,short os,short colour);
void displaylin(long points[400],long scale,short os,short colour);



/*-------------------------------------------------------------------*/
/*      Function to initialise the graphics system                   */
/*      EGA resolution used, to allow two ,graphics pages            */
                                                                        20

void initi(void)
{
int gdriver = 9, gmode = 1, errorcode;

initgraph(&gdriver, &gmode, "m:\\programs\\bcpp\\bgi");
errorcode = graphresult();

if (errorcode != grOk)              /* an error occurred */
        {                                                               30
        printf("Graphics error:  %s\n", grapherrormsg(errorcode));
        printf("Press any key to halt:");
        getch();
        exit(1);                    /* return with error code */
        }

cleardevice();

}
                                                                        40

/*-------------------------------------------------------------------*/
/*      Clears the current viewport and draws a set of axes          */
/*      in colour 7                                                  */

void drawaxes(void)
{
/*draw 400 200 axes */
int xcount,ycount;
                                                                        50
setfillstyle(0,0);
bar(20,0,420,210);
setcolor(7);

mousesensitivity(9000,9000);
```

```
for(xcount = 20;xcount <= 420;xcount += 40 )
        {
        moveto(xcount,10);
        lineto(xcount,210);
        }

for(ycount = 10;ycount <= 210;ycount += 20)
        {
        moveto(20,ycount);
        lineto(420,ycount);
        }
mousesensitivity(5,5);
}

/*-------------------------------------------------------------------*/
/*      Display 400 points using dots                                */

void displayp(long points[400],long scale,short os ,short colour)
{
int count;

for(count = 0;count<400 ;count++)
        {
        putpixel(count +20, 110 - points[count]/scale - os,colour);
        }
}

/*-------------------------------------------------------------------*/
/*      Display 400 points and connect them with lines               */

void displaylin(long points[400],long scale,short os ,short colour)
{
int count;

/*setwritemode(XOR_PUT);*/
moveto(20,110 - points[0]/scale -os);

setcolor(colour);
mousesensitivity(9000,9000);
for(count = 0;count<400 ;count++)
        {
        lineto(count +20, 110 - points[count]/scale - os);
        }
mousesensitivity(5,5);
}

/*-------------------------------------------------------------------*/

void Xaxes(float start, float step)
{
/*draw 400 200 axes */
int xcount;
char tstring[10];
```

```
setcolor(15);
settextstyle(2,0,2);
for(xcount = 20;xcount <= 420;xcount += 40 )
        {
        sprintf(tstring,"%.1f",start);
        outtextxy(xcount,220,tstring);
        start += step;
        }                                                                    120

}
/*-------------------------------------------------------------------*/

void Yaxes(float start, float step)
{
/*draw 400 200 axes */
int ycount;
char tstring[10];
                                                                             130
setcolor(15);
settextstyle(2,0,2);

for(ycount = 210;ycount >= 10;ycount -= 40 )
        {
        sprintf(tstring,"%.1f",start);
        outtextxy(0,ycount,tstring);
        start += 2 * step;
        }
                                                                             140
}
```

G.3 Grayscale Graphics Listing

```
#include <graphics.h>


/*-------------------------------------------------------------------*/
/*      Set up a grey scale palette                                  */


void grayshade()
{
int k;                                                                  10
struct palettetype pal;

getpalette(&pal);
for(k=0;k<16;k++)
        {
        setrgbpalette(pal.colors[k],k*4,k*4,k*4);
        }
}

/*-------------------------------------------------------------------*/  20
/*      Set up a colour palette                                      */


void normcol()
{
int k;
struct palettetype pal;
int cols[16][3]={0,0,0,                 /*0*/
                 8,8,8,                 /*1*/
                 18,18,18,              /*2*/                           30
                 23,23,23,              /*3*/
                 29,0,0,                /*4 dull red */
                 0,29,0,                /*5 dull green*/
                 0,0,29,                /*6 dull blue*/
                 0,22,20,               /*7 yellow green */
                 20,20,0,               /*8*/
                 1,8,28,                /*9 pale blue*/
                 31,0,0;                /*10 red*/
                 0,31,0,                /*11*/
                 0,0,31,                /*12 blue*/                     40
                 10,20,30,              /*13*/
                 31,0,5,                /*14*/
                 31,31,31};             /*15*/



getpalette(&pal);
for(k=0;k<16;k++)
        {
        setrgbpalette(pal.colors[k],cols[k][0]*4,cols[k][1]*4,cols[k][2]*4);    50
        }
}


/*-------------------------------------------------------------------*/
```

```
/*      Display 350 points as a vertical line af grey dots                */
/*      Subsequent Lines are drawn next to each other untill there        */
/*      are 400, then drawing starts at the screen edge again             */

void displaygs(long data[400],long scale,long os)                          60
{
static int trace =0;
int counter;
int colour;

for(counter = 0;counter <350;counter++)
        {
        colour = (data[counter] +os)/scale;
        if (colour < 0)
                colour = 0;                                                70
        putpixel( trace+10,350-counter,colour );
        }
trace++;
trace %= 400;
}
```

G.4 Cepstrum Program

```
;The cepstrum routine. It is similar to the Power Spectrum routine,
;but has an additional call to the Fourier transform routine
fftr2at         ident   1,1
                page    132,54
                opt     nomd,nomex,loc,cre,nocex,mu
        ;Public domain routines are loaded as macros
        include 'sincos'        ;sine/cosine macro
        include 'fftr2a'        ;fourier tranform macro

strt            equ     0
win             equ     2048            ;start of window
points          equ     1024
data            equ     0
coef            equ     1024            ;start of sine table
pcoef           equ     3072            ;polynomial coefficients
                sincos  points,coef     ;create the sine lookup table

                opt     mex
                org     p:0
                jmp     $100
                org     p:$100


;Initialize
                ;ori    #%00000100,omr
                andi    #%11111011,omr
                move    #$0,x0                  ;Set up 100kHz Sampling rate
                move    x0,y:$fffd
                move    #$2209,x0
                move    x0,x:$fffe              ;set bus control register
                move    #0.9981958,x0           ;store coefficients for
                move    x0,y:pcoef              ;logarithm function
                move    #-0.3372223,x0          ;the log function
                move    x0,y:(pcoef+1)          ;is a polynomial approximation
                move    #-0.6626105,x0          ;which is more accurate than
                move    x0,y:(pcoef+2)          ; a log lookup table
                jsr     rx_win                  ;load window from the host




main            jsr     samplf                  ;sample the input
                jsr     window                  ; window the data
                jsr     four                    ;perform fourier transform
                jsr     logd                    ;find log of data squared
                jsr     four                    ;perform another fft
                jsr     tx_host                 ;transmit x data
                jmp     main                    ;loop forever


;.....................................................................
;routine to recieve a window from the host
rx_win
                movep   #0,x:$ffe8
                move    #win,r1
                move    #1,n1
                move    #-1,m1
                do      #points,end_wrx
wn_rx           jclr    #0,x:$ffe9,wn_rx
                movep   x:$ffeb,y:(r1)+
                nop
```

```
end_wrx
                rts

;-----------------------------------------------------------------
;routine to window the data
window
                move    #win,r1                 ;start of window
                move    #1,n1
                move    #-1,m1
                move    #strt,r2                ;start of data
                move    #1,n2
                move    #-1,m2                  ;linear addressing
                do      #points,endwin
                move    x:(r2),x0               ;load data
                move    y:(r1)+,x1              ;load window
                mpy     x0,x1,a                 ;multiply
                move    a,x:(r2)+               ;save, increment data counter
                nop
endwin
                rts
;-----------------------------------------------------------------
;Routine to collect samples

samplf
                move    #strt,r1                ;point to data storage area
                move    #-1,m1                  ;use linear (mod 65536) addressing

                do      #points,end_sf          ;for i = 1 to number of points
low             jclr    #0,y:$ffff,low          ;Look for pulse edge
hi              jset    #0,y:$ffff,hi
                movep   y:$ffff,x:(r1)          ;Input from channel1 (signal)
                move    x:(r1),a
                rep     #5
                asr     a
                move    a,x:(r1)
                move    #0,y0
                move    y0,y:(r1)+
end_sf
                rts

;-----------------------------------------------------------------

four            ;A call to the fft macro
                fftr2a  points,data,coef
                rts

;-----------------------------------------------------------------
logd            ;find log of data squared
                move    #strt,r5
                move    #0,m5                   ;bit reversed
                move    #points/2,n5
                move    #win,r4
                move    #-1,m4
                move    #1,n4
                do      #points,endmov
                clr     a
                move    x:(r5),x0
                move    y:(r5)+n5,y0
                mac     x0,x0,a                 ;square
                mac     y0,y0,a                 ;square and add
                jsr     calc
                move    #0.6,x0
                add     x0,a
                rep     #6
                asr     a
                move    a,x:(r4)+
```

```
endmov
                clr     a
                move    #win,r4
                move    #strt,r5
                move    #-1,m5
                do      #points,lp_one
                move    x:(r4)+,x0
                move    x0,x:(r5)
                move    a0,y:(r5)+              ;zero fill imaginary
lp_one
                move    a0,x:(strt)
                rts


;-----------------------------------------------------------------
;Transmit to host routine

tx_host         movep   #0,x:$ffe
                move    #strt,r5                ;move base into r5
                move    #0,m5                   ;bit reversed
                move    #points/2,n5
                do      #points/2,end_tx        ;do points times
                clr     a
                move    x:(r5)+n5,x0

                movep   x0,x:$ffeb
                nop
wait            jclr    #1,x:$ffe9,wait         ;wait until clear
                nop
end_tx
                rts

;-----------------------------------------------------------------


calc            ;log calculations performed
;
; This program originally available on the Motorola DSP bulletin board.
; It is provided under a DISCLAMER OF WARRANTY available from
; Motorola DSP Operation, 6501 Wm. Cannon Drive W., Austin, Tx., 78735.
;
; Normalizing Base 2 Logarithm
; Modified by D.Birch sept 1992
;
;       This program calculates the base 2 logarithm of an unnormalized
;       24 bit fraction "x" in register A and returns a scaled fraction
;       "y" in register A.
;
;       y = log2(x)/32.0        where   2**(-23) =< x < 1.0
;                                       -23/32 =< y < 0.0
;
;       Note - "x" must be a non-zero, positive fraction.
;
;
;       Step 1 - Normalize A to get value between .5 and 1.0
;
        move    #-1,m1                  ;linear addressing
        move    m1,m7
        move    m7,r7                   ;initial count = -1
        rep     #23                     ;normalize to between .5 and 1.0
        norm    r7,a                    ;shift left and decrement r7 if needed
        move    #pcoef,r1               ;point to polynomial coefficients for log2
        move    a,x0                    ;put normalized number in x0
;
;       Step 2 - Calculate LOG2 by polynomial approximation.  8 Bit accuracy.
;
```

```
;       LOG2(x) = 4.0* (-.3372223 x*x + .9981958 x - .6626105)
;                         a2             a1            a0
;
;       r1 initially points to the coefficients in y memory in the
;       order: a1,a2,a0
;
        mpyr    x0,x0,a y:(r1)+,y0              ;x**2, get a1
        mpy     x0,y0,a a,x1 y:(r1)+,y0         ;a1*x, mv x**2, get a2
        mac     x1,y0,a y:(r1)+,y0              ;a2* x**2, get a0
        add     y0,a                            ;add in a0
        asl     a                               ;multiply by 4
        asl     a
;
;       Step 3 - Divide result by 32.
;
        asl     a                               ;shift out sign bit
        move    r7,a2                           ;new sign = characteristic
        rep     #6                              ;divide by 32, create sign bit
        asr     a
        rnd     a                               ;round result
        rts


;---------------end of program-------------------------------------
                end
```

G.5 Kurtosis Program

```
;This program was written in assembler
;It calculates the statistics of an array of 1024 points
;calcultions: the avg, (sum (x-avg)^2)/1024 and (sum (x-avg)^4)/1024

;program consists of apppx 70 instructions
;uses a,b,x,y as well as r0,n0,m0

points          equ     1024                    ;use a 1024 point block
start           equ     1024                    ;start at 1024 in x mem


                opt     fc,mu,s,w,mex
                org     p:0
                jmp     $100
                org     p:$100

;initialise
                andi    #%11111011,omr
                move    #0,x0
                move    x0,y:$fffd
                move    #$2209,x0
                move    x0,x:$fffe


main
                jsr     getsams                 ;sample the data
                jsr     calc                    ;perform calculations
                jsr     transm                  ;transmit the results
                jmp     main                    ;loop forever
;---------------------------------------------------------------
getsams         ;read data from the adc
                move    #start,r1
                move    #-1,m1

                do      #points,end_samp
low             jclr    #0,y:$ffff,low
high            jset    #0,y:$ffff,high
                movep   y:$ffff,x:(r1)+
end_samp
                rts


;---------------------------------------------------------------
calc            ;calculations performed in three loops
                ;calculating the average

                clr     a
                move    #start,r0
                move    #-1,m0
                move    #0.25,x0
                do      #points,endavg                  ;loop through
                move    x:(r0)+,x1                      ;mult by 0.25
                mac     x0,x1,a                         ;and add to total
endavg
                rep     #8
                asr     a                               ;devide by 256
                move    a,y0                            ;store in Y0
                move    y0,x:0                          ;and x:0

                ;calculate sum(x-avg)^2
                move    #start,r0
                clr     a
                do      #points,endsigsq
```

```
                move    x:(r0)+,b                       ;load b
                sub     y0,b                            ;subtract avg
                asr     b                               ;div 2
                move    b,x0                            ;move into x0
                mac     x0,x0,a                         ;square and add
endsigsq
                rep     #7                              ;div 128
                asr     a
                move    a1,x:1                          ;save in x:1
                move    a0,x:2                          ; and x:0

                ;calc sum(x-avg^4)
                move    #start,r0
                clr     a
                do      #points,endfour
                move    x:(r0)+,b                       ;load b
                sub     y0,b                            ;subtract avg
                asr     b                               ;div 2
                move    b,x0                            ;move into x0
                mpy     x0,x0,b                         ;square
                move    b,x0
                mac     x0,x0,a                         ;square and add
endfour
                rep     #6
                asr     a
                move    a1,x:3                          ;save sigma x^4
                move    a0,x:4                          ;as a 48 bit no.



                rts
;---------------------------------------------------------------

transm          ;transmit the  results to the host
                move    #0,r0                   ;data stored in x0..x4
                do      #5,endtrans
                move    x:(r0)+,x0
                movep   x0,x:$ffeb
wait            jclr    #1,x:$ffe9,wait
                nop
endtrans
                rts

;---------------------------------------------------------------
                end
```

G.6 LMS Program

```
;This program was written in assembler
;It performs an adaptive filter on block of data
;The filter coefficients are kept from block to block


points          equ     1024                    ;1024 points
start           equ     0                       ;start at 0 in x and y mem
filt            equ     1025                    ;start of filter coefficients
leng            equ     128                     ;filter length

                opt     fc,mu,s,w,mex
                org     p:0
                jmp     $100
                ;org    p:$100
                org     y:filt
                dc      0.01325,0.01555,0.01796,0.02042,0.02290 ;initial
                dc      0.02536,0.02776,0.03004,0.03218,0.03413 ;filter
                dc      0.03584,0.03730,0.03847,0.03931,0.03983 ;coefficients
                dc      0.95999
                dc      0.03983,0.03931,0.03847,0.03730,0.03584
                dc      0.03413,0.03218,0.03004,0.02776,0.02536
                dc      0.02290,0.02042,0.01796,0.01555,0.01325
                org     p:$100
;initialise
                andi    #%11111011,omr
                move    #0,x0
                move    x0,y:$fffd
                move    #$2209,x0
                move    x0,x:$fffe
                move    #-1,m0
                move    #-1,m1
                move    #-1,m2
                move    #filt,r0
                move    #-0.01000,x0
                do      #leng,fill
                move    x0,x:(r0)+
fill


main
                jsr     getsams         ;sample the data
                jsr     calc            ;perform addaptive filter
                jsr     calc            ;run through a second time
;               jsr     hpfilt          ;high pass filtering (option)
                move    #start,r0       ;transmit data
                move    #points,n0
                jsr     transm
                move    #filt,r0        ;transmit filter
                move    #leng,n0
                jsr     transm
                jmp     main
;---------------------------------------------------------------
getsams                                 ;read data from the adc

                move    #start,r1
                move    #-1,m0
                do      #points,rxloop  ;do 1024 times
low             jclr    #0,y:$ffff,low
hi              jset    #0,y:$ffff,hi
                movep   y:$ffff,x:(r1)
                move    x:(r1),a
                asr     a               ;devide by 4
```

```
                asr     a
                move    a,x:(r1)
                movep   y:$fffe,y:(r1)
                move    y:(r1),a
                asr     a
                asr     a
                move    a,y:(r1)+
                nop

rxloop
                rts


;---------------------------------------------------------------
calc                                            ;calculations performed
;
;
                move    #start,n0
                move    #filt,n1
                move    #@cvi(start+leng/2),n2  ;skip first 64 points
                do      #@cvi(points-leng),endblk       ;do 1024 -128 times
                move    n0,r0                           ;
                move    n1,r1                           ;
                move    n2,r2                           ;
                clr     a                               ;
                do      #leng,endfil    ;multiply       ;
                move    y:(r0)+,x0      ;filter         ;
                move    x:(r1)+,x1      ;times          ;
                mac     x0,x1,a         ;data           ;
endfil                                                  ;
                neg     a               ;subtract       ;
                move    x:(r2),x1       ;ref            ;
                add     x1,a            ;from data      ;
                move    a,x:(r2)+                       ;
                rep     #11                             ;
                asr     a                               ;
                move    a,y0                            ;
                move    n0,r0                           ;
                move    n1,r1                           ;
                do      #leng,endupdt   ;update         ;
                move    y:(r0)+,x0      ;filter         ;
                move    x:(r1),x1       ;coefficients   ;
                mpy     y0,x0,a         ;               ;
                add     x1,a            ;               ;
                move    a,x:(r1)+       ;               ;
endupdt                                                 ;
                move    n0,r0                           ;
                move    n2,r2                           ;
                move    (r0)+                           ;
                move    (r2)+                           ;
                move    r0,n0                           ;
                move    r2,n2                           ;
endblk                                                  ;end of filter
                rts

;---------------------------------------------------------------
hpfilt
                move    #start,r2
                do      #1024,outr
                move    r2,r0
                move    #filt,r1
                clr     a
                do      #31,innr
                move    y:(r1)+,x0
                move    x:(r0)+,x1
                mac     x0,x1,a
```

```
innr
                rep     #2
                asr     a
                move    a,y:(r2)+

outr
                rts
;---------------------------------------------------------------
transm                                  ;transmit the results to the host
                move    #-1,m0
                do      n0,endtrans
                movep   x:(r0)+,x:$ffeb
                nop
wait            jclr    #1,x:$ffe9,wait
                nop
endtrans

                rts

;---------------------------------------------------------------
                end
```

G.7 Power Spectrum Program

```
fftr2at         ident   1,1
                page    132,54
                opt     nomd,nomex,loc,cre,nocex,mu

                ; These two macros were taken fron shareware routines
                include 'sincos'        ;sine/cosine macro
                include 'fftr2a'        ;fft macro


strt            equ     0               ;start of data
win             equ     2048            ;start of window
points          equ     1024            ;number of points
data            equ     0
coef            equ     1024            ;sine cos table
pcoef           equ     3072            ;polynomial coefficients
                sincos  points,coef

                opt     mex
                org     p:0
                jmp     $100
                org     p:$100


;Initialize

                andi    #%11111011,omr
                move    #$0,x0                  ;Set up 100kHz Sampling rate
                move    x0,y:$fffd
                move    #$2209,x0
                move    x0,x:$fffe              ;set bus control register
                move    #0.9981958,x0           ;set up coefficients for
                move    x0,y:pcoef              ;the log routine
                move    #-0.3372223,x0
                move    x0,y:(pcoef+1)
                move    #-0.6626105,x0
                move    x0,y:(pcoef+2)
                jsr     rx_win                  ;load the window from the host




main            jsr     samplf                  ;sample the input
                jsr     window                  ;window the data
                jsr     four                    ;perform fourier transform
                jsr     tx_host                 ;transmit x data
                jmp     main                    ;loop forever


;.....................................................................
;routine to recieve a window from the host
rx_win
                movep   #0,x:$ffe8
                move    #win,r1
                move    #1,n1
                move    #-1,m1
                do      #points,end_wrx
wn_rx           jclr    #0,x:$ffe9,wn_rx
                movep   x:$ffeb,y:(r1)+
                nop
end_wrx
                rts
```

```
;-------------------------------------------------------------
;routine to window the data
window
                move    #win,r1
                move    #1,n1
                move    #-1,m1
                move    #strt,r2
                move    #1,n2
                move    #-1,m2
                do      #points,endwin
                move    x:(r2),x0
                move    y:(r1)+,x1
                mpy     x0,x1,a
                move    a,x:(r2)+
                nop
endwin
                rts
;-------------------------------------------------------------
;Routine to collect samples

samplf
                move    #strt,r1                ;point to data storage area
                move    #-1,m1                  ;use linear (mod 65536) addressing

                do      #points,end_sf          ;for i = 1 to number of points
low             jclr    #0,y:$ffff,low          ;Look for pulse edge
hi              jset    #0,y:$ffff,hi
                movep   y:$ffff,x:(r1)          ;Input from channel1 (signal)
                move    x:(r1),a
                rep     #5
                asr     a
                move    a,x:(r1)
                ;movep  y:$fffe,y:(r1)+         ;Point to next data location
                move    #0,y0
                move    y0,y:(r1)+
end_sf
                rts

;-------------------------------------------------------------

four            ; a call to the FFT macro
                fftr2a  points,data,coef
                rts

;-------------------------------------------------------------

;Transmit to host routine

tx_host         movep   #0,x:$ffe8
                move    #strt,r5                ;move base into r5
                move    #0,m5                   ;bit reverse addressing
                move    #points/2,n5
                do      #points/2,end_tx        ;do points times
                clr     a
                move    x:(r5),x0
                move    y:(r5)+n5,y0
                mac     x0,x0,a                 ;square
                mac     y0,y0,a                 ;square
                jsr     calc                    ;get log of data
                movep   a,x:$ffeb
                nop
wait            jclr    #1,x:$ffe9,wait         ;wait until clear
                nop
end_tx
                rts
```

```
;---------------------------------------------------------------


calc            ;log calculations performed
;
; This program originally available on the Motorola DSP bulletin board.
; It is provided under a DISCLAMER OF WARRANTY available from
; Motorola DSP Operation, 6501 Wm. Cannon Drive W., Austin, Tx., 78735.
;
; Normalizing Base 2 Logarithm
; Modified by D.Birch sept 1992
;
;       This program calculates the base 2 logarithm of an unnormalized
;       24 bit fraction "x" in register A and returns a scaled fraction
;       "y" in register A.
;
;       y = log2(x)/32.0        where   2**(-23) =< x < 1.0
;                                       -23/32 =< y < 0.0
;
;       Note - "x" must be a non-zero, positive fraction.
;
;
;       Step 1 - Normalize A to get value between .5 and 1.0
;
        move    #-1,m1          ;linear addressing
        move    m1,m7
        move    m7,r7           ;initial count = -1
        rep     #23             ;normalize to between .5 and 1.0
        norm    r7,a            ;shift left and decrement r7 if needed
        move    #pcoef,r1       ;point to polynomial coefficients for log2
        move    a,x0            ;put normalized number in x0
;
;       Step 2 - Calculate LOG2 by polynomial approximation.  8 Bit accuracy.
;
;       LOG2(x) = 4.0* (-.3372223 x*x + .9981958 x - .6626105)
;                         a2            a1            a0
;
;       r1 initially points to the coefficients in y memory in the
;       order: a1,a2,a0
;
        mpyr    x0,x0,a y:(r1)+,y0              ;x**2, get a1
        mpy     x0,y0,a a,x1 y:(r1)+,y0         ;a1*x, mv x**2, get a2
        mac     x1,y0,a y:(r1)+,y0              ;a2* x**2, get a0
        add     y0,a                            ;add in a0
        asl     a                               ;multiply by 4
        asl     a
;
;       Step 3 - Divide result by 32.
;
        asl     a                       ;shift out sign bit
        move    r7,a2                   ;new sign = characteristic
        rep     #6                      ;divide by 32, create sign bit
        asr     a
        rnd     a                       ;round result
        rts


;---------------end of program-----------------------------------
                end
```

Appendix H

The DSP Shell Program

H.1 Structure and use of the shell program

The shell program consists of a number of routines stored in several files. The main file is SHELL.C. This file contains global variable declarations, the main procedure and other important routines. The rest of the functions are contained in a series of include files:

- MINC.C This file contains the mouse routines and is included into the file GRAPHRU.C.
- GRAPHRU.C This file provides basic graphics functions for initialising the graphics and drawing graphs and axes.
- GSPLOT.C The routines for plotting in grey shades are provided in this file, as well as the routines for switching the palette to colour or grey shades.
- MENUINC.C Basic menu definitions and functions are defined in this file. Provision is made for users to add their own functions to the menu, as detailed in section H.2.
- LASDUMP.C Screen dumps to a laserjet or deskjet type printer are performed by the dump routine.

H.1.1 Using the Mouse Routines

The mouse routines provide a simple interface to the mouse. For more complex mouse functions consult the Genius Mouse documentation. There are seven mouse functions, defined as follows:

- void showmouse();
- void hidemouse();
- void getmouse(int *x,int *y,int *button);

- void getpresstate(int *x,int *y,int *button,int *status);
- void mousepage(int page);
- int resetmouse(int *buttons);

The resetmouse() function should be called at the beginning of the program to return the mouse to its default settings. The function will return TRUE if the reset was successful, and buttons will contain the number of buttons the mouse has (2 or 3). Once the mouse has been reset, calls to showmouse() and hidemouse() will display or hide the mouse cursor. Note that hidemouse() and showmouse modify counters, thus if hidemouse() is called more than once, showmouse() must be called an equal number of times for the cursor to be visible.

The getmouse() and getpresstate() functions can be used to obtain information on the mouse position and which buttons have been pressed. Getmouse() returns information about the current mouse state, x and y will contain the current cursor position and the lower three bits of the parameter but will be set according to which buttons are pressed : Bit 2 for left button, bit 1 for middle button and bit 0 for the right button.

Getpresstate() provides information on the history of the mouse. The x and y parameters tell the cursor position when a button was last released, and the times variable tells how many times a button has been pressed since the last call to this function. The button variable functions as in getpresstate() indicating which buttons are pressed.

The mousepage() function sets which of the video pages the cursor will be displayed upon. This is important when using page swapping, as the cursor should always be on the visible page.

H.1.2 The Graphics Routines

The graphics functions are defined as follows:

- void initi(void);
- void drawaxes(void);
- void displayp(long *points,long scale,short offset,short colour);
- void displaylin(long *points,long scale,short offset,short colour);

These functions are used in the graph type displays. Initi() is called at the beginning of the program, to initialise the VGA card to 640 by 350 mode. Drawaxes() is used to clear the graph area and draw new axes. Displayp and displaylin perform the same function of drawing the data as a graph. Displayp plots the data as points, and displaylin joins the points with lines.

H.1.3 Colour Control and Grey Shade Plotting

- void grayshade(void);
- void normcol(void);
- void displaygs(long *data,long scale,long offset);

The function grayshade() sets the vga palette to a grey shade palette. A colour palette can be restored by calling normcol(). Once the grey scale palette has been selected, plots can be made in 16 shades of grey using dispalygs(). Dispalygs() plots 350 points in a vertical line on the screen. Subsequent calls to displaygs() use adjacent lines, until 400 lines have been plotted, then plotting resumes at the first position. The colour to be plotted is calculated according to the formula $colour = \frac{data}{scale} - offset$ No checking is performed ont he data to ensure that it is in the range 0 to 15, thus it is up to the user to ensure that this formula will yield a valid colour for the given data.

H.1.4 The Menu system

In order to implement the menu system, a structure menuelem was defined. This consists of an array of characters, and a function pointer. A global, two dimensional $N \times 8$ array of these is declared in the program. The character string in each menu is initialised to be the string that will be printed on the screen for each menu option. The accompanying function pointer is set to the function that will be called when that option is selected. For simplicity, all functions return void and take an integer parameter, which is the number of the menu option that was selected. A global variable currentmenu contains the number of the current menu. When the mouse is polled and found to have been clicked on the third option in the current menu, then the following function call is made:

```
menuarray[currentmenu][3].func(3);
```

which will call the appropriate function.

Options can be added to menus by setting the string and function fields of the menu element, and providing the function to be called. A new menu can also be displayed by setting the currentmenu to be equal to the new menu, then displaying the new menu using the function drawmenu(menunumber). A function is provided that will use a lookup table to find the previous menu. This is useful when using a RETURN option in a menu. The lookup table in this function should be modified if new menus are added.

H.1.5 The Screen Dump Program

The screen dump program is a single function. As a parameter it takes a pointer to the output stream. The screen is scanned and output written to the stream in the form of HPPl raster

graphics. The routine uses error diffusion to obtain grey shades, with four printer pixels to every screen pixel. The printer is set in 300dpi mode , but the code can be modified to work with the lower resolution modes, and thus obtain a larger print size. For black and white printing a second function is provided, that will print the screen with all screen pixels not printed in colour 0, the background colour printed as black on the printer.

H.2 Modifying the Program to Include other Routines

The main body of the program is a continuous loop:

```
Start of loop
Check mouse and perform menu function
perform signal processing procedure
Goto the start of the loop
```

The signal processing function is called by using a function pointer. To include a new signal processing procedure comprises three main steps.

1. Write a function to read the data from the dsp card and display it:

   ```
   void spectrum(void)
   {
   long data[400];
   dspreadlong(data,400);
   displaylin(data,600000,400,2);
   }
   ```

2. Write the startup code and include it in function start. This will usually consist of code to download the dsp program and set up the menu to be displayed while the code is running.

3. Modify the main menu to include an option for running the new routine.

Bibliography

[1] R.L. Dunn. Advanced Maintenance Technologies. *Plant Engineering*, June 1987.

[2] N. Tandon and B.C.Nakra. The application of sound-intensity techniques to defect detection in rolling element bearings. *Applied Acoustics*, 29, 1990.

[3] Eschmann, Hasbargen, and Weigand. *Ball and Roller Bearings: Their Theory, Design and Application.* R Oldenbourg, Munich, 1958.

[4] Ram Cloete. Analysis of Rolling Element Bearing Vibration using an FFT Analyser. In *VISA Machine Vibration Monitoring Conference*, May 1990.

[5] G. White. Amplitude demodulation - a new tool for predictive maintenance. *Sound and Vibration*, September 1991.

[6] James E. Berry. How to Track Rolling Element Bearing Health with Vibration Signature Analysis. *Sound and Vibration*, November 1991.

[7] A.F. Khan and E.J. Williams. Predicting the remaining life of rolling element bearings. In *Institute of mechanical engineers international conference on Vibrations in rotaing machinery*, 1992.

[8] R.H Bannister. A Review of Rolling Element Bearing Monitoring Techniques. In *Condition Monitoring of Machinery and Plant*, June 1985.

[9] R.B. Noyes. The Damage Potential of Machine Vibrations. In *Machine Vibration monitoring conference*, 1990. Mechanical Engineering Department, UCT.

[10] J. Swartz. Active vibration cancellation. U.C.T, 1989.

[11] D. Rupert. What is kurtosis? *The American Statistician*, 41, 1987.

[12] D.Dyer and R.M.Stewart. Detection of Rolling Element Bearing Damage by Statistical Vibration Analysis. *Journal of Mechanical Design*, April 1978.

[13] P.V. Bruel. Spike Energy. Technical Report 2850, Bruel and Kjaer, 1970.

[14] Acoustic Technology Ltd. Machinery Condition Monitoring - Vibration Diagnostic Techniques: part2. *Noise and Vibration, Worldwide*, June 1990.

[15] Irwin Miller and John E. Freund. *Probability and Statistics for Engineers.* Prentice-Hall, 3 edition, 1985.

[16] James E. Berry. How to Specify Machinery Spectral Alarm Bands. *Sound and Vibration*, September 1990.

[17] Ray l. Martin. Detection of Ball Bearing Malfunctions. *Instruments and Control Systems*, pages 79,82, December 1970.

[18] Ronald L. Eshleman. The use of Sum and Difference Frequencies in Rotating Machine Analysis. In *Machinery Vibrations IV Seminar*, November 1980.

[19] P.G. Vaidya and M.J. Anderson. Use of the Trans-Spectral-Coherence Technique to Separate Signals from Noise. *Journal of the Acoustical Society of America*, 89(5):2370 to 2378, May 1991.

[20] Crysotomos L. Nikias and Mysore R. Raghuveer. Bispectrum estimation: A digital signal processing Framework. *Proceedings of the IEEE*, May 1987.

[21] R.M.Stewart. Application of Signal Processing Techniques to Machinery Health Monitoring. In R.G.White, J.G.Walker, and Ellis Horwood, editors, *Noise and Vibration*, chapter 23. unknown, 1982.

[22] Hewlett Packard. *Effective Machinery Measurements using Dynamic Signal Analysers*, 1990. Application note 243-1.

[23] G.K.Chaturvedi and D.W.Thomas. Bearing Fault Detection Using Adaptive Noise Cancelling. *Transactions of the ASME*, April 1982.

[24] P. Bremer and A. Jongens. Development of a digital signal processing package for use in machine condition monitoring. Central Acoustics Laboratory UCT.

[25] P. G. Bremer. Adaptive Noise Cancelling Applied to Machine Condition Monitoring. Master's thesis, U.C.T., 1990.

[26] M.C. White, P.G. Bremer, and A.W.D. Jongens. The application of adaptive noise cancelling to machine condition monitoring. In *Second Machine Condition Monitoring Conference*, May 1991.

[27] B. Leggat. The Development of a Rule Based Expert System to Automate the Digital Analysis of Condition Monitoring Parameters Captured on Rolling Element Bearings Subjected to Simulated Failure. Master's thesis, U.C.T, 1990.

[28] *SKF General Catalogue*, 4000 e edition, 1989.

[29] C. Cempel. Passive diagnostics and reliability experiment: Application in machine condition monitoring. *Transactions of the ASME*, 111, January 1989.

[30] Richard P. Lippmann. An Introduction to computing with Neural Nets. *IEEE ASSP magazine*, April 1987.

[31] K. Knight. Connectionist Ideas and Algorithms. *Communications of the ACM*, November 1990.

[32] M. Levy. DSP Toolbox for Machine Condition Monitoring. Undegradute Thesis, Electrical Engineering, November 1990.

[33] P. Horowitz and W. Hill. *The Art of Electronics.* Cambridge University Press, 1980.

[34] Motorola Semicomductor. DSP56ADC16 16-bit sigma-delta analog-to-digital converter. Advance Data DSP56ADC16D, Motorola Inc., 1989.

[35] F. Goodenough. Grab distributed sensor data with 16–bit delta-sigma ADCs. *Electronic Design*, page 49..56, April 1988.

[36] C.S. Burrus and T.W. Parks. *DFT / FFT and Convolution Algorithms Theory and Implementation.* John Wiley and Sons, 1984.

[37] A.E.H. Love. *A Treatise on the Mathematical Theory of Elasticity.* Cambridge University Press, 1934.