# A PROTOTYPE SYSTEM FOR GRAVITATIONAL WAVE DATA ANALYSIS

by

**W.J.Watkins**

A thesis submitted to the
University of Wales
for the degree of
Doctor of Philosophy

**September, 1991**

This thesis is dedicated to my Mother and the memory of my Father.

## DECLARATION

I declare that this work has not already been accepted in any substance for any degree, and is not being currently submitted in candidature for any degree.

W. J. Watkins

Candidate

Except where otherwise stated, this work is wholly the result of the candidate's own investigation. Suitable credit is given to joint work with colleagues, and to work of others throughout the thesis.

W. J. Watkins

Candidate

[illegible]

Supervisor

## ACKNOWLEDGEMENTS

I would like to thank my supervisor, Professor B.F. Schutz, for his constant encouragement, guidance and patience.

I should also like to thank the Gravitational Physics group at the University of Glasgow, headed by Professor J. Hough, who supplied the data analysed in this thesis.

I would like to acknowledge the help and encouragement of members of the Department of Physics and Astronomy at Cardiff, M.Alcubierre, G.D.Allen, M.Antonioletti, T.Barnett, S.J.Chapman, J.I.Davies, J.R.Davies, C.A.S.Dickson, R.H.Evans, R.Greenhow, J B.Jones, G.S.Jones, V.Kluckers, S.P.A.Lawrence, J.S.Panesar, H.Pongracic, M.Shaw, J.R.Shuttleworth, R.M.Smith and in particular D.Nicholson.

## Abstract

This thesis is concerned with a prototype gravitational wave data analysis system. Work will shortly be underway on the construction of large scale broad-band laser interferometers that should produce data containing gravitational waves. Subsequently, systems that can analyse this data, extracting any waves within the noise, need to be developed now. The prototype system described in this thesis is the first fully automated gravitational wave data analysis system to be developed. It is designed in parallel, to run on several processors simultaneously, as future systems are certain to employ parallelism to some degree. This thesis is also concerned with the analysis of actual data by this system, produced by the prototype interferometer owned and operated by the University of Glasgow.

The software is described in detail, beginning with a description of the parallelism within the system. The procedure employed to split the multiplexed data into its constituent streams is discussed followed by a description of the methods employed to optimise the detector noise for the extraction of possible signals from short duration gravitational wave bursts. This is followed by a description of both the statistical analysis and the search for events carried out on the calibrated and optimally weighted interferometer stream. This search looks for wideband burst events in the time series and also coalescing binary signals found by cross-correlating the noise with a bank of templates.

An account of the results of the analysis is given, concentrating on the statistics of detector noise, identifying its main characteristics both when the detector is operating properly and also when the housekeeping data suggests it is not. A description is given of a parameter, easily calculated from the noise, that is shown to be a good diagnostic indicator of the state of the detector. The distribution of events found by the analysis is discussed, and it is shown how the number of events vary with the state of the detector.

# Contents

# Chapter 1

# INTRODUCTION.

### 1.0.1 Gravitation

It is necessary to know only that a body has mass to also know that it exerts a gravitational force on every other body possessed of mass in the universe. This was first realised by Isaac Newton who developed the concept of gravitational attraction from the experiments of Galileo and Kepler's 3 kinematic laws of planetary motion. He arrived at the following expression for the force of attraction due to gravity, $F_{12}$, between two masses, $M_1$ and $M_2$ separated by a distance $d$,

$$F_{12} = \frac{GM_1M_2}{d^2}, \tag{1.1}$$

where $G$ is the *Gravitational constant* which has the value $6.673 \times 10^{-11}\mathrm{Nm}^2/kg^2$. This equation appeared in Newton's *Principia* (1687), which effectively defined man's understanding of gravitation for well over two centuries. It remains to this day accurate enough for almost all practical purposes on Earth as well as in the solar system.

Thinking changed at the start of this century when Albert Einstein formulated the theories of Special and General Relativity (1905,1915). Einstein utilised the mathematics of curved spaces constructed by Riemann (1854), to produce a mathematical model of space and time. He represented gravity in *tensor* form not as a force or a field but as a *spacetime* geometry, and derived the *field equations* that described spacetime in the region around gravitating bodies be they moving or stationary.

### 1.0.2 Gravitational Waves

#### History

Gravitational wave research started in 1916, when Einstein began to investigate whether solutions to his field equations predicted the propagation of energy in the form of waves in spacetime, analogous to solutions of Maxwell's equations predicting electromagnetic waves. Einstein found a wave equation that suggested that they did. In the following years, Einstein himself (1918), Weyl (1922) and Eddington (1924) further developed this work, until the linearised theory for gravitational waves from non self-gravitating systems was thought to be fully understood.

The first major work on emission from non-linear, self-gravitating systems was produced by Landau and Lifshitz (1941). Following this physicists such as Bondi (1957,1960), Penrose (1963) and Isaacson (1968) continued the work on the subject steadily until the early 70's since when, motivated by the considerable activity surrounding detector research, an ever increasing number of theorists have striven to develop models for sources of gravitational radiation and the associated waveforms that detectors may discover. Consequently, many potential sources have been identified and analysis techniques developed to extract their characteristic wave forms from detector noise.

## The nature of gravitational waves

Gravitational waves are a consequence of any relativistic theory of gravity. They are ripples in the existing curvature of spacetime that propagate with the speed of light. Within the most widely accepted of these theories,*ie* the General Theory of Relativity, they are described by $h^{\alpha\beta}$, the weak field deviation of the spacetime metric from flat spacetime. $h^{\alpha\beta}$ is given by the solutions of Einstein's field equations in a vacuum. These are all contained within,

$$\left(-\frac{\partial^2}{\partial t^2} + \nabla^2\right) \bar{h}^{\alpha\beta} = 0. \tag{1.2}$$

The solutions to the above are well documented in many texts such as Schutz (1985) and Misner, Thorne and Wheeler (1973) so they are not included here.

Gravitational waves are produced only by non-symmetric motions. They are emitted in all directions, to varying degrees dependent upon the motions of the system producing them. Gravitational waves influence matter in directions perpendicular to their direction of propagation, altering the separation of particles according to the waves' two distinct linear polarisation states, represented by $h_+$ and $h_\times$. These, in contrast to the polarisation states for electromagnetic waves which are separated by 90°, are rotated by only 45° relative to each other. The differing physical effects of these states on a ring of free particles is shown in Figure 1.1. The top row shows the deformation of the ring for various phases for the wave cycle of the $h_+$ polarisation of a gravitational wave. The bottom row is the equivalent representation for the $h_\times$ polarisation. Notice that the axes of deformation are at 45° to each other.

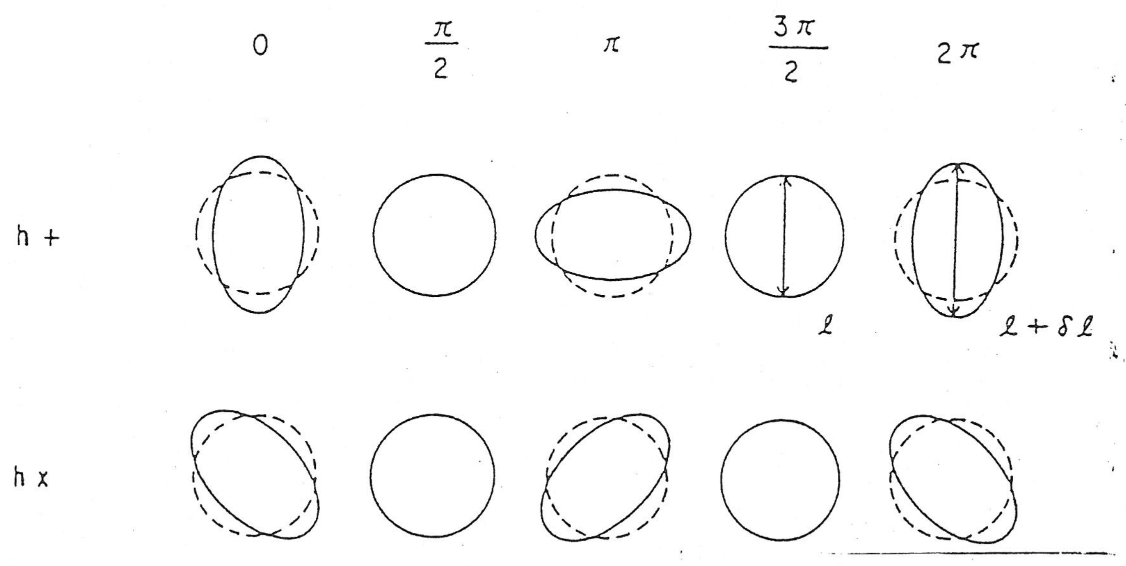


Figure 1.1: Elastic ring deformed by both polarisation states separately. The strain in space is $\delta l/l = 1/2h$. Diagram taken from Kawamura *et al*(1989)

## Sources of gravitational waves

Any two particles moving non-symmetrically in some way relative to each other should emit gravitational waves. For these waves to be of sufficient magnitude to be even potentially detectable on Earth however, necessitates very rapid non-symmetrical motions in very massive bodies.

These sources could produce waves with a number of fundamental characteristics. The most important in terms of this thesis is the short duration *burst* signal, whether it is spread over a wide frequency range, e.g. stellar core collapse, or whether it is in some way frequency specific, confined to some small frequency range, e.g. coalescing binary system where the components are neutron stars and/or black holes. Observation of one such system, the binary pulsar PSR 1913+16 discovered by Hulse and Taylor in 1974, has already provided indirect evidence of the existence of gravitational waves. Its observed rate of energy loss due to orbital decay is correctly predicted in terms of energy loss due to emission of gravitational waves, and in fact it will coalesce in a time less than the Hubble time (Weisberg and Taylor(1984), Taylor(1987)).

The waves could also be *continuous*, and produce steady signals over periods of time long enough for the signal to vary significantly because of the rotation of the Earth. This effect is known as *frequency and amplitude modulation*. The best known prospective candidates for the production of continuous signals are spinning neutron stars or pulsars. A third detectable source could be the *stochastic* background of gravitational radiation whose various contributors could include cosmic strings and events in the early universe. This thesis will not be concerned with the detection of such wave. For continuous waves, see the thesis of Livas(1987).

**Motivation behind gravitational wave research.**

The development and construction of a gravitational wave detector is an extremely large undertaking, requiring much money and manpower. In order to justify such a commitment, a strong scientific case had to be put, listing the benefits of the detection of gravitational waves. These benefits are many, both astrophysical and technical.

The most immediate consequence would be the potential detection of the waves themselves. Gravitational waves are the last major prediction of Einstein's theory yet to be independently confirmed by experimentation. Their detection could further strengthen General Relativity in relation to other gravitational wave predicting theories of gravity, if the information received from the wave, in terms of the amplitude, direction and polarisation are related in a way consistent with General Relativity. Of course if the arrival of gravitational waves could be associated with an optical event, such as the brightening of a supernova, then the delay between the two observations could be used to confirm, or otherwise, that gravitational waves also travel at the speed of light.

The real benefits to astrophysics, however would come later with the vast new window that gravitational waves would open on the universe. As welcome as radio and x-ray astronomy were as additions to optical astronomy, their information content is limited in that they are all specifically produced by individual ions, atoms and molecules in various states of excitement, frequently in sparse, low velocity regions, whose emissions can be easily blocked by matter between Earth and the source. Gravitational wave astronomy will be radically different, their sources specifically being dense, massive and necessarily subject to high velocities. Gravitational waves pass through matter essentially unaffected. Hence they could enable investigation of regions *never* seen directly before.

Specifically, detection of supernovae by gravitational wave detectors could lead to a greater understanding of core collapse, providing statistics showing the frequency and distribution of local core collapses, the ratio of the production of black hole to neutron stars and also the frequency of core collapses not producing visible supernovae. Additionally we may be able to imply the extent of asymmetry in the collapse as well as the collapse and rebound timescales, the mass and angular velocity of the neutron star (if produced), and possibly even be able to put an upper limit on the masses of neutron stars. One of the major consequences of the detection of coalescing binaries,

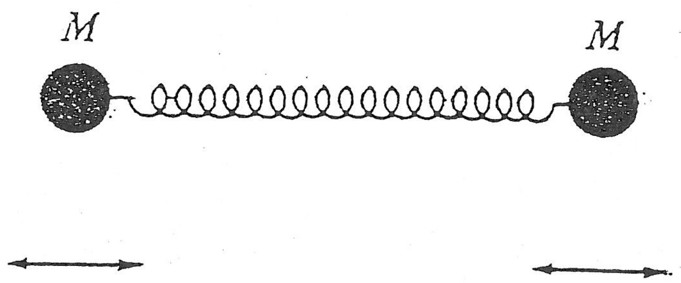


Figure 1.2: Masses oscillating due to a gravitational wave. Diagram from Davies(1980)

apart again from the obvious distribution statistics and component mass information, is that their distances can be quite easily calculated. Schutz (1986b) has shown that this could lead to an accurate determination of the Hubble constant which in turn would determine the age of the Universe as well establishing a distance scale to galaxies external to our Local Group. Continuous sources and stochastic background radiation would also supply astrophysical information, concerning mass distribution and the early universe, which could influence theories on galaxy formation. Of course it's entirely possible that there may be gravitational waves detected from unexpected or indeed completely unknown sources.

There are of course implications outside of astrophysics to gravitational wave research. Nuclear physics could benefit from studies of neutrons in neutron stars, under conditions not obtainable on Earth. Also there are many potential technological benefits from detector development in such areas as laser technology, vacuum systems and seismic isolation systems.

### 1.0.3 Detectors

Weber (1960) showed that a mass quadrupole harmonic oscillator would be excited by a gravitational wave passing through it. The simplest form of this is just two masses joined by a spring, as shown in Figure 1.2. The impinging wave would alter the separation of the masses against the action of the spring, setting up oscillations within the system of the form of those produced by a forced, damped harmonic oscillator. This provides the physical basis for the two main gravitational wave detector designs. In *bar* detectors the masses are represented by a single solid bar of material such as aluminium, which also supplies the spring, in the form of the material's elasticity. In *laser inter-*

*ferometers* the masses are supported as pendula with the opposing force supplied by the component of $g$, the acceleration due to gravity along the direction of the pendulum arc.

### Bar detectors

The first serious effort made towards developing a detector for gravitational radiation was by Professor J.Weber at the University of Maryland in the late 50's. Weber's bar detectors consisted of a cylindrical bar of aluminium a few tons in weight and a few meters in length, freely suspended from some fixed frame. The component of the gravity wave passing through the bar, perpendicular to its length, would cause the ends of the bar to displace relative to each other, the amplitude of which depends upon the amplitude of the wave and the inter-molecular forces within the bar. These displacements are converted into electrical signals by sensors, amplified and displayed on a monitor. Burst events would pass through the bar in less then 1ms, but using certain suitable materials for the bar such as niobium or aluminium, a very large damping time is possible, maintaining vibrations in the bar long after the wave has passed, improving the chances of detection. The wave passing through the bar would excite several different modes of vibration in the bar. If the bar is tuned so that one of these modes is the same as the bar's fundamental mode of vibration then large excitement could be obtained. Most bars in operation today are tuned to a fundamental frequency of about 1kHz and have a very narrow bandwidth. This optimises them for the detection of core collapses and also makes data analysis much less troublesome.

There are three types of bar in operation today,

- Room temperature bars such as the one at Maryland, these have a strain sensitivity between $10^{-16}$ and $10^{-17}$. These detectors are limited by thermal noise in the material of the bar.
- The thermal noise problem is reduced in cryogenic bars where the bar is cooled to only a few degrees K, achieving thus far, sensitivity of about $10^{-18}$ . Examples of this type of detector may be found in Rome, Perth, Stanford and Louisiana State University.
- The third type of detector is the so called Torsion Pendulum, which consists of two large masses connected by a torsion fiber. These detectors are as sensitive as other bars but are tuned to a continuous source, *eg* the pulsars, rather than looking for burst events. They gain sensitivity for continuous sources as,

$$\text{sensitivity} \propto [\text{observing time}]^{1/2}.$$

A detector of this type has been constructed in Tokyo specifically to look for radiation from the Crab pulsar (Owa *et al*(1986)) .

With a best sensitivity of around $10^{-18}$, improvements of at least 3 orders of magnitude are necessary, for the detectors to become useful. Current bars however, could potentially detect supernova in our galaxy and they remain our best hope of a detection in the next few years at least. The way towards increased sensitivity for bars is via ultra cryogenic bars now being developed at temperatures below 100mK or less.

Ultimately bar sensitivity must go below the Standard Quantum Limit to reach $10^{-21}$. This limit is reached when the energy of vibration of the bar is exactly 1 phonon or $h\omega$. Caves(1980) showed that *squeezing* could allow this limit to be exceeded. If M is the mass of bar, $\omega$ the angular frequency of the bars mode, $h$ Planck's constant and $X_1$ and $X_2$ are a pair of quantum-mechanically conjugate observables, then the uncertainty principle confines the accuracy to which $X_1$ and $X_2$ are measured, by

$$\Delta X_1 \Delta X_2 \geq \hbar/(2M\omega) \tag{1.3}$$
$$\text{where}, \hbar = h/(2\pi). \tag{1.4}$$

It is then possible in principle, by squeezing, to reduce the uncertainty $\Delta X_1$ in $X_1$ at the expense of $X_2$ and then use $X_1$ as a measure of the gravitational wave amplitude. This could eventually allow strain sensitivities below $10^{-20}$, but thus far no practical implementation of squeezing for bars has been demonstrated because the technical problems seem formidable. For an up to date discussion see Michelson,P.F.*et al* (1991). Most recent attention has been focused on interferometric detectors.

**Laser interferometric detectors**

Although Weber, in collaboration with Forward (see Forward(1971)) originally suggested the use of laser interferometers to detect gravity waves in the early 60's, it was not until the early 70's that research and development began in earnest.

The basic design is similar to that of the Michelson interferometer, with 3 freely suspended masses, in the form of 2 orthogonal arms of equal length with a beam splitter attached to the mass at the intersection and mirrors attached to the masses at the ends. The beam splitter transmits equal intensities of light along both arms, the reflected light from both is then recombined at the splitter and transmitted to the photo diode.

When a gravity wave is incident upon the detector the relative length of the arms change, which produces a shift in the interference fringe from which the difference in lengths, $\delta l$, of the arms can be inferred.

The change $\delta l$ is dependent on the length of the arms, and a linear combination, $h$, of the amplitudes of the polarisation states, $h_+$ and $h_\times$,

$$\delta l = l_0 h \tag{1.5}$$

where $l_0$ is the stationary arm length.

To get an idea of the magnitude of the problem faced in detecting $\delta l$ consider a detector with arm length 10m (as in the case of the Glasgow prototype ), and a wave of amplitude $10^{-21}$ then, $\delta l = 10^{-20}$m, which is 5 orders of magnitude smaller than the *classical* radius of an electron.

To make measurements of this sensitivity possible, modern interferometers will have long arm lengths (3-4km) which is effectively increased by keeping the light in the arms for many reflections. Light power is built up further by power recycling (Drever 1983), while narrow banding may be achieved by signal recycling (Meers 1988), demonstrated in the lab by Strain and Meers (1991).

There are several working prototype interferometers in operation at the moment, in Glasgow, Munich, Caltech and Japan, with all but the Japanese prototype giving sensitivity of order $10^{-18}$ comparable with the most sensitive cryogenic bars. Many long armed detectors are currently planned by these groups and others around the world, particularly the VIRGO collaboration between Italy and France. They should eventually achieve sensitivities of $10^{-22}$ or even better.

**"Space Based Detectors"**

Currently prospective gravitational wave detection in space amounts only to tracking interplanetary spacecraft. The future however, could see space based Laser Gravitational Wave Observatories, consisting of triangular arrangements of 3 spacecraft , separated by $10^7$km. This huge arm length should enable the detection of waves with comparable wavelengths with subsequently far lower frequencies than any detectable on Earth, *ie* in the range $10^{-5} - 10^{-2}$Hz. This is made possible by the lack of low frequency ground vibration, that Earth based detectors are unable to escape. This could enable the detection of very many white dwarf binaries as well as long period massive binary systems long before coalescence.

### 1.0.4 Data analysis

One of the problems associated with an operating broadband gravitational wave detector, such as a laser interferometers, is the quantity of data produced. To be effective in the required frequency range requires a sampling rate of at least 10000Hz for 2 Byte data, which amounts to of order $10^{12}$ Bytes per year when in continual operation, without even consideration of any addition diagnostic or environmental data that may also be taken. This quantity of data, regardless of the method of storage, would amass very rapidly if its analysis could not keep up with the data taking rate. Hence the development of an automatic analysis system that continually analyses the data, online, immediately reporting certain types of events such as coalescing binaries and supernovae explosions.

Immediately reporting supernovae would be particularly important as they may well be apparent to gravitational wave detectors before any other observatories, giving optical and other types of astronomers the chance to observe one actually brightening, as opposed to studying it only after the event as has been the case up to now. Any system should also be conducting pulsar searches and be able to cross-correlate results with those obtained by other detectors as well as cross-correlating actual data from several detectors looking for the stochastic background as well as new unpredicted sources, although this beyond the scope of this thesis.

### 1.0.5 Contents of the thesis

The work described in thesis is the first attempt to build an automated data analysis system, which was tested on actual data taken by a prototype gravitational wave detector. This data was taken by the Glasgow prototype detector during a 100 hour data taking run, coincident with a similar run on another prototype in Germany, in March 1989. The data from the Glasgow run was recorded on video cassettes by an *Exabyte* device. The analysis software was designed in parallel, splitting the analysis up into 5 distinct parts each of which was carried out simultaneously, on 5 distinct processors. The decision to make the system parallel was taken mainly because the final system, having to deal with large amounts of data very rapidly, would probably also have to be parallel in some way, hence future parallel designs could benefit from experience gained now.

The scope of the analysis was obviously limited by the processors used, the *transputers*, both in terms of their processing speeds and their memories. A system could have been developed that analysed the data as completely as wanted, searching for burst events- reconstructing many hundreds of filters for each new stretch of data- as well as passing large amounts of the data continuously back and forth to the hard disk of the PC for the pulsar searches and even auto-correlating large stretches of data, searching for stochastic background and new sources. Such a system would be ungainly, complex and very very slow but perhaps justifiable if there were any real chance of the data containing any signals. The sensitivity of the detector is such, however that this is almost certainly not the case. For this reason therefore, more than any other, the routines were designed primarily, to be efficient in terms of time, searching only for time series bursts and coalescing binaries, crosscorrelating the data with only as many filters as can be used without slowing the system, the main output being a statistical picture of both the noise and the environmental *housekeeping* data as well as providing false alarm statistics.

Briefly then the process read the data into the transputer network for analysis. The data was composed of the noise potentially containing the gravitational wave signal and also streams of housekeeping data. The noise was firstly optimally filtered, before any event searches were carried out. The time series was searched for wide frequency bursts as well as being cross-correlated with a suite of *filters* based on the predicted characteristic shape of the signal from coalescing binaries. An event search was also carried out on this data. The results of the analysis were written to another video cassette.

## Summary of Chapters

Chapter 2 describes in more detail the sources mentioned above, concentrating on core collapse and coalescing binaries providing mathematics necessary for the analysis.

Chapter 3 is in two separate parts. The first gives an outline of all relevant mathematics used, concentrating on the Fast Fourier Transform (FFT), which is the main mathematical tool used in the analysis, giving it both in its analytical and discrete forms, and showing how it can be used to find the crosscorrelation between two data streams. Noise is briefly discussed statistically and the mathematical technique employed to extract signals from noise, *Matched Filtering*, is described in some detail. The second part of this chapter briefly describes the computational hardware used and also discusses parallel processing.

Chapter 4 describes the Glasgow prototype detector in detail, describing it as a *Fabry-Perot* laser interferometer. It is compared and contrasted with the German detector which is of a different design, being a *delay line interferometer.* The structure of the data taken by the Glasgow detector is described in detail.

Chapter 5 describes the software in detail. It begins by describing the overall parallel structure of the program before going on to describe the separable procedures within the program in more detail. Included in this are many figures that show the changes in the noise as it passes through the system.

Chapter 6 gives the results of the analysis. This includes an assessment of the noise, including how it correlates with the various housekeeping channels. It also includes an an assessment of the numbers and distributions of the events seen, investigating if they could be due to something that may appear in the housekeeping channels.

Chapter 7 gives a summary of the work described in the previous chapters, commenting on the results gained and lessons learned in the development and implementation of the system.

Beyond this there is a complete bibliography, followed by 3 Appendices, one of which is a complete listing of the software described in Chapter 5.

# Chapter 2

# SOURCES OF GRAVITATIONAL WAVES

## 2.1 Introduction

Much of the incentive behind the drive to design and construct a gravitational wave detector comes from an understanding of its potential sources and the vast new avenues of research that their discoveries could promote. Compared with those regions of the electromagnetic spectrum that have been explored for years now, gravitational radiation is of low frequency and would provide a new observational perspective, allowing the universe to be considered in terms of mass distribution rather than as just a collection of regions emitting various forms of electromagnetic radiation.

Gravitational wave signals looked for on Earth are generally considered to fall into three main classes; *burst*, *continuous* and *stochastic.* Any signal whose duration is of such length that the rotation of the Earth does not have to be taken in account, when extracting it from a data stream, is called a burst. Any longer signal then is considered to be continuous. Numerous weak signals would overlap producing varying levels of stochastic background radiation at all frequencies.

A good upper bound for the amplitude of the gravitational waves $h$, from these sources at the Earth is given by

$$\mid \mathrm{h} \mid \leq \frac{\Phi_N \Phi_{int}}{c^4}, \tag{2.1}$$

(Schutz 1984) where $\Phi_N$ and $\Phi_{int}$ are the Newtonian potentials at the observer's distance ($r$) and inside the source respectively, with

$$\Phi_N = \frac{GM}{r}.$$

## 2.2 BURSTS

### 2.2.1 Gravitational collapse.

Detection of the gravitational radiation produced by the collapse of the cores of massive stars has, over the years, been the main motivation behind gravitational wave detector research and development. It remains to this day the most likely form of event that bar detectors are likely to find. These collapses are possibly associated with type II supernovae and produce neutron stars or black holes in the mass range 1-10$M_\odot$, radiating waves in the frequency range 1-10 kHz.

The amplitude of the gravitational radiation received on Earth from one of these events, would depend on the energy released (and therefore the mass of the star) during the collapse, the distance of the source from Earth and the extent to which the core collapse was nonspherical. The axisymmetric calculations of Muller(1982) and Piran and Stark(1986) show that the greater the mass of the body produced, the greater the percentage of that mass that is radiated away as gravitational waves. For example for a 1 $M_\odot$ neutron star 0.00001% of its mass is realised in radiation whereas for a black hole of mass 10 $M_\odot$ the figure is $10^{-2}$% .

This nonaxisymmetric consideration hinges largely on the rotation speed of the collapsing core. If the core rotates only moderately rapidly then during the collapse it will flatten out axisymmetrically. With a relatively larger rotation speed however, initially insignificant nonaxisymmetric instabilities may start growing rapidly, altering the shape of the core, making the rotation less and less axisymmetric increasing the amount of gravitational radiation produced.

The structure of the waves emitted in either of the above scenario s is still not known analytically, (although numerical models for core collapse are currently under development,- Finn(1989) and Muller(1991)) hence detection by matched filtering is not possible. For this reason it is necessary for the waves to be emitted at amplitudes great enough to cause them to stand out above the noise within a detector, to enable them to be detected.

Using figures computed by Piran and Stark(1986) based on axisymmetric collapse an estimate of the upper limit for the wave amplitude is

$$h \approx 4 \times 10^{-22}, \tag{2.2}$$

for supernovae leading to the production of black holes in the Virgo cluster of galaxies, where roughly 50 supernovae are thought to occur each year. Hence with a sensitivity of $10^{-22}$, which detectors should be achieving within the next ten years or so, the hopefully large number of appropriate events in Virgo that approach this upper limit should be detected. An estimate for the amplitude of waves produced by nonaxisymmetric collapse can be obtained from Schutz(1990) using equation 2.1 taking

$\Phi_{int} \approx 0.3$ and $M \approx 10M_\odot$ for a black hole in the Virgo cluster we get

$$h \approx 1 \times 10^{-20}.$$

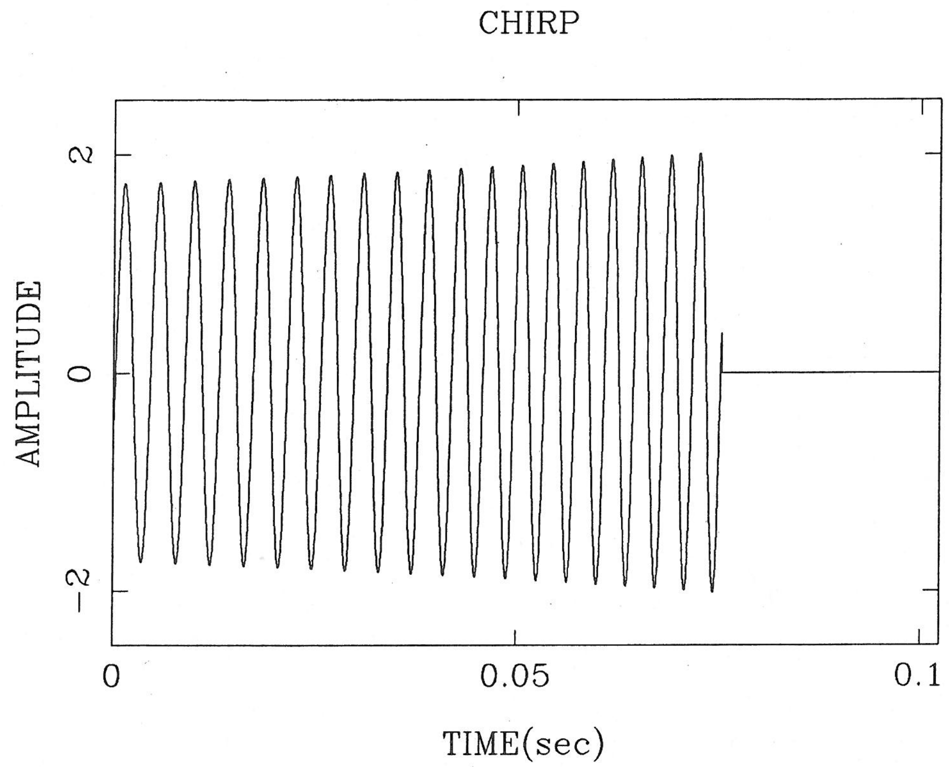
However if a gravitational collapse took place in our galaxy then $h$ could be of the order $10^{-18}$ which is a sensitivity achieved now by cryogenic bars and some prototype LIGO's. Estimates of how frequently such events may occur in the Milky Way vary from about 3-30 years or so, but bars are fairly easy to run over long periods of time, so a supernova in the next few years, in our galaxy, may well provide the first direct evidence for the existence of gravitational waves.

## 2.2.2 Coalescing Binaries

Binary systems emit gravitational waves along their rotational axis. These waves carry energy away from the system, causing both the separation of the bodies to decrease and the frequency and amplitude of the gravitational waves emitted to increase until coalescence. The process by which a particle experiences a reaction force when it emits radiation and hence loses energy is called *radiation reaction*. When both components of the system are compact, *ie* neutron stars and/or black holes, then frequencies and amplitudes of an order that should be detected by planned Earth based detectors, may be achieved before coalescence.

The mathematical model describing the coalescence and the nature of the gravitational radiation emitted is believed to be well known. Clarke and Eardley (1977) established a reasonably straight forward Newtonian approximation to the time evolution of the waves that Krolak (1989) found to be close to the post-Newtonian order. The model is supported by observations of the most famous example of a binary system that will, eventually, coalesce *ie*, PSR 1913+16, which lies in our own galaxy, with both components at about 1.4 solar masses. One component of the system is a pulsar and its companion a neutron star. Currently the system's orbital period is about 8 hours, and falling very slowly. Observations of it are sufficiently accurate to quantify the extent of this orbital decay and have shown that the figure achieved agrees to a high degree of accuracy with that predicted by the model calculations due to the loss of energy by the emission of gravitational waves (see Taylor and Weisberg (1989) and references there in). This provides indirect evidence for the existence of gravitational waves. In $1.49 \times 10^8$ years the components of PSR 1913+16 will coalesce, with it emitted waves attaining in the last few hundredth s of a second before coalescence, frequencies which would enable it to be seen by currently planned ground detectors on Earth, with

$$h \approx 10^{-19}$$


CHIRP
AMPLITUDE
2
0
−2
0
0.05
0.1
TIME(sec)

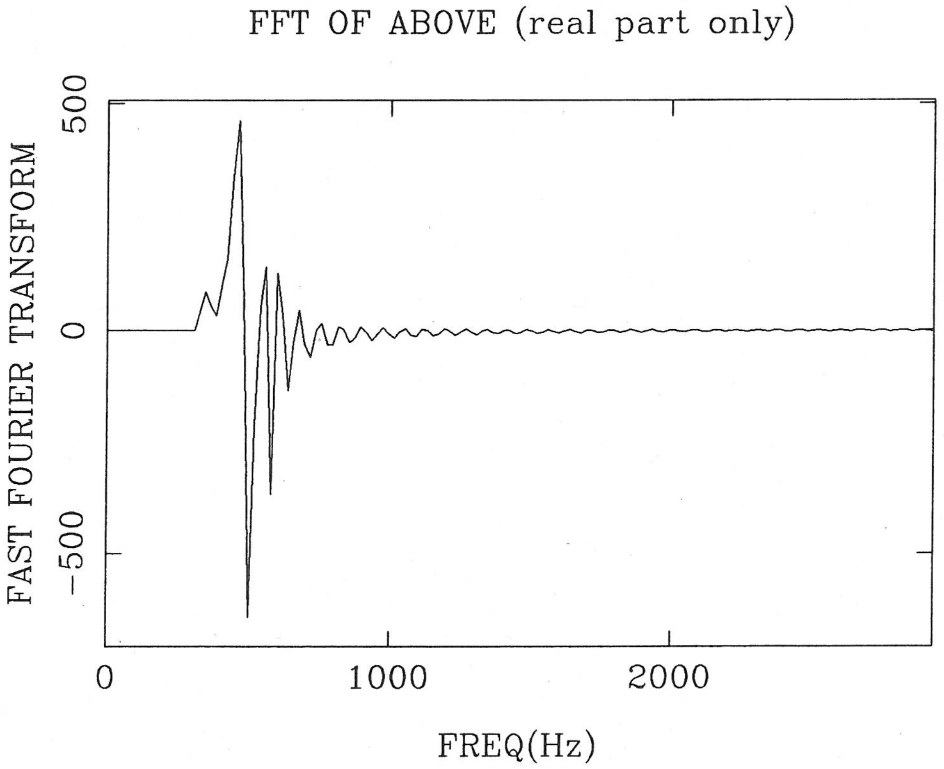

FFT OF ABOVE (real part only)
FAST FOURIER TRANSFORM
500
0
−500
0
1000
2000
FREQ(Hz)

FIGURE 2.1

at frequencies of around 300-1000 Hz. This type of characteristic waveform, with increasing amplitude and frequency, is known as a *chirp.* A chirp with a frequency range between 300 and 1000 Hz is shown in Figure 2.1, together with its Fourier transform.

Having a good model of the signal produced near coalescence greatly increases the chances of detecting such an event. The nature of the chirp, between those frequency ranges detectable on Earth, is dependent on both the masses of the components of the system, (which can be combined into one number referred to as the *mass parameter* and defined below) and also the intial *phase* of the wave as it achieves some appropriate frequency, chosen to be 100Hz in this analysis. In detecting an incoming chirp these parameters must be established along with its time of arrival. This is achieved by *Matched Filtering* which extracts the signal from the detector noise by cross correlating the data with a template of filters each based on a different expected gravitational wave signal *ie* varying in mass parameter and intial phase. This technique will be discussed further in Chapter 3. The expressions used to calculate the filters are given below.

## Mathematical description of coalescing binaries.

It can be shown by equating the rate of energy loss in the system with the gravitational luminosity that the time scale, $\tau$, over which the orbit decays is given by

$$\tau = \frac{\mathrm{f}}{\dot{\mathrm{f}}} = 7.8 M^{-2/3} \mu^{-1} \left( \frac{\mathrm{f}}{100\mathrm{Hz}} \right)^{-8/3} \mathrm{sec} \tag{2.3}$$

where f is linear frequency, $M$ is the total mass of the system ($= M_1 + M_2$) and $\mu$ is the reduced mass ($= M_1 M_2 / M$), both in units of solar masses. This can be written in the form,

$$\tau = \frac{\mathrm{f}}{\dot{\mathrm{f}}} = \alpha \mathrm{f}^{-8/3}, \tag{2.4}$$

where $\alpha$ is,

$$\alpha = \frac{7.8 \times 100^{8/3}}{\rho} \mathrm{sec}^{2}.$$

$\rho$ is the *mass parameter* given by,

$$\rho = \mu M^{2/3} / M_{\odot}^{5/3} .$$

It should be noted that as the rate of decay increases so rapidly, the true time to coalescence is only $\frac{3}{8}\tau$.

By integrating equation 2.4 over f and t from some intial point when f and t are known and given by $\mathrm{f}_0$ and $\mathrm{t}_0$ to some later point, an expression giving the evolutionary relationship between f and t is found,

$$f = \left(\frac{-8}{3\alpha}(t - t_0) + f_0^{-8/3}\right)^{-3/8} \text{Hz} \tag{2.5}$$

and also therefore $\omega$, the angular frequency, since

$$\omega = \pi f.$$

The frequency of the waves is twice the orbital frequency of the system because the initial configuration occurs twice in every orbit.

The amplitudes of both the polarisation states are given by

$$h \approx \frac{4G^{5/3}\pi 2/3}{c^4}[\frac{\mu}{r}][f\ M]^{2/3}\cos(2\pi t \int f(t)dt + \xi) \tag{2.6}$$

where $\xi$ is some phase constant contained with the received waves.

By substituting equation 2.5 into the above we obtain an expression for the chirp in terms of time that has unit amplitude and a frequency of 100Hz when t = 0,

$$\text{Chirp}(t) = \beta^{-1/4}\cos\left(\frac{320\pi(1-\beta)^{5/8}}{0.34\rho} + \phi\right) \tag{2.7}$$

where $\phi$ is the phase of the chirp at 100Hz and

$$\beta = 1 - 0.34\rho t.$$

This is used as the model for the formation of the filter template used for matched filtering. Using equation 2.5 an expression for the time t in terms of the frequency f is found,

$$t_f = \frac{1 - (f/100\text{Hz})^{-8/3}}{0.34\rho}\text{sec}. \tag{2.8}$$

From the above, an expression for the time taken for the frequency of the chirp to go from one frequency, $f_0$, to a higher frequency, $f_1$, is given by,

$$t_{(f_0 \to f_1)} = \frac{(f_0/100\text{Hz})^{-8/3} - (f_1/100\text{Hz})^{-8/3}}{0.34\rho}\text{sec}. \tag{2.9}$$

The maximum value of the gravitational wave amplitude $h_{max}$ from a binary system at a distance $r$ is achieved along the axis of the system and is given by,

$$h_{max} = 2.6 \times 10^{-23}\rho\left(\frac{f}{100\text{Hz}}\right)^{2/3}\left(\frac{100\text{Mpc}}{r}\right). \tag{2.10}$$

This can be shortened to

$$h_{max} = \gamma\left(\frac{f}{100\text{Hz}}\right)^{2/3}\left(\frac{100\text{Mpc}}{r}\right), \tag{2.11}$$

where $r$ is the distance in Mpc and

$$\gamma = 1.207 \times 10^{-22}\rho.$$

It can be seen comparing equation 2.4 and equation 2.11 that the product $h_{max}\tau$, for a particular frequency is dependent solely on the distance to the binary. Hence, compensating for whatever non alignment there is between the Earth and the axis of the system, it is possible to reliably work out the distances to any coalescing binaries detected (see Schutz (1986b) and Bourzeix *et al* (1990)) and hence establish the distances to the system, be it a galaxy or a group of galaxies, containing the binary. Comparing these figures with those obtained by studying the redshift of the system should allow the most accurate assessment of the Hubble constant $H_0$ yet made.

An obvious consideration in the design of equipment and data analysis techniques specialised towards finding coalescing binaries is that there should be enough potential events of sufficient amplitude to make the search worthwhile. Current estimates on the event rate, based on the pulsar birth rate and the observed percentage of these that turn out to be binary pulsars suggest a figure of 3 events per year out to 200Mpc rising to 24 at 400Mpc and 81 at 600Mpc. Krolak (1989) arrived at an equation for the best signal-to-noise ratio achievable for a binary system at a distance $r$ in terms of the lower cut off frequency, $f_{lc}$, of the detector, its sensitivity $h$ and the system's mass parameter $\rho$,

$$\left(\frac{S}{N}\right) = 22\rho^{1/2}\left[\frac{h}{10^{-22}}\right]^{-1}\left[\frac{f_{lc}}{100\text{Hz}}\right]^{-7/6}\left[\frac{r}{100\text{Mpc}}\right]^{-1}. \tag{2.12}$$

Applying the above equation to the Glasgow prototype, taking $h \approx 10^{-19}$, $f_{lc} = 300$Hz and the $S/N$ to be 3.5, we obtain $r = 0.2$Mpc for a system of two $1.4M_\odot$ neutron stars. This represents a volume of space roughly equal to only 1/8 of the volume of the local group of galaxies and comparing it with the figures quoted above make it unlikely in the extreme that any event may have picked up by the Glasgow detector during its hundred hour run of March 1989. A prospective future detector with $h \approx 10^{-22}$ and $f_{lc} \approx 100$Hz, would be observing out to 650Mpc when there should be several events picked up per month. Obviously if either one or both the components of the system is a black hole then they would be detectable out to far greater distances.

## Black Hole Ringing

There are several other possible applications of matched filtering in gravitational wave detection. One of these may be the result of a non axisymmetrical gravitational collapse leading to the production a deformed black hole, which would then undergo damped vibrations emitting gravitational radiation. The parameter dependent shape of the waveform, produced by the ringing down, is believed to be known, (see Chandrasekhar and Detweiler (1975)), hence filter templates could again be built up that could pick the signal out of the data stream.

## 2.3 CONTINUOUS WAVE SIGNALS

Gravitational wave signals lasting more than 30 minutes are classified as continuous. When trying to detect signals of this duration, frequency broadening due to the doppler shifts that occur because of the motions of the Earth must be taken into account. In the short term only sources of known frequency and direction may be detected, because the sensitivities over long periods of time that detectors would need to achieve so as to search for unknown sources are not attainable. Knowing the direction and frequency of a source would enable a detector to be optimised, concentrating its efforts in looking specifically for that source. The chances of detection could then be further increased by cross-correlating data from several detectors.

### 2.3.1 Periodic and other Sources

#### Pulsars

It is thought that pulsars would be the main source of continuous wave signals. They are thought to be supernovae remnants, and emit gravitational radiation by virtue of their non-axisymmetric magnetic fields. The amplitude of this radiation by itself is sufficiently slight as to be beyond the scope of any detector likely to appear in the near future. If, however, the neutron star has any significant deviation from symmetry around its rotation axis then the amplitude of the radiation produced may be sufficiently large as to be detected. This asymmetry may be caused by the deformation of the surface of the neutron star, due possibly to non uniform core collapse and internal magnetic pressures. The extent to which the overall shape of the star may be distorted in this way is the subject of much debate but if the ellipticity (or fractional distortion) of the star is $\delta$ then an estimate of the amplitude h of the radiation emitted for a star of mass 1.4 $M_\odot$ and radius 10 km is,

$$h \approx 6 \times 10^{-22}\,\delta \left[\frac{\mathrm{f}}{100\mathrm{Hz}}\right]^2 \left[\frac{10\mathrm{Kpc}}{r}\right], \tag{2.13}$$

where f is the frequency of the wave (which is twice the frequency of rotation of the neutron star) and $r$ is its distance in units of 10 Kpc. Obviously the energy radiated away from the star cannot exceed its observed loss of kinetic energy in any given time period. Hence knowing its mass allows the calculation of an upper limit for $\delta$. Knowing the pulsar's distance and frequency also enables an estimate of the upper limit of h for the pulsar, *eg* for the *Crab* pulsar, which emits waves of frequency

60 Hz at a distance of 2 Kpc, $\delta$ and $h$ are bounded by

$$\delta < 10^{-3}$$

and

$$h < 1.1 \times 10^{-24}.$$

### Wagoner Radiation

If a rapidly spinning neutron star has a binary companion from which it accretes mass, its angular momentum grows and may eventually reach a point of instability after which the system will, in striving for stability, radiate away angular momentum in the form of gravitational waves. This would continue until a balance point is reached when the amount of energy radiated away matches that gained through accretion. When this point is reached the system becomes and remains, a fixed frequency emitter. Gravitational radiation from this source is known as Wagoner Radiation and is another potentially detectable periodic source (see Wagoner, 1984).

## 2.3.2 Stochastic Background

The stochastic background of gravitational radiation could consist of many waves of unpredictable amplitudes and frequencies. Possible causes would be all potential emitters of gravitational radiation, predicted and unpredicted, including sources already discussed in this chapter whose radiative amplitude, because of the source distance or whatever, is tooslight to detect on Earth. Possible other sources could include cosmic strings and events in the early universe.

This background could only be detected by cross-correlating the output of several detectors. The detectors must be closer than roughly half the wavelength of the expected radiation if they are to have optimum sensitivity. For a detailed description Michelson (1987).

## 2.3.3 Low Frequency Sources

As previously mentioned, these sources would be the province of the space based detector. Possible sources could include long period binaries in our own galaxy and the formation and coalescence of massive ($> 10^6 M_\odot$) black holes in other galaxies.

# Chapter 3

# MATHEMATICAL TECHNIQUES

## 3.1 Introduction

This Chapter discusses the principles behind the essential mathematics used in this thesis. It is concerned mainly with Fourier analysis, and how associated techniques are used to extract signals from streams of noise. It begins by defining the Fourier transform of a function, and then shows how this transform is related to the cross-correlation of two functions. Both the Discrete Fourier transform and the Fast Fourier transform are then defined, and the equivalent cross-correlation relationship is shown. This is followed by a discussion about the statistics of noise and how matched filtering methods can be used to extract signals from this noise.

## 3.2 Fourier analysis

### 3.2.1 The Fourier transform

One of the principal tools of signal analysis is the Fourier transform which decomposes a signal down into a combination of sinusoidal waveforms. The Fourier transform of a continuous function $g(t)$ is given by,

$$G(f) = \int_{-\infty}^{+\infty} g(t)e^{-2\pi ift}\mathrm{d}t, \qquad -\infty < f < \infty, \tag{3.1}$$

where $G(f)$ is a function of the frequency $f$.

This transform enables a process described in the time domain to be described in the frequency domain and vice versa, because the inverse transform also applies,

$$g(t) = \int_{-\infty}^{+\infty} G(f)e^{2\pi ift}\mathrm{d}f, \qquad -\infty < t < \infty. \tag{3.2}$$

**Power**

The power spectrum is defined as $|G(f)|^2$ , which leads to

$$\text{total power} = \int_{-\infty}^{\infty} |G(f)|^2 \mathrm{d}f = \int_{-\infty}^{\infty} |g(t)|^2 \mathrm{d}t. \tag{3.3}$$

This famous result is known as Parseval's Theorem.

### 3.2.2 Correlation

The *cross-correlation* of two functions, $g(t)$ and $h(t)$, is a function $c(t)$ of $t$, and given by,

$$\mathrm{CrossCorr(g,h)} = c(t) = \int_{-\infty}^{\infty} g(t_1 - t)h(t_1)\mathrm{d}t_1 = \int_{-\infty}^{\infty} g(t_2)h(t_2 + t)\mathrm{d}t_2. \tag{3.4}$$

If $G(f)$ and $H(f)$ are the Fourier transforms of $g(t)$ and $h(t)$ respectively, and if the Fourier transform of the correlation is $\mathrm{F}[\mathrm{CrossCorr}(g,h)]$, then

$$\mathrm{F}[\mathrm{CrossCorr}(g,h)] = G(f)H(f)^*, \tag{3.5}$$

where $*$ represents the complex conjugate. From this obviously $c(t)$ can be found by taking the inverse Fourier transform,

$$c(t) = \mathrm{CrossCorr}(g,h) = \int_{-\infty}^{\infty} G(f)H(f)^* e^{2\pi i f t} df. \tag{3.6}$$

If a function $g(t)$ is cross-correlated with itself, the resulting function $a(t)$ is called the *auto-correlation* function and is given by,

$$\mathrm{AutoCorr}(g) = a(t) = \int_{-\infty}^{\infty} g(t_1 - t)g(t_1)\mathrm{d}t_1 = \int_{-\infty}^{\infty} |G(f)|^2 e^{2\pi i f t}\mathrm{d}f. \tag{3.7}$$

### 3.2.3 The Discrete Fourier Transform (DFT)

The data streams we deal with are not read continuously but are sampled discretely at evenly spaced intervals of time. If $\triangle$ is the time interval between consecutive samples then a function $g(t)$ would be read and stored in memory as a series $g_n$ given by,

$$g_n = g(n\triangle) \quad n = 0, 1, 2, 3 ... N - 1. \tag{3.8}$$

The sampling rate (or frequency), *ie* the number of samples each second, is just $1/\triangle$ .

Because of this discrete sampling it is not possible to use the Fourier transform in its continuous form.

**Discrete Fourier Transform**

If we define our signal as a series of $N$ numbers, $g_k$ ($k = 0, 1, ..., N-1$), where $N$ is a power of 2, and the DFT of the signal as a series $\tilde{g}_j$ ($-N/2, ..., N/2-1$), then these series are related by the transform pair,

$$\tilde{g}_j = \frac{1}{N}\sum_{k=0}^{N-1} g_k e^{-2\pi ijk/N} \qquad j = -N/2, .., N/2-1 \tag{3.9}$$

and

$$g_k = \sum_{j=-N/2}^{N/2-1} \tilde{g}_j e^{2\pi ijk/N} \qquad k = 0, .., N-1. \tag{3.10}$$

The time duration of the signal is $(N-1)\Delta$ and the frequencies are resolved only at discrete values given by,

$$f_n = \frac{n}{N\Delta}, \quad n = -N/2, ..., N/2. \tag{3.11}$$

The above equation gives $(N+1)$ values of $f_n$, but only $N$ distinct values as $f_n$ for $n = -N/2$ and $n = N/2$ are identical.

The discrete Fourier transform of a data stream can only correctly pick out frequencies of up to half the sampling frequency. This critical frequency $f_c$, is known as the *Nyquist frequency*, and is given by,

$$f_c = \frac{1}{2\Delta}. \tag{3.12}$$

If a sampled function contained no frequencies greater than the Nyquist frequency, then it would be completely determined by sampling over an infinite time duration. However if $f_c$ is exceeded then problems can arise due to *aliasing*, which allows relatively large frequencies to masquerade as frequencies less than $f_c$. This can be seen by considering a sine wave of frequency slightly greater than half the sampling frequency. If the wave is sampled at its positive peak, the next sampled point is just after the wave trough. However an external observer would interpret the point as being just prior to the trough, getting the wrong wavelength and hence the wrong frequency. This can be seen mathematically for a sine wave of frequency $f > f_c$ sampled with frequency $f_s$,

$$\begin{aligned}
\sin 2\pi ft &= \sin 2\pi[(f-f_s)t + f_s t] && (3.13)\\
&= \sin 2\pi(f-f_s)t \, \cos 2\pi f_s t + \sin 2\pi f_s t \, \cos 2\pi(f-f_s)t && (3.14)\\
&= \sin 2\pi(f-f_s)t_n \quad \text{at } t_n = n/f_s \quad n = 0, 1, 2, ... \,. && (3.15)
\end{aligned}$$

Then by a simple inductive argument,

$$\sin 2\pi ft_n = \sin 2\pi(f-f_s)t_n = \sin 2\pi(f-nf_s)t_n, \quad n = 1, 2, 3, .., \tag{3.16}$$

and if $|f - nf_s| < f_c$, it would be picked up as a lower frequency, by the discrete Fourier transform.

**Power**

Parseval's Theorem has a discrete form also,

$$\sum_{k=0}^{N-1} |f_k|^2 = \frac{1}{N}\sum_{n=0}^{N-1} |\tilde{f}_n|^2. \tag{3.17}$$

**Discrete Correlation**

When carrying out actual analysis we may want to cross-correlate a finite data set $h_j$ of length $N$ with an infinitely long set $g$. This is not possible, so it becomes necessary to break $g$ up into many data sets $g_k$ again of length $N$, and then carry out the correlations set by set.

Therefore if we have two data sets $g_k$ and $h_j$ each of $N$ elements, with $k, j = 0, 1, ..., N - 1$, then their crosscorrelation, $c_j$, is given by the *Circular Correlation* formula,

$$c_j = \frac{1}{N}\sum_{k=0}^{N-1} g_k h_{k+j}, \quad j = 0, ..., N-1 \, . \tag{3.18}$$

It is necessary to make $h_j$ periodic, *ie*

$$h_{j+N} = h_j \quad \text{for all } j \, , \tag{3.19}$$

unfortunately however problems will then arise from this periodicity. As $k$ increases the beginning of the time series would be multiplied by the end of the filter which is clearly erroneous. This *wraparound* can be partially overcome by ensuring that $h_j$ is largely made up of zeros, in which case $g_k$ is much longer than the non zero part of $h_j$ as they are the same total length. In this case, if there are $N_h$ non-zero points in $h_j$, then the last $N_h$ points of the correlation would have to be ignored, provided that the non zero part of the filter is at its start.

Analogous to crosscorrelation using the continuous Fourier transform, the DFT of the circular correlation $c_j$, given by $\tilde{c}_k$, is related to the DFT's of $g_j$ and $h_j$, given by $\tilde{g}_k$ and $\tilde{h}_k$ respectively by the discrete version of equation given in 3.5 , *ie*

$$\tilde{c}_k = \tilde{g}_k(\tilde{h}_k)^* \quad k = -N/2, .., N/2-1, \tag{3.20}$$

where * again represents the complex conjugate.

### 3.2.4 The Fast Fourier Transform (FFT)

Calculating the DFT of a series of $N$ numbers normally would require $O(N^2)$ multiplications, which, when many DFT's are required of long series, can be very time consuming, even when the fastest of today's computers are used. Hence the widespread use of the fast Fourier transform.

The FFT is a computer algorithm that, given a series of $N$ numbers, produces the DFT of the series with only $O(N\log_2 N)$ multiplications, which compares very favorably with $O(N^2)$ when $N$ is large. The proof is discussed in detail in Bracewell(1986) and many other publications, including Press *et al*(1986) from which the FFT algorithms used in this work were taken. It follows generally from the fact that when $N$ is an even number, the DFT of the series can be decomposed into a linear combination of the DFT's of two series of length $N/2$, the first series being the even numbered points of the original stream, and the second the odd,

$$\begin{aligned}
\tilde{a}_j &= \frac{1}{N}\sum_{k=0}^{N-1} a_k e^{-2\pi ijk/N} && (3.21)\\
&= \frac{1}{N}\sum_{k=0}^{N/2-1} a_{2k} e^{-2\pi ij(2k)/N} + \frac{1}{N}\sum_{k=0}^{N/2-1} a_{2k+1} e^{-2\pi ij(2k+1)/N} && (3.22)\\
&= \frac{1}{2}\left\{\frac{2}{N}\sum_{k=0}^{N/2-1} a_{2k} e^{-2\pi ijk/(N/2)} + e^{-2\pi ij/N}\frac{2}{N}\sum_{k=0}^{N/2-1} a_{2k+1} e^{-2\pi ijk/(N/2)}\right\}. && (3.23)
\end{aligned}$$

In equation 3.23, the summations within the brackets are simply the DFT's of the even and the odd numbered elements of $\tilde{a}_k$ respectively, for $j < N/2$. When $j \geq N/2$, the summations are equal to those for $j - N/2$. This can be shown, considering only the even numbered summation, as follows,

$$\begin{aligned}
\frac{2}{N}\sum_{k=0}^{N/2-1} a_{2k} e^{-2\pi ijk/(N/2)} &= \frac{2}{N}\sum_{k=0}^{N/2-1} a_{2k} e^{-2\pi i(j-(N/2)+(N/2))k/(N/2)} && (3.24)\\
&= \frac{2}{N}\sum_{k=0}^{N/2-1} a_{2k} e^{-2\pi i(j-(N/2))k/(N/2)} e^{-2\pi i(N/2)k/(N/2)} && (3.25)\\
&= \frac{2}{N}\sum_{k=0}^{N/2-1} a_{2k} e^{-2\pi i(j-(N/2))k/(N/2)}. && (3.26)
\end{aligned}$$

Analogously, the same can be applied to the other summation over the odd numbered elements of $\tilde{a}_k$. Thus, from the two transforms, over $(N/2)$ elements, $\tilde{a}_k$ can be completely constructed with only the $N$ extra complex multiplications due to the factor $e^{2\pi ij/N}$ in front of the second summation. If $N$ is a power of 2, these steps can be repeated recursively until the DFT is just a linear combination of transforms of length 1. This occurs after $\log_2 N$ steps. Each step requires $N$ complex multiplications to produce the transforms needed one step higher, hence the total number of operations is of order $N\log_2 N$. The Inverse Fast Fourier Transform (IFFT) is performed using a very similar method.

When performing circular correlations computationally it is most efficient, in terms of time, to use equation 3.20, utilizing FFT algorithms to find the DFT's.

## 3.3 Extraction of a signal from a noisy background

As described in Chapter 2, the three main classes of signal are continuous, burst and stochastic. This analysis is primarily concerned with bursts, in particular milli-second broadband bursts, *ie* events whose energy is spread throughout a wide frequency range, typically produced by supernovae and also 'chirp' waveforms produced by coalescing binaries.

Either of these types of bursts could stand out above the noise, and could hence be detected by simply exceeding some preset threshold in $S/N$, but often this would not be the case, they would be hidden within the noise and therefore be invisible to the simple $S/N$ criteria. As the chirp waveform is relatively long and, hopefully at least, well known, the probability of its detection in particular, would benefit from *Matched Filtering* which is used to pull the signal out, above the noise and make it detectable by the same $S/N$ criteria as above.

### 3.3.1 Noise

In developing a model for the matched filtering it was necessary to assume that the noise in a detector would be a random variable in both frequency and time, with zero expectation value $\langle\, n(t)\, \rangle$ (or mean), *ie*,

$$\langle\, n(t)\, \rangle \;=\; \langle\, \tilde{n}(f)\, \rangle \;=\; 0\,. \tag{3.27}$$

From this the variance, $\sigma^2(f)$, of the noise is given by,

$$\sigma^2(f) \;=\; \langle[\tilde{n}(f) \;-\; \langle\tilde{n}(f)\rangle]^2\rangle \;=\; \langle\tilde{n}^2(f)\rangle. \tag{3.28}$$

It was also necessary to assume that the general statistical properties of the noise don't change with time, in which case the noise is said to be *stationary* and its *spectral density*, $S(f)$, is defined by the equation,

$$\langle\, \tilde{n}(f)\, \tilde{n}^*(f')\rangle \;=\; S(f)\, \delta(f \;-\; f')\,, \tag{3.29}$$

where $\delta(f \;-\; f')$ is a delta function.

This implies that noise at any frequency $f$ is not correlated with the noise at any other frequency $f'$ with $f \neq f'$, and that the function, $S(f)$, is equal to the variance of the noise at that frequency $f$, *ie*,

$$S(f) \;=\; \sigma^2(f) \tag{3.30}$$

If the prevailing statistic within the noise, *ie* the variance or the mean, changed during the course of an analysis run, then the results of the filtering would be called into question, hence the noise statistics must be monitored continually during the run. It is assumed here that the noise is white, *ie* the spectral density $S(f)$, is not a function of $f$.

For noise that is both white and Gaussian, the probability density function $P(x)$ is given by,

$$P(x) = \frac{1}{(2\pi)^{1/2}\sigma} \mathrm{e}^{-(\mathrm{x}-\bar{\mathrm{x}})^2/2\sigma^2}. \tag{3.31}$$

This means that the probability of any one sample lying in an interval d$x$ at $x$ is given by $P(x)$d$(x)$. This number is numerically equal to the number of samples in the interval d$x$, divided by the total number of samples.

This also has the property, now assuming zero mean, that

$$P(-\infty < x < \infty) = \frac{1}{(2\pi)^{1/2}\sigma} \int_{-\infty}^{\infty} \mathrm{e}^{-\mathrm{x}^2/2\sigma^2}\, \mathrm{dx} = 1, \tag{3.32}$$

as the sample has to lie somewhere.

The probability then, that a sample would lie at a distance, greater than some threshold $T$ from the mean (still assumed to zero), is given by,

$$P(|x| > T) = \frac{1}{(2\pi)^{1/2}\sigma} \int_{-\infty}^{-T} \mathrm{e}^{-\mathrm{x}^2/2\sigma^2}\, \mathrm{dx} + \frac{1}{(2\pi)^{1/2}\sigma} \int_{\mathrm{T}}^{\infty} \mathrm{e}^{-\mathrm{x}^2/2\sigma^2}\, \mathrm{dx} \tag{3.33}$$

$$= \frac{1}{\sigma}\left(\frac{2}{\pi}\right)^{1/2} \int_{T}^{\infty} \mathrm{e}^{-\mathrm{x}^2/2\sigma^2}\, \mathrm{dx} \tag{3.34}$$

$$\approx \left(\frac{2}{\pi}\right)^{1/2} \left(\left(\frac{\sigma}{T}\right) - \left(\frac{\sigma}{T}\right)^3 + \ldots\right) e^{-T^2/2\sigma^2}. \tag{3.35}$$

The last line is an asymptotic approximation to the previous equation.

In order to limit the expected number of false alarms to one in $N$ data points we must solve,

$$P(|x| > T) = \frac{1}{N}, \tag{3.36}$$

for $T$. If the Glasgow sampling rate of 20kHz were applied for a whole year, and we wanted only one false alarm in that time, then we would need $T \approx 7.1\sigma$.

When gravitational wave detectors are established around the world, clearly a suspected event in one could only be credited as an actual event if witnessed in other detectors as well. This *coincidence* criterion allows the threshold $T$ to be reduced as a false alarm would be a threshold crosser simultaneously in two or more detectors.

For two detectors with noise levels $\sigma_1$ and $\sigma_2$ the equation to solve is now,

$$P(|x| > T_1)P(|x| > T_2) = \frac{1}{N}. \tag{3.37}$$

If we consider two identical detectors at the same site running at 20kHz for a year, the threshold for both is $T \approx 4.85\sigma$ . For three it reduces still further to $T \approx 3.8\sigma$ .

There are several kinds of noise produced within interferometers, and their sources will be discussed in Chapter 4. In the Glasgow prototype, *seismic noise* is dominant at low frequencies, $f_s \leq 300\,\mathrm{Hz}$, which effectively provides a low frequency cutoff at $f_s$ which would have to be taken into account when forming filters.

## 3.3.2 Matched Filtering

Matched filtering is a mathematical technique that is employed to extract signals of a known form from noise. It is performed by cross-correlating a stream of noise with a filter resembling the signal to be found. Essentially the filter moves along the noise point by point, comparing each stretch of data with the filter in turn. The value returned then would be proportional to the extent of the matching of the filter, with the length of data, following the first point where the filter is applied. For a more complete discussion of matched filtering and it's applications within the search for gravitational waves, see Schutz(1990).

Quantitatively if a signal $h(t)$ is present, buried in noise $n(t)$, then the output $o(t)$ of the detector is given by,

$$o(t) = n(t) + h(t). \tag{3.42}$$

Given the detector output, to establish whether or not there is a signal with certain parameters present, the output should be cross-correlated with the corresponding filter, *ie* a filter designed on the basis of the same parameters. If the filter is $q(t)$ then the correlation $c(t)$ is given by,

$$c(t) = \int_{-\infty}^{\infty} o(t')q(t'+t)\mathrm{d}t', \tag{3.43}$$

which from equation 3.6, can also be written in terms of $\tilde{o}(f)$ and $\tilde{q}(f)$ the Fourier transforms of $o(t)$ and $q(t)$ respectively as,

$$c(t) = \int_{-\infty}^{\infty} \tilde{o}(f)\tilde{q}(f)^{*}e^{2\pi ift}\mathrm{d}f. \tag{3.44}$$

Provided the noise is Gaussian, the expectation value of $c(t)$ is given by,

$$\begin{aligned} \langle\, c(t)\,\rangle &= CrossCorr(h,q) && (3.45)\\ &= \int_{-\infty}^{\infty} \tilde{h}(f)\tilde{q}(f)^{*}e^{2\pi if(t-\tau)}\,\mathrm{d}f. && (3.46)\end{aligned}$$

The correlation is maximised when $t = \tau$ , the time shift between the filter and the signal in the noise, which gives,

$$\langle\, c_{max}\,\rangle = \int_{-\infty}^{\infty} \tilde{h}(f)\tilde{q}(f)^{*}\,\mathrm{d}f. \tag{3.47}$$

The standard deviation of the correlation noise is given by the square root of the variance of the correlation, where the variance is given by,

$$\begin{aligned} \langle[c(t) - \langle c(t)\rangle]^2\rangle &= \langle[\mathrm{CrossCorr}(n,q)]^2\rangle && (3.48)\\ &= \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} \langle\tilde{n}(f)\tilde{n}(f')\rangle\, |\tilde{q}(f)^{*}\tilde{q}(f')^{*}\, e^{2\pi ift}e^{2\pi if't}|\,\mathrm{d}f'\,\mathrm{d}f && (3.49)\\ &= \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} S(f)\delta(f-f')\,|\tilde{q}(f)^{*}\tilde{q}(f')^{*}|\,\mathrm{d}f'\,\mathrm{d}f && (3.50)\\ &= \int_{-\infty}^{\infty} S(f)|\tilde{q}(f)^{*}|^2\,\mathrm{d}f && (3.51)\\ &= \int_{-\infty}^{\infty} S(f)|\tilde{q}(f)|^2\,\mathrm{d}f. && (3.52)\end{aligned}$$

The signal to noise ratio then is given by,

$$\frac{S}{N}(t) = \frac{\langle\, c(t)\,\rangle}{(\langle[c(t) - \langle c(t)\rangle]^2\rangle)^{1/2}} = \frac{\int_{-\infty}^{\infty} \tilde{h}(f)\tilde{q}(f)^{*}e^{2\pi ift}\,\mathrm{d}f}{[\int_{-\infty}^{\infty} S(f)\,|\tilde{q}(f)|^2\,\mathrm{d}f]^{1/2}}. \tag{3.53}$$

It can be shown that the filter $q(t)$, that maximises the chance of finding a signal $h(t)$, is as expressed in the Fourier plane,

$$\tilde{q}(f) = \frac{\tilde{h}(f)}{S.(f)}, \tag{3.54}$$

which leads to the largest attainable $S/N$ which from equation 3.53. is,

$$\left(\frac{S}{N}\right)^2 = \frac{\langle\, c_{max}\,\rangle}{\langle[c(t) - \langle c(t)\rangle]^2\rangle} \tag{3.55}$$

$$= \frac{(\int_{-\infty}^{\infty} \tilde{h}(f)\tilde{q}(f)^* \, \mathrm{d}f)^2}{\int_{-\infty}^{\infty} S(f)|\tilde{q}(f)|^2 \, \mathrm{d}f} \tag{3.56}$$

$$= \frac{(\int_{-\infty}^{\infty} \tilde{h}(f)\tilde{h}(f)^*/S\,(f) \, \mathrm{d}f)^2}{\int_{-\infty}^{\infty} S(f)|\tilde{h}(f)/S\,(f)|^2 \, \mathrm{d}f} \tag{3.57}$$

$$= \frac{(\int_{-\infty}^{\infty} |\tilde{h}(f)|^2/S\,(f) \, \mathrm{d}f)^2}{\int_{-\infty}^{\infty} |\tilde{h}(f)|^2/S\,(f) \, \mathrm{d}f}, \tag{3.58}$$

$$= \int_{-\infty}^{\infty} \frac{|\tilde{h}(f)|^2}{S\,(f)} \, \mathrm{d}f \tag{3.59}$$

$$= 2\int_{0}^{\infty} \frac{|\tilde{h}(f)|^2}{S\,(f)} \, \mathrm{d}f. \tag{3.60}$$

Thus by the use of matched filtering, chirp signals can be made to stand above the noise within the correlation and hence can be found by the application of the $S/N$ threshold criteria mentioned at the start of this section.

It should be noted that the search for broadband bursts, although effectively simpler - just applying the threshold criteria to a stretch of data- is strongly analogous to matched filtering, with the filter being a *delta* function in time. This enables us to demonstrate why narrow band sources may be easier to detect than broadband sources of the same energy.

Assuming that $S\,(f) \approx \sigma^2(f_c)$ where $f_c$ is some central frequency then equation 3.60 becomes,

$$\left(\frac{S}{N}\right)^2 = \frac{2}{\sigma^2(f_c)} \int_0^{\infty} |\tilde{h}(f)|^2 \, \mathrm{d}f \tag{3.61}$$

$$= \frac{2}{\sigma^2(f_c)} \int_0^{\infty} |\tilde{h}(t)|^2 \, \mathrm{d}t. \tag{3.62}$$

If the broadband burst is centered at frequency $f_b$ then its duration would approximately be $1/f_b$ and $|h(t)| \approx h$ then from equation 3.62 we would get,

$$\frac{S}{N} \approx \frac{h}{\sigma(f_b)f_b^{1/2}}. \tag{3.63}$$

Narrow band signals centered at frequency $f_n$ would last for a time $n/f_n$ where n is the number of cycles so equation 3.62 would become,

$$\frac{S}{N} \approx \frac{hn^{1/2}}{\sigma(f_n)f_n^{1/2}}. \tag{3.64}$$

It is apparent from the above that $S/N$ is a factor of $n^{1/2}$ times higher in the narrow band case where clearly it is beneficial to include as many cycles in the filters as is computationally expedient. The narrow band case, when comparing coalescing binaries with supernovae, can also benefit from having a lower central frequency.

If a signal $h(t)$ is picked up over a stretch of data by its corresponding filter $s(t)$ then,

$$\tilde{h}(f) = A\,\tilde{s}(f), \tag{3.65}$$

where $A$ is just some linear multiple. When $\tilde{s}(f)$ is normalised to unity then $A$ represents the actual amplitude of the wave. So putting equation 3.65 in equation 3.60 we get,

$$\left(\frac{S}{N}\right)^2 = 2\int_0^\infty \frac{|\tilde{h}(f)|^2}{S\ (f)}\,\mathrm{d}f \tag{3.66}$$

$$= 2A^2\int_0^\infty \frac{|\tilde{s}(f)|^2}{S\ (f)}\,\mathrm{d}f. \tag{3.67}$$

This gives an expression for $A$,

$$A = \left(\frac{S}{N}\right)\left(2\int_0^\infty \frac{|\tilde{s}(f)|^2}{S\ (f)}\,\mathrm{d}f\right)^{-1/2}. \tag{3.68}$$

Again assuming that $S_h(f) \approx \sigma^2(f_c)$, equation 3.68 becomes ,

$$A = \sigma\left(\frac{S}{N}\right)\left(2\int_0^\infty |\tilde{s}(f)|^2\,\mathrm{d}f\right)^{-1/2}. \tag{3.69}$$

In applying equation 3.69 to real, sampled data, its discrete version is used, *ie*,

$$A = \sigma\left(\frac{S}{N}\right)\left(2\,\delta\mathrm{f}\sum_{i=1}^{N}|\tilde{s}_i|^2\right)^{-1/2}, \tag{3.70}$$

where $\delta$f is the frequency spacing between samples and $N$ the total number of samples.

# Chapter 4

# The GLASGOW PROTOTYPE DETECTOR

## 4.1 Introduction

Work on the development of a gravitational wave detector, by a group in the Department of Physics and Astronomy at Glasgow University, began in the very early seventies, with the intial emphasis on resonant bar detectors. The group, headed at that time by Professor R.W.P Drever, concluded in about 1976 that the detection of changes in the relative displacements of nearby masses by optical interferometry, an idea first suggested by Weber in the 60's, represented a more promising means of detecting gravitational waves. For roughly the next three years they worked on a small scale *delay line* interferometer, switching in 1979 to the development of a 10 m *Fabry-Perot* prototype interferometer. It is this detector that the group, now headed by Professor J Hough, are currently working on. In the Max-Planck-Institut für Quantenoptik, Garching, Germany a group have been developing their own delay line interferometer, with 30 m long arms. In March 1989 the two groups carried out a 100 hour coincident data taking run. The run took place shortly after the reported optical observation of a 2 kHz pulsar in SN 1987a. The success of the run showed that long period coincidence runs are possible, maintaining the integrity of the data taken a high percentage of the time. The recorded data taken at both sites has been sent to Cardiff for full analysis. The purpose of this thesis is the analysis of the data obtained in Glasgow. In the rest of this chapter, I discuss the prototype in Glasgow in more detail, including the structure of the data produced during the run. I go on briefly to mention the Garching prototype, comparing it with the prototype in Glasgow and end by briefly discussing the practical implications of the 100 hour run.

### 4.1.1 The 10m Prototype.

The Glasgow prototype detector, shown in figure 4.1, consists of two perpendicular Fabry-Perot cavities, each comprising two mirrors optically contacted on to two test masses. These masses are suspended as pendula, by wires, this giving them a large degree of mechanical isolation. A beamsplitter is also suspended as a pendulum at the intersection of the lines through the two cavities. There are servo systems attached to each of the suspended masses within the system which allow their orientations to be controlled and would also serve to damp any unwanted low frequency oscillation of the masses. The light from an argon laser is polarised and then directed at the beamsplitter which transmits it equally into both cavities. The cavities and the beamsplitter are housed in a vacuum system maintained at a pressure of about $5 \times 10^{-4}$mbar in order to try to minimise the effect on the masses of the pressure due to any residual gas particles. The beams are bounced up and down the length of the cavities over 1000 times along the same lines without significant loss of energy because the mirrors are designed in such a way as to minimise the amount of light energy absorbed at each reflection. This dramatically increases the length of the light path and multiplies any relative displacements between the end masses by the number of bounces making any incoming gravitation radiation easier to detect. The light leaks from the cavities where it is then recombined to form an interference pattern which can be analysed by the use of a photodetector.

The laser beam is frequency locked to the *primary* cavity, —*ie* the cavity perpendicular to the direction of the laser— to keep the cavity resonating by ensuring that the length of the cavity is always a whole number of wavelengths of the light. In practical terms, as any change in the relative displacements of the masses within the cavity, due to impinging gravitation radiation or otherwise, is detected the information is fed back to the laser which alters its frequency accordingly. The signal transmitted by the feedback circuit is recorded as one of the relevant data streams and is called the *primary error point signal.*

The *secondary* cavity is locked to the laser in such a way as to maintain resonance within the cavity by altering its length when the frequency of the light changes. A signal measuring the response of the photodiode is continually fed back to the magnetic servo drive at the end of the cavity and as the beams go out of resonance the drive accordingly alters the length of the cavity. The signal transmitted to this feedback circuit is called the *secondary feedback signal.* Another data stream associated with this signal contains the *secondary error point signal*, that is searched for gravitational waves in this thesis. .

The whole process, by which the laser is locked to the primary cavity and the length of the secondary is locked to the laser, serves to double the effect that any incoming gravitational wave

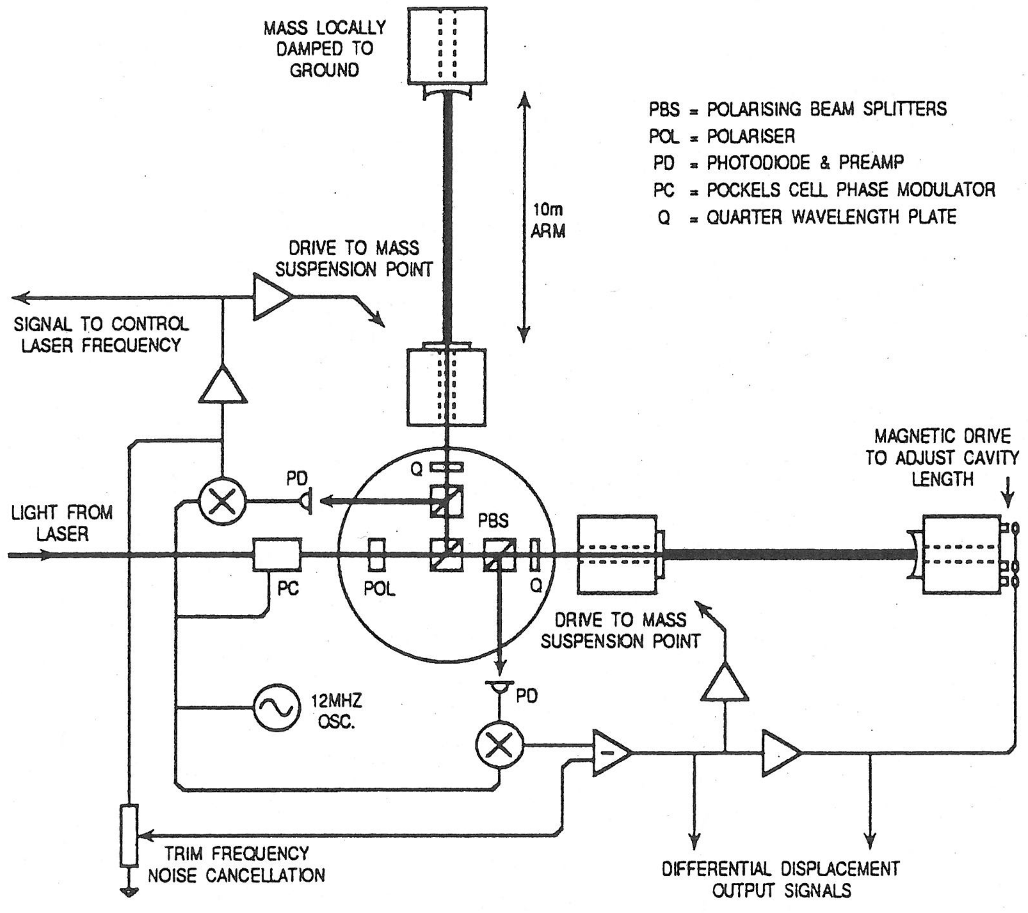

MASS LOCALLY DAMPED TO GROUND
10m ARM
PBS = POLARISING BEAM SPLITTERS
POL = POLARISER
PD = PHOTODIODE & PREAMP
PC = POCKELS CELL PHASE MODULATOR
Q = QUARTER WAVELENGTH PLATE
DRIVE TO MASS SUSPENSION POINT
SIGNAL TO CONTROL LASER FREQUENCY
MAGNETIC DRIVE TO ADJUST CAVITY LENGTH
LIGHT FROM LASER
PD
Q
PBS
PC
POL
Q
DRIVE TO MASS SUSPENSION POINT
12MHZ OSC.
PD
TRIM FREQUENCY NOISE CANCELLATION
DIFFERENTIAL DISPLACEMENT OUTPUT SIGNALS

Figure A.3: *Diagram showing the main optical components of the Glasgow prototype detector.*

Figure 4.1: Taken from Hough *et al*(1989)

may have on the system. This is due to the influence that the wave would have upon the lengths of the cavities. As the primary cavity expands the secondary contracts and vise versa, thus as the laser wavelength changed to accommodate the change in the primary cavity the signal transmitted to the drive at the end of the secondary would alter its length to follow the laser, in effect to cause the secondary to follow the contraction or expansion of the primary in opposition to the influence of the gravitational wave. So if at some point the change in the length of the primary cavity due to an incoming wave was $\approx x$, then the change in the length of the secondary should be $\approx -x$ but the drive would compel it to be $\approx x$ hence changing the length of the cavity by $\approx 2x$ *ie* double the effect of the gravitational wave.

### 4.1.2 Data taken by the Glasgow prototype

During the 100 hour coincident data taking run, mentioned above, sensors tracking various aspects of the detector and its surroundings produced many streams of data. These *channels* were interleaved producing *cycles* of 10 bytes, each cycle being sampled at 10 kHz. The analogue channels comprise the secondary error point data, which is the stream upon which the search for gravitational wave events was actually carried out, and also substantially more *housekeeping* data, which is the output of sensors that monitored the actual running of the detector and its peripherals and also the detectors environment. There is also one digital byte of housekeeping.

#### Secondary error point data.

This signal is the combination of two 12 bit signals each sampled at 10 kHz symmetrically in time to give an overall sampling rate of 20 kHz. This symmetry ensures that the time delay between the signals is the same in either direction. It is essentially just a stream of noise in which a gravitational wave may be lurking. As mentioned above this data is the feedback signal that adjusts the length of the secondary cavity and as such is likely to pick up noise from in and around the detector. It was hoped that the noise would be both white and Gaussian, however with so many potential noise sources it is inevitable that there would be a number of periods of bad noise, *ie* stretches where the noise statistics could not be confidently predicted.

Non white noise was produced both because the feedback system from which the secondary error point data was taken has a non-uniform frequency and also because filters were applied to suppress seismic noise at low frequencies and to cope with aliasing at high frequencies. To compensate for these factors, *calibration signals* were applied periodically through out the data taking run. This was achieved by applying a variation on the square wave function for 1.6384 sec every 210 sec. The data taken during this application should be deemed to be bad. These signals appear in frequency space as the teeth of a *comb* at odd multiples of 234.375Hz with the first comb at 234.375 Hz representing a displacement of $1.19 \times 10^{-13}$m and all the others a displacement of $1.19 \times 10^{-16}$m. Additional calibration combs were also applied after periods when the laser was out of lock allowing the data to be re-calibrated when lock had been re-established and the system was thought to be working properly. In that the combs represented *real* displacements of known amplitude, their application provided the means of attaching a real amplitude to any threshold crossing *event* picked up in the analysis. This is described in Chapter 5.

**Housekeeping data**

The housekeeping data consists of a number of streams taken from various sensors in and around the detector, the most notable of which, *ie* those considered in this thesis, I briefly describe below.

- The Microphone signal is the combined signal from 3 microphones, one at each end mass and the third at the center. It is an 8 bit signal sampled at 10 KHz.

- The Secondary feedback signal is obtained from the same cavity feedback coils that produce the secondary error point signal. Because of the feedback systems bandwidth it contains information below about 1kHz only. Normally this stream would be searched for low frequency gravitational waves, but not in this analysis as the calibration combs bring out the low frequency regions in the error point signal.

- The Primary cavity error point signal is input to the cavity feedback coil in the primary cavity. It is also a 12 bit signal sampled at 10kHz.

- The Multiplexed signal is another 12 bit signal sampled at 10KHz, this signal comprises 6 slow signals *multiplexed* together. This means only one of the slow signals is sampled during each cycle and only after 6 cycles is the same signal sampled again, giving each of these signals an effective sampling rate of 1667Hz. Of these 6 streams only 2 are analysed in this work.

  1. The Seismometer signal is taken from 4 seismometers in the system.

  2. The Visibility signals are slow signals monitoring the extent of the resonance in both the cavities. They contain no gravitational wave information. They may show the laser going

out of lock before it becomes very apparent in any of the other housekeeping streams. The secondary visibility signal only was examined in this analysis.

- Within the digital byte, each bit is given a different interpretation, either being set *OFF* or *ON*. For example one of the bits is set *ON* only when a calibration signal is applied. Another is set *ON* only at the start of every minute enabling the time to be updated continually whilst reading the data.

**Data format**

The data was written by an *Exabyte* machine to tape in *files* of blocks, each block containing 32768 bytes of information. This means that each block contains 3276 cycles, each of 10 bytes with the last 8 bytes not containing data. The last 4 bytes of each block were used as block count. This means that each block contains 6552 points of secondary error point data, 3276 points of all those streams sampled at 10000Hz and 3276/6 = 546 points of each of the multiplexed channels. Preceding the first data block on any of the tapes is a *filemark* and also 5 blocks of *header* information. A single filemark acts as a tag identifying the tape as one containing data at some point after the filemark. The header blocks contain, in the first block, an information text file and in the second, time and data information that can be used as a reference throughout the tape. The last 3 header blocks are empty. Should the tape contain more than one file of data then any file after the first will begin with a single filemark and then 5 header blocks. Two consecutive filemarks indicate the end of the tape. Further information about the data channels as well as the structuring of the cycles and the header blocks is given in Chapter 5.

### 4.1.3 The Garching prototype detector.

The Max-Planck-Institut für Astrophysik began work on laser interferometers in 1974, when a prototype delay line interferometer with arms of length 3m was constructed. The knowledge gained in the development of this prototype was put to good use when the 30m detector was commissioned in 1983, and built thereafter. The detector is shown diagrammatically in figure 4.2.

The major structural differences between the Glasgow and Garching detectors are mainly due to the differences between delay line and Fabry-Perot interferometers in general. In a delay line detector the light is bounced up and down the cavity along spatially *separate* paths. This necessitates bigger mirrors and also restricts the number of bounces within the cavity.

Against these disadvantages, delay line interferometers are far less complex to operate in that they do not need such precise control of the laser frequency and the arm length to maintain resonance

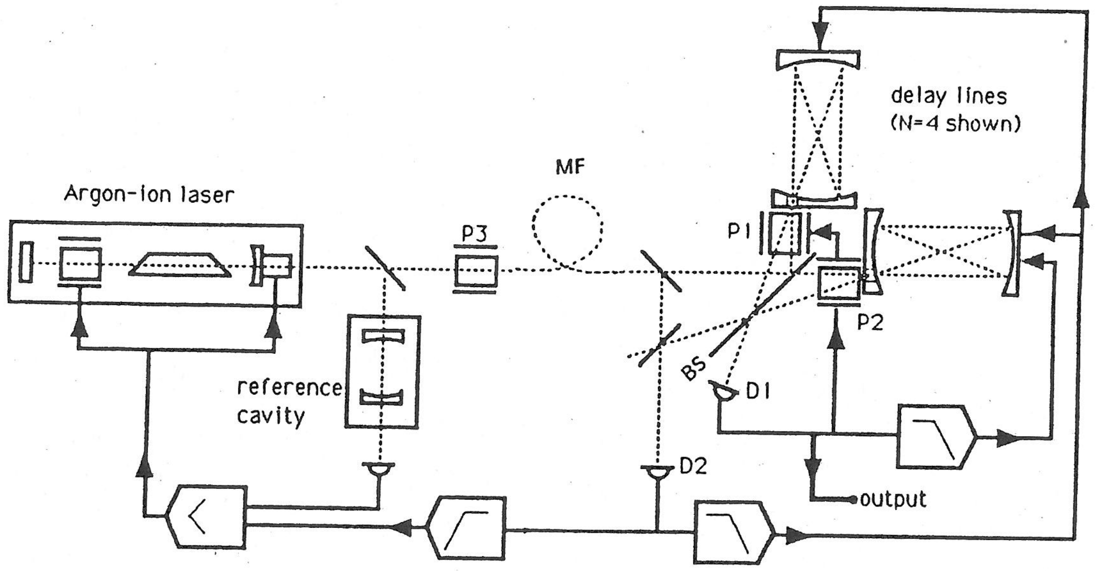


Figure A.1: *Schematic diagram of the Garching 30 metre prototype. The light from an argon ion laser is sent to the interferometer via a monomode glass fibre (MF). The Pockels cells P1, P2 serve for modulating the optical path as well as for maintaining the proper point of operation. The frequency of the light is stabilised in two stages: firstly with respect to a reference cavity, and then to the full path length of the interferometer, monitored by photodiode D2.*

Figure 4.2: Taken from Hough *et al*(1989)

in the cavity, they just need to maintain control over the relative phases of the light coming from the cavities.

The Garching data was sampled at 10 kHz, only half the frequency of the Glasgow data and there was also far less housekeeping data taken. For these reasons there was far less data taken in Garching in total and it was written to standard nine track tape rather than to video tape via the Exabyte machine.

### 4.1.4 The 100 hour run.

As mentioned in the introduction the 100 hour run demonstrated that both detectors could be run over sustained periods, taking information *reliably* a large portion of the time. The percentage of the total time when the arms are in lock, is called the *duty cycle.* During the runs, the duty cycle was a high 89% for the Glasgow detector and an extremely high 99% for the Garching detector. Both detectors were observed to measure *strains*, *ie* displacement noise level divided by arm length, of $\approx 10^{-19}/\sqrt{\text{Hz}}$ during the run, at the best sensitivity between 1500Hz and 2000Hz.

# Chapter 5

# DESCRIPTION OF SOFTWARE

## 5.1 Introduction

The software was written for the purpose of analysing the data produced by the Glasgow prototype gravitational wave detector in March 1989. The data was written on 28 Sony video cassettes. It was read by an *Exabyte-CTS* tape drive, (hence forth referred to as *Drive0*) and the results of the analysis were written to a similar drive (*Drive1*). The data was read directly from the tapes in Drive0 onto a board of 5 *transputers* housed in the Compaq 386/20 PC, an IBM compatible operating under MSDOS. The software was written in a version of Parallel FORTRAN utilising the *3L* Parallel FORTRAN Compiler, it was divided into five distinct parts, one running on each of the transputers. These routines ran independently and communicated by passing data back and forth to each other when necessary.

The analysis then was carried out in parallel on the 5 transputers. The overall structure was a 3 tier *pipeline.* This meant that 3 distinct groups of data would be at some stage of their analysis within the network at any one time. The *Root* transputer reads the raw data from the tapes in Drive0 and extracts various data streams from this raw data. The *housekeeping* data are held within the Root where the relevant statistics are calculated. The secondary error point data, on which the search for gravitational waves is actually carried out, is then passed to *Slave1*, the first of the slave transputers. Here the FFT of the secondary error point data is found and the data are *calibrated*, which serves to remove any frequency bias, and also go through a *weighting* procedure that optimises the data for the the extraction of possible signals within. Simultaneously the root is reading and unpacking a second data group. From Slave1 the data are sent on to the three remaining transputers, *Slave2*,*Slave3* and *Slave4*, where specific analysis is carried out. Slave2 conducts a search for wideband burst events such as supernovae, while Slave3 and Slave4 both employ matched filtering techniques to attempt to locate the characteristic *chirp* waveform that is thought to be emitted by coalescing binaries in the last few moments, before coalescence. Again these analysis routines were carried out simultaneously with those in the Root and Slave1 dealing with 2 other distinct data groups.

In this Chapter I describe all aspects of the software construction, firstly giving a brief description of the transputer and the ideas behind parallel processing, followed by an overview of the software design and then a more detailed description of all the routines.

### 5.1.1 The Transputer

#### Parallel processing

Over the last 40 years or so, the vast majority of the work done in the development of computing systems has been towards sequential processing *ie* the development of faster and faster processing chips that would carry out whole programming tasks by themselves.

In recent years however a lot of the attention has shifted towards the design and development of *parallel* processing systems, *ie* networks that split long processes up into smaller *tasks* and then farm them out to a number of smaller/slower/cheaper processors that are connected in parallel in such a way that data transfer between different processors is possible.

The best known parallel processor is the human brain in which neurons and synapses take the place of processing chips and wire links. Consideration of this gives some idea of the relative advantages and disadvantages of parallel processing. The brain is excellent at pattern recognition as is obvious from every day life, immediately recognizing objects and faces from visual data supplied by the eyes. It has however, proved very difficult to design and implement a mechanical system, of one or several processors, that in receiving the same kind of input from an optical scanner has anything like the pattern recognizing ability of the brain. Against this the brain is very poor at simple arithmetic calculations even when compared only to a single processing chip.

Single processing chips are not going to be able to be made faster forever. At some point, in the fairly near future, *quantum effects* will become insurmountable and the fastest possible chip will have been made. So in the long term, improved computational performance will only be achievable through the development of parallel systems. This is part of the reason why this prototype data analysis system was designed in parallel. There is no prohibitive reason why the analysis could not have been carried out by a fast single processor, however a parallel design was chosen, partly because the analysis lends itself to parallelism and partly because, in the long term, the final system is almost certain to be some kind of parallel system and its design could benefit from experience gained now.

### Transputers

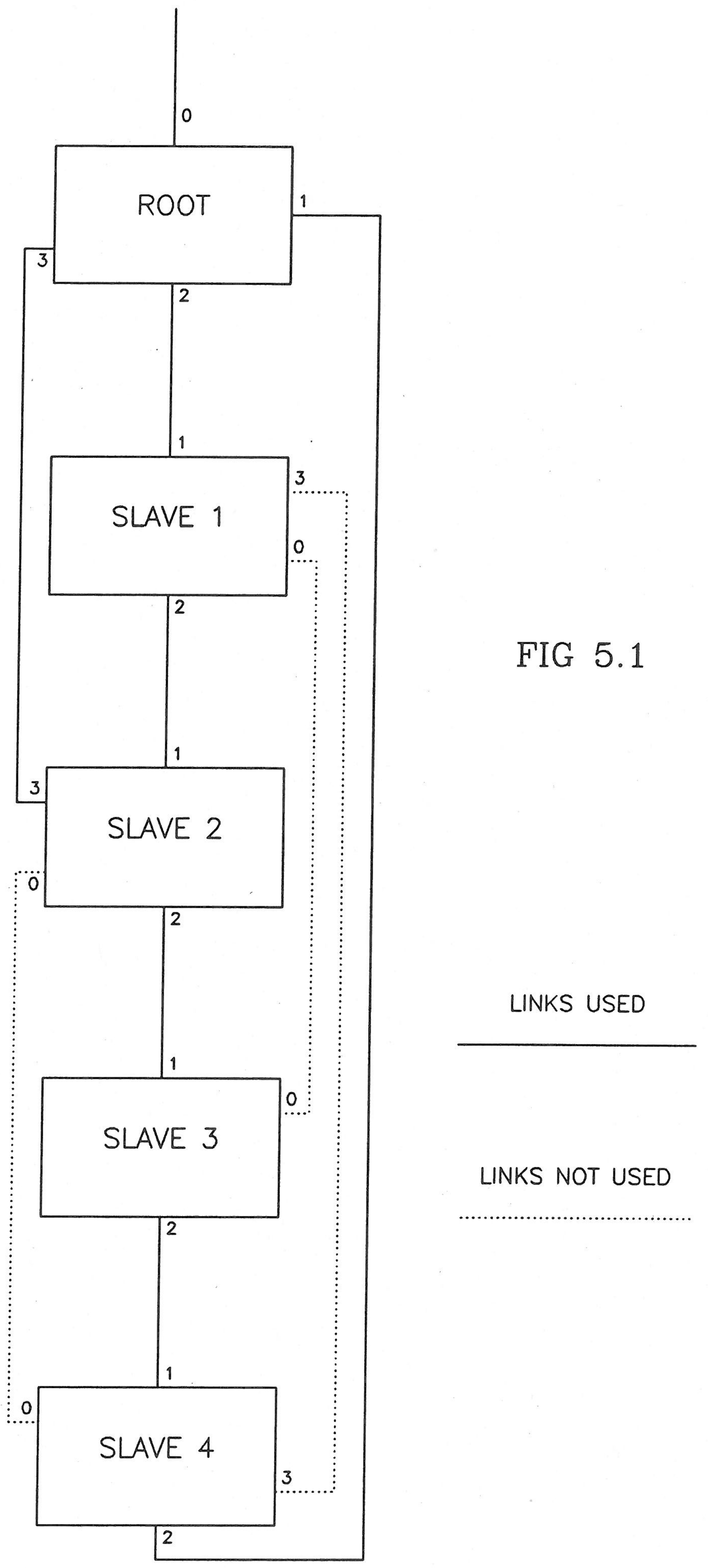
Transputers are manufactured by *Inmos*, they are basically single processing chips with a small amount of *fast* on-chip memory as well as on-chip input-output processors. They are connected together on a circuit board, each being individually associated with blocks of, rather more, external memory held elsewhere on the board. We used five T800 transputers, each having 4kB of fast on chip memory, four with $1MB$ of external memory and the fifth, the *Root* transputer, with 4MB of external memory. The board used was the *TBX05 Transputer Module Motherboard* manufactured by *Systems West.*

Each transputer has four *links* that can be used to connect it with other transputers or other devices. Each link is a two way connection enabling information to be passed to and from the transputer at a rate of 20Mbits/s. Transputers in a *network* can only be connected via these links, numbered 0,1,2 and 3, in certain preset ways. They form a *pipeline* with link 2 of any transputer being connected with link 1 of the following transputer and links 3 and 0 being connected with links 3 and 0 with either the last-but-one or next-but-one transputer. The 0 link in the Root transputer is always connected to the host processor. In our network the pipeline was circular which means the final transputer in the network is connected to the Root. The links and connections, both used and unused, are shown in Figure 5.1. The Root transputer is the only hardware link between the host processor and the network and therefore the only way off getting data in to the network is via the Root transputer.

The transputers were found to be able to operate at .455 Mflops and .5 Mflops for floating point multiplications and additions respectively and at 1.11 Mflops for both integer multiplication and addition. These figures, although somewhat lower than advertised by the manufactuers, mean that the transputers are still marginally faster than the 386/387 chip housed in the Compaq. Since FFT's play a large part in this work, Figure 5.2 shows the time taken for the $3Nlog_2(N)$ operations involved in finding FFT's of various data streams of length $N$.

### The Exabyte-CTS tape drive

The transputer reads data directly from, and write results directly to two Exabyte drives. The drive's full title is the *Exabyte EXB-8200 cartridge tape subsystem.* It is a "high performance, high capacity" device that utilizes " helical scan technology" to write data to an 8mm tape with a very high recording density and hence a very large data storage capacity. As mentioned earlier, the tapes used are the easily available V8 video cassettes and, depending on the means of writing, up to 2.2 Gbytes of information can be written to each cassette. The drives are connected to the Compaq by a *Datarace*



0
ROOT
1
3
2
1
SLAVE 1
3
0
2
1
3
SLAVE 2
0
2
1
0
SLAVE 3
2
1
0
SLAVE 4
3
2
LINKS USED
LINKS NOT USED

FIG 5.1

| *No of points* | *Time of FFT (sec)* |
|---|---|
| 128 | 0.007 |
| 256 | 0.015 |
| 512 | 0.030 |
| 1024 | 0.064 |
| 2048 | 0.1375 |
| 4096 | 0.300 |
| 8192 | 0.600 |
| 16384 | 1.333 |
| 32768 | 2.733 |
| 65536 | 5.800 |

Figure 5.2: Transputer benchmarking using FFT's.

*SCSI* (Small Computer Systems Interface) card housed in the back of the PC.

### 5.1.2 Overview

The Glasgow prototype detector was not thought to be sensitive enough for there to have been any realistic chance of finding any real events. For this reason, hardware constraints affecting the overall *time* of the analysis were allowed to become the main priority in the software construction rather than, for example, the number of filters used in the cross-correlation.

In parallelising the analysis procedure it was first necessary to consider it sequentially, to assess how best to divide it into parallel tasks. The adopted criterion had to take into account the available memory within each of the transputers and also, more importantly the time taken for each of the procedures in each of the transputers. Obviously for the sake of time efficiency, the duration of each of the tasks, in each of the transputers should be roughly the same and also the programs should be written in such a way as to minimise the amount of time that any of the transputers stands idle whilst waiting to either send or receive data from any other transputer. This was achieved by basically writing the program in serial and then testing each part separately for the time taken and other considerations. Figure 5.3 shows the coarse sequential break down of the program and also how it may be divided for parallelisation.

Bench marking routines established that the most time consuming single operation was the reading and unpacking of the data from the tape. This could only be done in the Root as only this transputer can read directly from Drive0 and also access the subroutine library that is used to

```
  BUILD UP A TEMPLATE OF FILTERS (FOR MATCHED FILTERING) IN ARRAY.
                                 |
                  READ RAW DATA FROM TAPE IN DRIVE 0.
                                 |
                   EXTRACT THE VARIOUS DATA STREAMS.
                                 |
                   COMPILE HOUSEKEEPING STATISTICS.
--------------------------------|-------------------------------------
                 FIND FFT OF SECONDARY ERROR POINT DATA.
                                 |
                         CALIBRATE THE DATA.
                                 |
                  DIVIDE THROUGH BY TRANSFER FUNCTION.
--------------------------------|-------------------------------------
  FIND THE CROSS-CORRELATION OF THE DATA WITH EACH FILTER IN TURN.
                                 |
                       CONDUCT AN EVENT SEARCH.
                                 |
                       BUILD UP A RESULTS ARRAY.
--------------------------------|-------------------------------------
                       FIND THE IFFT OF THE DATA.
                                 |
                       COMPILE DATA STATISTICS.
                                 |
                       CONDUCT AN EVENT SEARCH.
                                 |
                       BUILD UP A RESULTS ARRAY.
--------------------------------|-------------------------------------
                           COMBINE RESULTS.
                                 |
                        OUTPUT RESULTS TO TAPE.
FIGURE 5.3                       |
                        READ NEXT RAW DATA SET.
```

unpack the raw data. The volume of housekeeping data concerned was so great that it could only be held in the memory of the Root which has four times the memory of any of the other transputers, hence its statistical analysis was also confined to the Root. The duration of these processes dictate the minimum overall analysis time for the data and hence, for time efficiency, must be the only processes running on the Root.

The analysis required one FFT and also one IFFT for every group of secondary error point data. These groups each contained 32768 elements therefore, as shown in Figure 5.2, each FFT and IFFT took 2.733 seconds, making them relatively time consuming; hence it made sense to place them in different transputers. The total time taken for the FFT, together with the times for the calibration and weighting procedures that must follow the FFT, was found to be only a little less than the total time for the processes conducted in the Root, hence it was logical to place these processes exclusively in the first Slave transputer. This would receive the data from the Root and carry out its own functions as the Root read and unpacked more data.

When Slave1 completes its task the true analysis procedures begin. The wideband event search was basically just a threshold crossing time series search and hence required an IFFT of each whole 32768 point data stream, before the search could be carried out. This IFFT followed by the event search and then the accumulation of time series data statistics took just enough time to warrant sole occupation of Slave2.

The last two transputers were then left to conduct the search for chirps in the data. This search involved cross-correlating the secondary error point data with filters that were built up into a template at the initiation of the program and held in memory. The filters were stored in frequency space each being only 4096 points in length (representing only 0-1250 Hz), their construction is described later in this Chapter. The cross-correlation for each of these filters was calculated by multiplying the filter with the first 4096 points of the FFT of the secondary error point data supplied by Slave1, and then taking the IFFT of the resulting 4096 point array. A time series event search was then carried out. The total time for these processes determined how many cross-correlations could be done in total for each data stream. Obviously as the *minimum* total time taken for the analysis was dependent on the time taken for the procedure in the Root transputer, the duration of the tasks in both Slave3 and Slave4 were not allowed to exceed this time. Hence only 20 filters were used in total.

The analysis conducted in Slave2, Slave3 and Slave4, is simultaneous with another stretch of data being processed in Slave1 and a third in the Root. Following these event search procedures the results are passed back to the Root which writes them to the results tape in Drive1 before reading a fourth data group from the tape in Drive0.

A diagrammatic overview of the parallelisation and pipelining is given in Figure 5.4.

## 5.2 Comunication within the network.

All communications between transputers within the program are controlled by a small number of Parallel FORTRAN commands. These commands refer to addresses, each address being associated with one transputer. These connections and addresses are defined within the *Configuration* (`.CFG`) file.

### The Configuration file

This file describes both the hardware connections between the transputers and also the software connections that are used in the main body of the program.

In Figure 5.5 taken from the beginning of the `.CFG` file, the processors are declared and given names. The `host` is the 386 chip within the Compaq that houses the board. The `root` is the Root transputer and similarly `slave1` is Slave1 etc. The `wire` command defines the physical hardwire connection between the named transputers. The numbers in [ ] are the link numbers, from 0-3, and are defined as shown in Figure 5.1.

The next stage, shown in Figure 5.6, defines all the software `tasks` that run separately in the system. For each task the total number of connections to other tasks must also be defined. Each connection enabling data to be received is summed under `ins`, those through which data is sent, summed under `outs`.

The `place` command associates each of the tasks with its intended processor. The `afserver` task runs on the host processor. Upon execution it loads and then executes the other tasks on the appropriate transputers as defined by the `place` commands. Throughout the program run it acts as the software interface between the transputer board and the host system, performing any MSDOS functions that may be necessary, such as accessing a file stored on the hard disc of the PC.

The `connect` commands define all the data transfer connections between each of the tasks. A separate `connect` command has to be written for each data transfer direction, *ie* the last `connect` defines the connection that allows data to be transferred *from* task `filt5` *to* task `filt4`. The numbers in [ ] are again the link numbers used. For each task the number of links used must not exceed the number given in the `task` command. All the software connections must lie on one of the hardware connections defined at the start of the file. The hardware connections are bi-directional *ie* they support data transport either way, hence every hardware connection supports two software

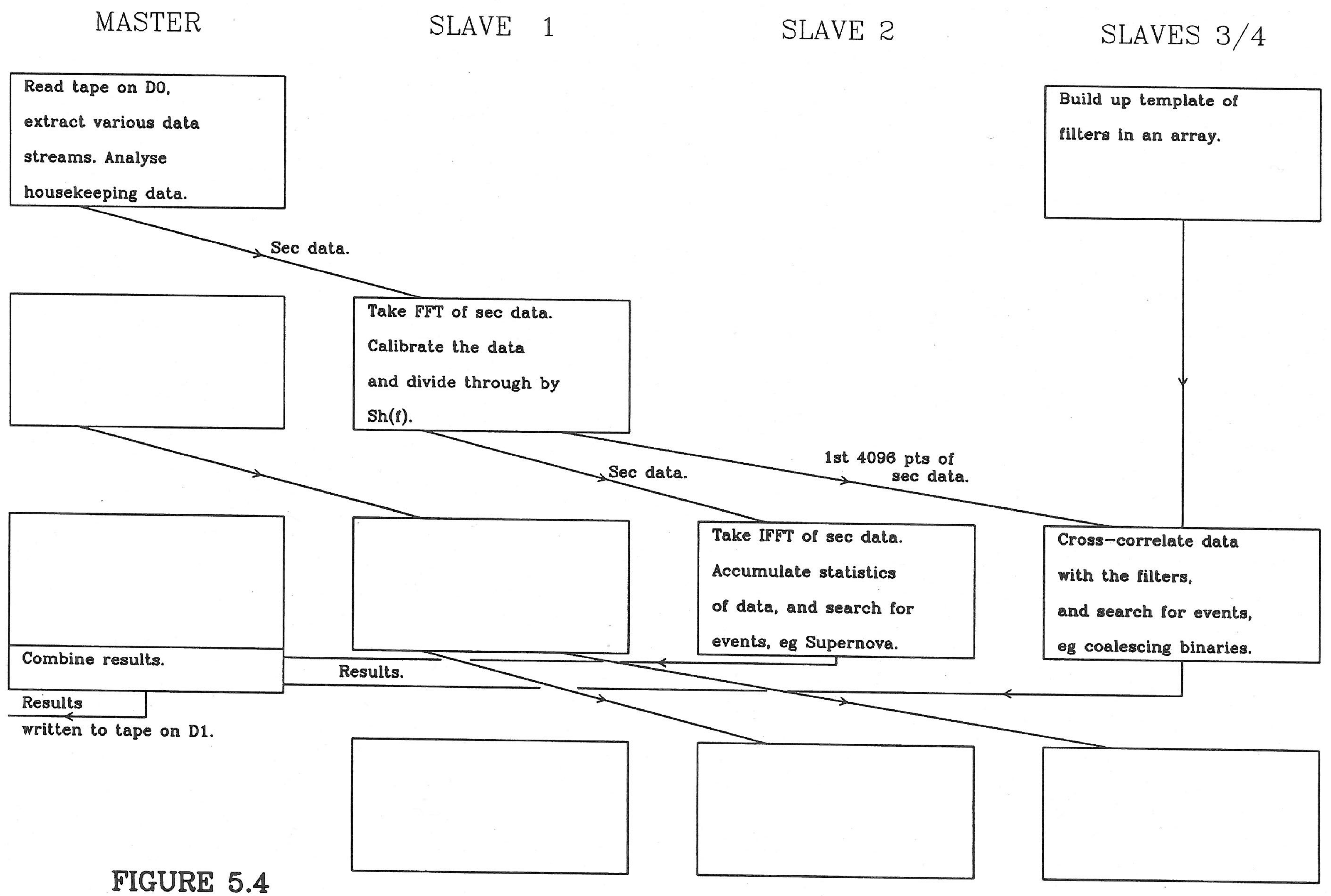

MASTER
SLAVE 1
SLAVE 2
SLAVES 3/4
Read tape on D0, extract various data streams. Analyse housekeeping data.
Build up template of filters in an array.
Sec data.
Take FFT of sec data. Calibrate the data and divide through by Sh(f).
Sec data.
1st 4096 pts of sec data.
Take IFFT of sec data. Accumulate statistics of data, and search for events, eg Supernova.
Cross-correlate data with the filters, and search for events, eg coalescing binaries.
Combine results.
Results.
Results written to tape on D1.

FIGURE 5.4

```
****HARDWARE****

processor host
processor root
wire ? host[0] root[0]
processor slave1
wire ? root[2] slave1[1]
processor slave2
wire ? slave1[2] slave2[1]
wire ? root[3] slave2[3]
processor slave3
wire ? slave2[2] slave3[1]
processor slave4
wire ? slave3[2] slave4[1]
wire ? slave4[2] root[1]
```

FIGURE 5.5

connections.

## Communication within Parallel FORTRAN

A full list of the commands within parallel FORTRAN that can expedite communications between separate tasks is given in the manual Parallel FORTRAN written by 3L Ltd that describes their compiler along with the communication commands specific to their version of Parallel FORTRAN. The commands are contained within several groups of files called *packages*, with each package of files serving a different general purpose. The files within any package are declared for use within any task program by the use of the `INCLUDE` command at the start of each of the programs. The general form of the command is `INCLUDE` '*package name*.`INC`' The only package used in this work is the `CHAN` package and so the command line at the start of each task is,

```
INCLUDE 'CHAN.INC'
```

Also towards the start of each of the tasks, before any data is transferred within the network, the *channel addresses* must be defined in accordance with the connect commands within the `.CFG`

```
****SOFTWARE****

task afserver ins=1 outs=1
task root               ins=5 outs=5
task secd1 ins=2 outs=2
task secd2 ins=3 outs=3
task filt4 ins=2 outs=2
task filt5 ins=2 outs=2

place afserver host
place root root
place secd1 slave1
place secd2 slave2
place filt4 slave3
place filt5 slave4

connect ? root[1] afserver[0]
connect ? afserver[0] root[1]
connect ? root[2] secd1[0]
connect ? secd1[0] root[2]
connect ? secd1[1] secd2[0]
connect ? secd2[0] secd1[1]
connect ? secd2[1] root[3]
connect ? root[3] secd2[1]
connect ? secd2[2] filt4[0]
connect ? filt4[0] secd2[2]
connect ? filt5[1] root[4]
connect ? root[4] filt5[1]
connect ? filt4[1] filt5[0]
connect ? filt5[0] filt4[1]
```

FIGURE 5.6

file. These are integer quantities and have the following form,

```
      INCHAN = F77_CHAN_IN_PORT (2)
      OUTCHAN = F77_CHAN_OUT_PORT (2)
```

The number in the brackets is that referred to in the **connect** commands listed in the .CFG file given above.INCHAN refers to the software channel in task *Root* that receives data from task *Secd1* and OUTCHAN, (initially defined as an integer), refers to the channel in task *Root* that sends data to task *Secd1.*

To send or receive one four byte WORD, (*ie* any single precision real or integer), the commands are of form,

```
CALL F77_CHAN_IN_WORD (WORD, INCHAN)
CALL F77_CHAN_OUT_WORD (WORD, OUTCHAN)
```

The top line asks for a four byte WORD from the address defined by INCHAN, and the bottom line tries to send WORD to the address defined by OUTCHAN.

The format is similar for longer stretches of data called MESSAGES. Here the length in bytes of the message has also to be given. So the commands to give or receive the message DATA of length LENGTH bytes are,

```
CALL F77_CHAN_IN_MESSAGE (LENGTH, DATA, INCHAN)
CALL F77_CHAN_OUT_MESSAGE (LENGTH, DATA, OUTCHAN)
```

These are the only data transfer commands used.

## 5.3 ROOT.F77

### Introduction

This program operates within the Root. Its main purpose to act as a liaison between the transputers that actually conduct the gravitational wave search and the outside world which, in this instance, was the tape drives, the hard disc of the PC and the monitor screen. It reads data from the Glasgow tapes in Drive0, it then *unpacks* the data *ie* splits the raw data up into its various constituent streams and sends the *Secondary error point* data on into the rest of transputer network for analysis. It compiles and stores statistics calculated from the various *Housekeeping* streams which are then merged with the received results from the other transputers, which it then outputs to a results tape in Drive1. It's designed to incorporate these features within a 3 tier structure which means that all the housekeeping data associated with 3 separate groups has to be contained within the memory of the Root at any one time. For this reason the Root has 4MBytes of memory, four times the memory of any of the others transputers. Also the results from any one group were returned to this task after the following but one group had been read from the tape so the passage of each of the groups through *Root* had to be monitored, with the commands to receive data from the other tasks positioned within *Root* so as to optimise the transfer of data. Below I describe the main features of the routine, additional details can be found in the program listing. This task along with the others is listed in full in Appendix 3. The lines in this Appendix are numbered and I refer to them occasionally in certain parts of the rest of this Chapter.

### 5.3.1 Program details.

#### Comunication with the tape drive and extraction of data streams

### 5.3.2 Software

J.R.Shuttleworth (Dept of Physics,UWCC) wrote a file server using Parallel FORTRAN source codes, supplied on request by 3L and partially based on software written by N.L.Mackenzie (Dept of Physics, Glasgow University) that enabled data to be read from and written to the two CTS Drives by commands contained in software running on the Root transputer. These routines are held in the precompiled subroutine library `EXLBNOTG.BIN`. When reading from Drive0, data is transferred in blocks of 32 Kbytes, stored momentarily in the memory of the Compaq and then passed via the hardwire link into the memory of the Root transputer. Here the raw data is *unpacked* into its various constituents data streams by calling a number of subroutines written in C, again by Shuttleworth to some extent based on routines written by N.L.Mackenzie.

Once unpacked the data is passed around the transputer network and analysed by routines written in the *3L* version of *Parallel FORTRAN.* Parallel FORTRAN is basically FORTRAN 77 with additional commands to expedite the transfer of data between transputers. The software was written for each transputer separately and contained information about the links to other transputers within its structure. Each `.F77` source code is individually compiled to a `.BIN` object file and then linked to a .B4 executable file. The configuration (.CFG) file describes the various hardware and software connections and under the CONFIG command, produces an *application* (.APP) file .

The application file is an executable file run in the host processor. When executed within the host, it installs and executes the .B4 files in the workspaces of the appropriate transputers. All 5 transputers run their .B4 files simultaneously, in parallel, with the only periods of inactivity being due to any one transputer having to wait to give or receive data from any other.

At the execution of this program commands are sent that initialize the tape drives, carry out diagnostic procedures and make ready Drive0 to read the tape. These commands are sent via routines contained in EXLBNOTG.BIN, called within the program.

The first routine called is SETUPOK,

```
      CALL SETUPOK (GOOD)
```

The logical variable GOOD returned is set to .TRUE. if the SCSI bus is operating properly and .FALSE. if not. TAPEOK returns .GOOD. set to .TRUE. only after successfully confirming that there is a tape in the drive, loading the tape, moving over the first filemark and setting the data transfer size to 32 kbytes. Its written in the form,

```
      CALL TAPEOK (0,GOOD)
```

where 0 (or 1) is the drive number. The last routine needed before the analysis can begin is HEADER which attempts to read the header blocks, *ie* the first five blocks after the intial filemark.

```
      CALL HEADER (0,INDATA,GOOD,EOTAPE,SVHEAD,'NULL',ITIME)
```

The only returned arguments relevant to the task, are the logical variable GOOD, set to .TRUE. if the routine is successful and the 6 element array ITIME into which the date and time at the start of the experiment are loaded. Other routines sometimes called are listed below. See program listing for more details.

```
      CALL SPACE(0,NTMIN,NUMOUT,RES,FILMRK) ! Attempts to space over NTMIN
                                            ! blks.
      CALL WTFMARK (1,GOOD)                 ! Attempts to write filemark on 1.
      CALL RWOUND (1,GOOD)                  ! Attempts to rewind tape in 1.
      CALL EJECT (0,GOOD)                   ! Attempts to eject tape in 0.
```

As explained in Chapter 4, the various data streams produced during the detector run were interleaved into one data stream and written to exabyte tape. The routine RDTAPE used within this task to read this stream from the tape, straight into the memory of the Root is called in the following way,

```
    CALL RDTAPE(0,INDATA,NBLK,GOOD,FILMRK).
```

Here 0 is the drive number, INDATA the array to which the data are read, and NBLK the number of the block on the tape used. GOOD and FILMRK are logical variables that are returned with GOOD set to .TRUE. and FILMRK set to .FALSE. if the routine is successful. INDATA is a stream of 1 byte digits which can only be read within Parallel FORTRAN as a character array and therefore INDATA is an array of 32768 1 byte character elements.

Almost all the remaining routines used all take INDATA as an argument and return one of the signals interleaved within INDATA.

```
    CALL GTSERR(INDATA,SECERR)    !  secondary error point data  6552 pts
    CALL GTMICR(INDATA,MICRO)     !  microphone signal  3276 pts
    CALL GTPERR(INDATA,PRIM)      !  primary error point data  3276 pts
    CALL GTSEIS(INDATA,SEIS)      !  seismic signal  512 pts
    CALL GTSVIS(INDATA,SECV)      !  secondary visibility signal  512 pts
    CALL GTSFED(INDATA,SFED)      !  secondary feedback signal  3276 pts
    CALL GTDIGT(INDATA,DIGITS)    !  digital data  3276 pts
```

The digital data is again a 1 byte character stream but the rest is all 4 byte reals.

There are two further logical functions used that when passed an element from the character array DIGITS, interpret it as string of digital bytes and extract certain bits from whose setting, on or off, the presence of a minute mark or a calibration comb can be inferred and, if necessary, looked for.

```
    IF (CALIB(DIGITS(I))) THEN
      WRITE(6,*) 'CALIBRATION COMB PRESENT IN DATA.'
    ENDIF

    IF (MINMRK(DIGITS(I))) THEN
      WRITE(6,*) 'MINUTE MARK PRESENT AT THIS POINT.'
    ENDIF
```

Finally when trying to write data to the tape in Drive1 the routine WTTAPE is used.

```
CALL WTTAPE(1,CHRES,GOOD)
```

CHRES is the 32768 element character array written to the tape in Drive1. GOOD is again set to .TRUE. if the routine is successful.

## Timing within the analysis

The date and time at the start of the experiment are given in the header blocks in the array ITIME. This is not however the time at the start of the first actual data block so before the commencement of the actual analysis this time was found by moving into the real data and looking for the first minute mark. This was achieved by moving through the blocks, extracting the digital array and then giving the function MINMRK, each element of the digital array as an argument, one by one until MINMRK returned .TRUE..

The real data starts about 10 seconds after the header, so by using the time at the start of the experiment it is possible to judge roughly where the first minute mark occurs. Should this point be some way into the data then the search for the first minute mark is started after the start of the actual data but definitely before the first minute mark. This procedure is carried out between lines 238 and 296.

If the first minute mark is found in block NBLK (which is 0 at the start) and then at point JNBLK within NBLK, the time, $T_{minmrk}$, taken since the start of the actual data is,

$$T_{minmrk} = \mathtt{NBLK} \times 0.3276 + \mathtt{JNBLK} \times 0.0001 .$$

Each block contains 3276 digital points sampled at 10 khz. One block therefore represents 0.3276 seconds and each element is 0.0001 second. The minute marks are set at the start of each minute when ITIME(6) and possibly ITIME(5) and ITIME(4) as well are updated to the start of the actual data with ITIME(6) given by,

$$\mathtt{ITIME(6)} = 60 - T_{minmrk} = 60 - \mathtt{NBLK} \times 0.3276 - \mathtt{JNBLK} \times 0.0001 .$$

When the change of minute occurs after the header but before the start of the data proper, then the minute mark representing the next minute is found and the start then found accordingly.

The array ITIME is then updated for every subsequent minute mark found. There are 183.15 blocks in a minute's data, so the search for minute marks in the data is not begun until 183 blocks after the start of the block that contained the previous mark. This minimised the time taken looking for the minute marks with an average of only 5 blocks in every 183 being searched over the whole.

Finding a minute mark where it should be allows confirmation of the timing within the analysis and allows reliably accurate times to be attached to each of the events. For each minute mark its position is recorded in F1BLK, which is a real number that gives the position of the mark to the nearest 1000th of a block, *ie* to the nearest 0.0003276th of a second. If an event has a position EBLK, the time in seconds since the last minute mark is $T_{\mathrm{EBLK}}$ which is given by,

$$T_{\mathrm{TBLK}} = (\mathtt{EBLK} - \mathtt{F1BLK}) \times 0.3276. \tag{5.1}$$

This allows the time of the event in hours minutes and seconds to be found.

### 5.3.3 Main body of program.

Each analysis group consists of 32768 or $2^{15}$ secondary error data points. This represents 5 blocks of 6552 points each plus 8 additional points set to zero to achieve the power of 2. The program is dominated by the contents of a loop, that reads data from the tape, builds up an analysis group and sends it on to the other tasks for analysis. In each group only the first 26208 points (or 4 blocks) are analysed hence in each loop only 4 new blocks are read and the final block from the previous group is placed at the start of the next. This means also that 5 blocks must be read initially so one is read before the start of the first loop. This is necessary to prevent the *wraparound* effect in the discrete correlations as described in Chapter 3.

With each group there is more than twice as much associated housekeeping data that must be stored and statistically analysed. When the results are returned to *Root* they are combined with the appropriate housekeeping data and written to the results tape in Drive1. In the period between a group being read from the tape in Drive0 and its results being written to tape in Drive1, two more groups are read from the tape in Drive0. This housekeeping data is also stored hence each group of housekeeping data must be tagged in someway to associate it with a particular results group. This is achieved by use of the three tier structure mentioned earlier.

Initially 1 block of data is read from the Glasgow results tape into the array INDATA. This array is then passed to the routines listed above that produce the Secondary error point signal as well as the housekeeping streams. The housekeeping is stored and the Secondary error point data is passed on to the next task in *Slave1*.

At this stage the major loop begins. In each loop 4 blocks are read from the tape. For each one the Secondary error point signal as well as the housekeeping streams are extracted. The Secondary error point data is sent to the next task and the housekeeping stored. The separate housekeeping arrays are stored by use of the *equivalence* statement in single large arrays. The equivalence statement when applied to 2 arrays compels them to occupy the same memory areas. This is utilised in this task to build the small separate housekeeping arrays produced for each block into single larger arrays. This is shown for the microphone signal in Figure 5.7.

```
     EQUIVALENCE (MICROP(1),MICRO(1)),(MICROP(3277),MICRO1(1)),
    +(MICROP(6553),MICRO2(1)),(MICROP(9829),MICRO3(1)),
    +(MICROP(13105),MICRO4(1)),(MICROP(16381),MICRO5(1)),
    +(MICROP(19657),MICRO6(1)),(MICROP(22933),MICRO7(1)),
    +(MICROP(26209),MICRO8(1)),(MICROP(29485),MICRO9(1)),
    +(MICROP(32761),MICRO10(1)),(MICROP(36037),MICRO11(1))
```

FIGURE 5.7

MICROP is an array of 39312 elements and MICRO, MICRO1 etc are all of length 3276 elements. Initially, before the start of the loop, the routine GTMICR is supplied with INDATA from the first block and MICRO is returned. MICRO, although in itself only 3276 elements in length, immediately occupies the first 3276 elements of MICROP as well. Within the first loop the next 4 blocks are read and MICRO1, MICRO2, MICRO3 and MICRO4 are found. As can be seen above MICRO1 also occupies 3276 elements of MICROP from 3277 to 6552. Similarly for MICRO2, MICRO3 and MICRO4, the first 5 blocks worth of MICROP are now filled, up to element 16380. Analogously the same is true for the other housekeeping streams although the multiplexed arrays are 6 times shorter.

This structuring forms the basis of the 3 tier structure mentioned earlier because in the second loop MICRO5, MICRO6, MICRO7 and MICRO8 are found and these occupy elements 16381 to 29484 of MICROP. In the third loop MICRO9, MICRO10, MICRO11 and MICRO12 are found but only the first 3 are included in the equivalence statement and these occupy the remaining elements of MICROP. Again this is also true for the other housekeeping steams and therefore all the housekeeping data associated with 3 distinct groups of data is stored in several large arrays. Each of the tiers has an associated flag number, 1 for the first loop, 2 for the second and 3 for the third. The results data from the analysis of the secondary error point data taken in the first loop *ie* when the flag is 1, is returned at the end of the third loop when the flag is set at 3. This will always be the case thus when data is returned and the flag is set at 3 it is associated with the first third of the large housekeeping data arrays. Similarly when data is returned and the flag is set to 1 or 2 then it is associated with the second and third parts of the housekeeping arrays respectively.

At the end of the third tier when the results have been combined with the housekeeping data and output to tape the flag is then reset to 1 and the 13th small housekeeping array (MICRO12 in the case of the microphone signal) is written in a loop into the first blocks worth of the larger arrays (the first 3276 points of MICROP) becoming the first block of the next group. As the next 4 blocks are read the housekeeping data extracted overwrites the larger arrays. More results are returned, combined with the appropriate housekeeping data and written to tape. The flag is reset from 1 to 2 and the next loop begins etc.

The blocks of Secondary error point data sent on to *Slave1* have one extra point appended to them. This point is array element `SECER1(6553)` (or `SECER2` or 3 or 4). Usually this extra point contains no useful information in which case it is set to 0, but it can tell the other tasks to look for calibration combs or to return statistics. It is also used to halt the tasks in the other transputers.

After the extraction of each block the digital arrays are searched for indications of the presence of calibration combs in the secondary error point data. If found `SECER1` is given the value 1.

The analysis routines running the slaves produce histogram statistics of the data that the root writes to disc occasionally to prevent their complete loss should the program crash for some reason. The signal to return these statistics to the root is `SECER1` equal to 2.

When the program has completed its allotted number of loops `SECER1` is set to -1 which triggers the termination procedures in each of the slaves.

The housekeeping statistics are calculated after the last block from each group is read from the tape, and before the results from the previous but one group are retuned to the Root. The means and standard deviations for each of the relevant housekeeping streams are found except for the secondary visibility signal which acts as a long period diagnostic for the system for which reason only its maximum value for each group is found. When the event search results are returned to the Root, the deviation from the group mean for each of the housekeeping streams at those points are found and divided by the calculated standard deviation for that stream. The ratio's obtained can be examined to establish if any of the streams were possibly the cause of the event. The event information is combined with this information and output to the results tape in drive1. The structure of each results block is given in more detail in Appendix 2.

## 5.4 SECD1.F77

### 5.4.1 Introduction

This task is located in the first slave transputer, Slave1. It receives data in blocks from the root which it builds into groups of five blocks. The position in the 3 tier structure in task *Root* from which the data is sent is irrelevant in this task, as it treats all groups in the same way. Within this task several procedures are carried out that compensate for any 'bad' noise due to either nonwhite noise produced by the components of the system or frequency bias in the actual detector. These adjustments are made in frequency space. Hence firstly the Discrete Fourier Transform, of 32768 points (5 blocks plus 8 points set to 0), is found through the use of two Fast Fourier Transform algorithms, REALFT and FOUR1, taken from Numerical Recipes by Press et al (1986).

The data was then calibrated according to *combs* that were periodically applied to the system and appear in the data. The calibration data is recalculated whenever a new comb is found. This calibration serves to remove any frequency bias within the data. The calibration routines are CALCOM and UPDCOM. The presence of a comb in the stream is noted in the first element of a element array SDT, which is essentially used just to pass useful numbers through the system.

The second element of SDT is the calculated mean of the current group of raw data before it is altered in any way.

After the calibrated data is returned to the main program it is immediately sent to the routine WEIGHT in which the extent of the influence of the bad noise in the data is reduced and also statistical data used to calculate event amplitudes is found and written to the elements 3 and 4 of the array SDT.

Finally the calibrated and weighted FFT data is sent on to the remaining 3 transputers for the actual event search procedures after which the routine asks for the next four blocks of time series data.

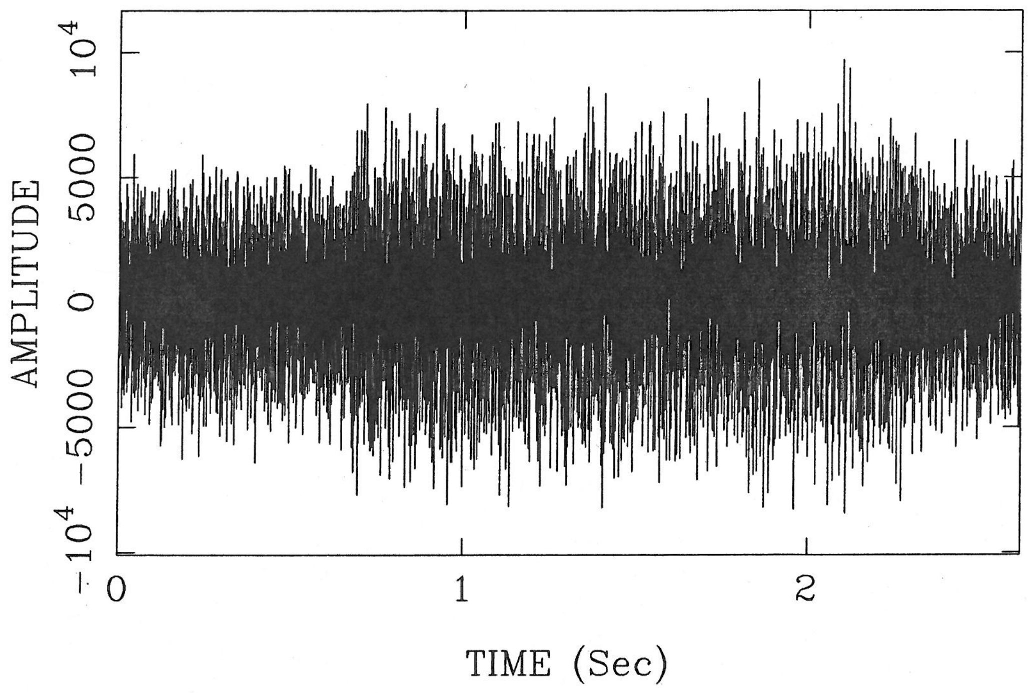
### 5.4.2 Main body of task

Initially it received 5 blocks from the task *Root* which it concatenated into a 32768 element array using equivalence statements (lines 8-10). This array was then the basic analysis *group* upon which this task performed several procedures after which it passed the altered group onto the remaining tasks running in the other transputers.

The fifth block initially received, was copied into an array which in its unaltered state became the first block of the second group which was then completed by the next 4 blocks received from the root (lines 165-166 and 210-211). The process is then repeated through all the data, the last block of the previous group becoming the first block of the next, regardless of the current position in the tier. No statistical analysis is actually carried out on the final block of any group hence by putting it at the start of the next group it is not analysed twice.

The 6553rd element of each of the blocks is examined for a relevant message. If it was found to be set to either -1 or 5 the instruction were to either terminate all the tasks or return histogram data respectively. In either case the same information is conveyed onwards to the rest of the tasks by setting point 32761 of the group to the same values at the end of the task. If the 6553rd element

TIME SERIES DATA CONTAINING COMB.



LOG(POWER SPECTRUM OF ABOVE)

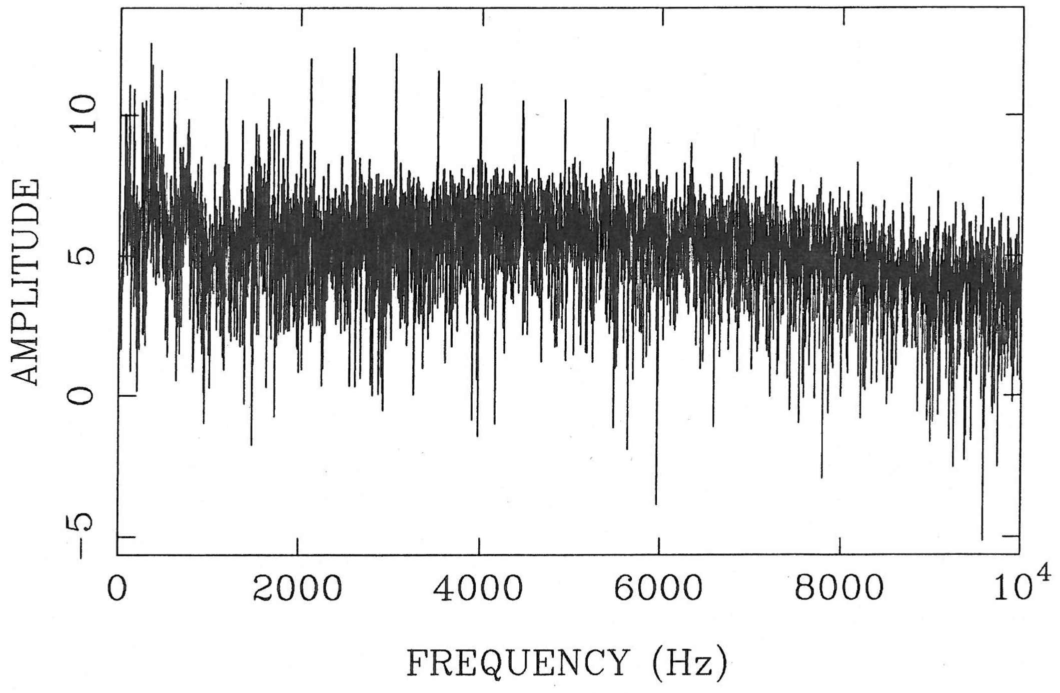


FIGURE 5.8

is set to 1 then a new calibration comb has been found to start somewhere in that block. The very *next* block is then sent to the subroutine `CALCOM` where, as described below, the relevant arrays are updated. This is the first block that is sure to have a comb applied over all its total length as the comb is applied for 1.6384 seconds which is equivalent to 5 blocks.

After the whole group is formed, its FFT is found prior to it being passed to the subroutines described below. Following this the next four blocks are received from *Root*.

### 5.4.3 Calibration

There are two subroutines within *secd1* that are concerned with the calibration of the data. The first, `CALCOM` (lines 220-252), is only called every time the presence of a comb is picked up in the time series. The routine is sent a block of data over the whole of which a comb was known to have been applied. Only the first 4096 points of the block are used to provide the FFT from which the comb is extracted. The teeth of the comb are at fixed frequencies $f_n$ given by,

$$f_n = 234.375 + 468.75(n-1), \qquad n = 1, 2, ...., 21.$$

These frequencies fall exactly on points given in the discreetly sampled FFT. Hence the comb's amplitudes can be exactly found. Apart from the first tooth of the comb, which is 1000 times larger, all the other teeth of the comb produce a displacement in the detector of $1.19 \times 10^{-16}$ m. Figure 5.8 shows a section of time series data with a comb applied in the central region and also the log of the power spectrum of the time series showing the combs.

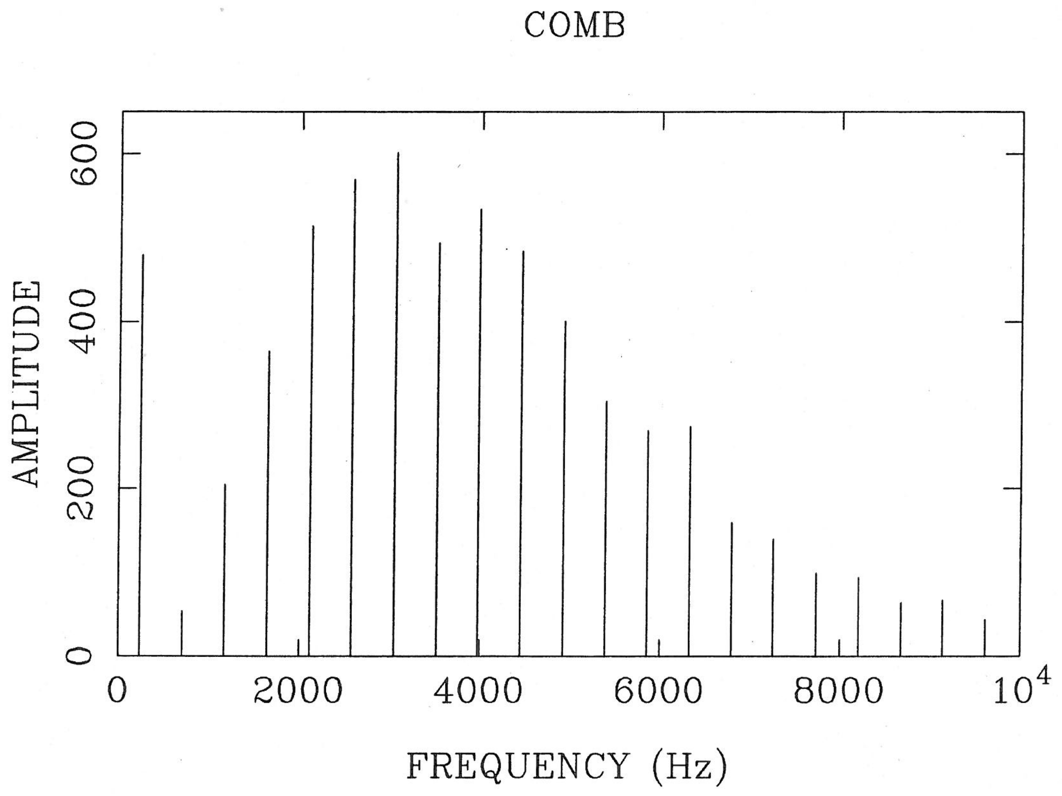
For the comb an *inverse* comb is then calculated whose teeth have amplitudes such that when any of its teeth are multiplied by the equivalent tooth from the original comb, the original comb's true displacement is produced. So if $\texttt{COMB}(n)$ is the height of the $n$'th tooth, the equivalent tooth from the inverse comb, $\texttt{ICOMB}(n)$ is given by

$$\texttt{ICOMB}(n) = \frac{1.19 \times 10^{-16}}{\texttt{COMB}(n)} \qquad n = 2, 21 \tag{5.2}$$

$$= \frac{1.19 \times 10^{-19}}{\texttt{COMB}(n)} \qquad n = 1. \tag{5.3}$$

The comb and its inverse comb are shown in Figure 5.9. The inverse comb is stored in the real array `ICOMB`. The points on the inverse comb between the calculated peaks are found by linear interpolation. To enable these points to be found the slopes of the lines between the combs are found and stored in the 20 element array `SLOPE`. The slope of the line between inverse combs $\texttt{ICOMB}(n)$ and $\texttt{ICOMB}(n+1)$ is written to $\texttt{SLOPE}(n)$ which is given by,

$$\texttt{SLOPE}(n) = (\texttt{ICOMB}(n+1) - \texttt{ICOMB}(n))/768\,, \quad n = 1, 20\,. \tag{5.4}$$


COMB
AMPLITUDE
0
200
400
600
0
2000
4000
6000
8000
$10^4$
FREQUENCY (Hz)

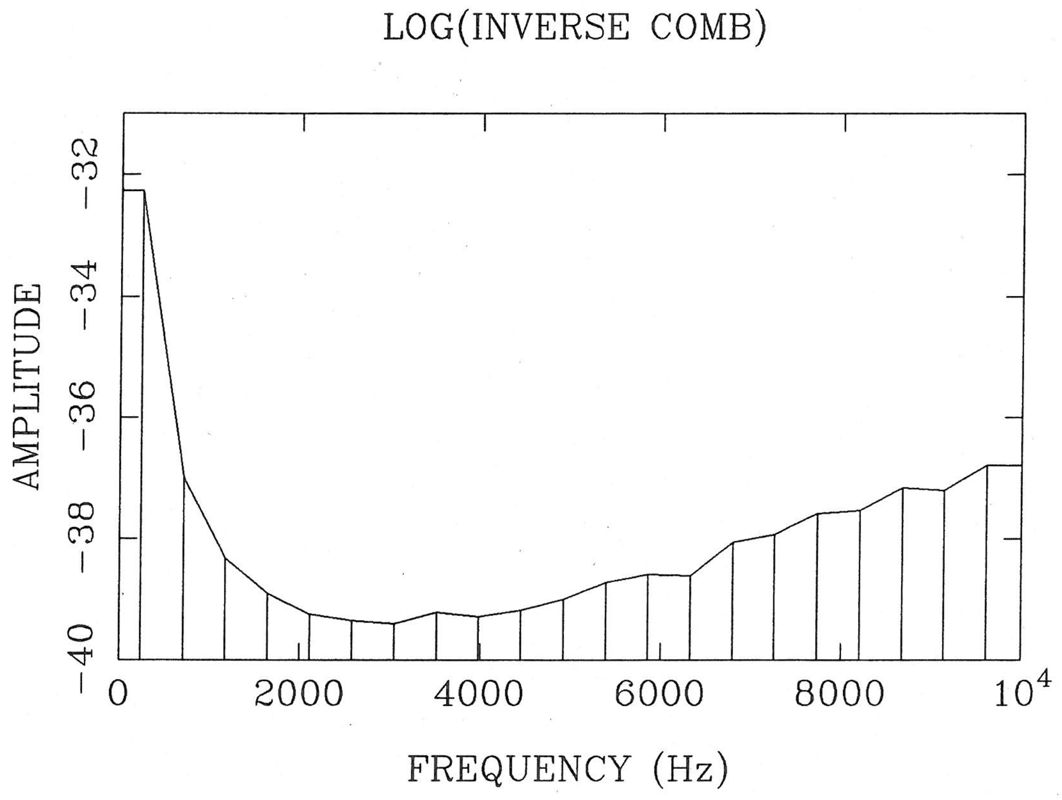

LOG(INVERSE COMB)
AMPLITUDE
−40
−38
−36
−34
−32
0
2000
4000
6000
8000
$10^4$
FREQUENCY (Hz)

FIGURE 5.9

In the above equation, 768 is used because it is the number of complex points between the combs when 16384 complex points are considered in total. These arrays are then passed back to the main program.

The second subroutine directly associated with the combs is UPDCOM (line 257-290). The whole 32768 data points of the Fourier transform of each group are sent to this subroutine along with the arrays ICOMB and SLOPE that are used to calibrate the data. The calibration is achieved by multiplying each point in the data by the calculated height of the curve through the inverse comb. This height is calculated using ICOMB and SLOPE. If ICOMB($n$) is the height of inverse comb $n$ and SLOPE($n$) the slope of the line between comb $n$ and $n+1$ then at any complex point $j$ between the combs the heights $\mathrm{IH}_{real(j)}$ and $\mathrm{IH}_{imag(j)}$ are given by,

$$\mathrm{IH}_{real(j)} = \mathtt{ICOMB}(n) + (2j) \times \mathtt{SLOPE}(n), \quad \text{and} \tag{5.5}$$

$$\mathrm{IH}_{imag(j)} = \mathtt{ICOMB}(n) + (2j-1) \times \mathtt{SLOPE}(n), \quad j = 1, 2, ..., 384. \tag{5.6}$$

For the region before the first tooth, IH is set to ICOMB(1) and for the region after the last tooth, IH is set to ICOMB(21).

The data is calibrated by multiplying each complex point by the calculated height of the inverse comb at that point. This is done for all regions of data between two combs separately. Hence at some point $j$ of complex heights $\mathrm{H}_{real(j)}$ and $\mathrm{H}_{imag(j)}$ between two combs where $j = 0$ at the previous comb (or, if it is the start of the data at the first data point) the calibrated heights, $\mathrm{C}_{real(j)}$ and $\mathrm{C}_{imag(j)}$ are given by,

$$\mathrm{C}_{real(j)} = \mathrm{IH}_{real(j)} \times \mathrm{H}_{real(j)}, \quad \text{and} \tag{5.7}$$

$$\mathrm{C}_{imag(j)} = \mathrm{IH}_{imag(j)} \times \mathrm{H}_{imag(j)}, \quad j = 1, 2, ..., 384. \tag{5.8}$$

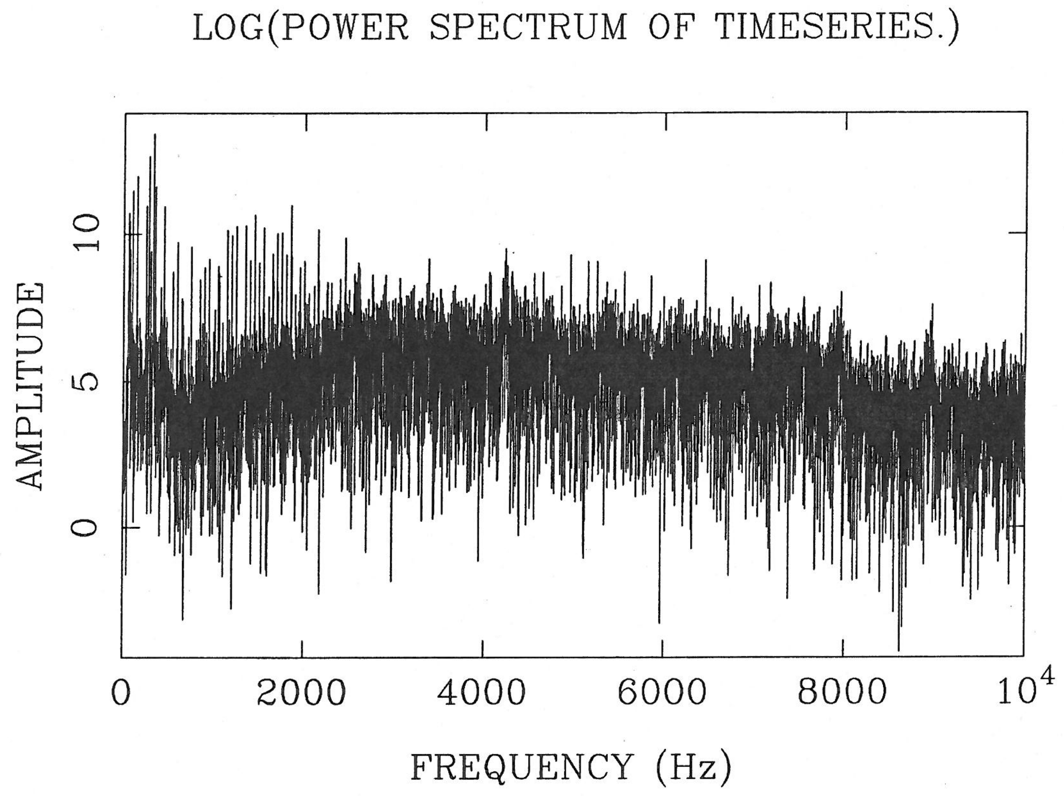
Figure 5.10 shows the log of the power spectrum of a stretch of real data before and after calibration. It should be noted that with the frequency bias removed, the second plot gives a true representation of the displacement noise in the detector.

### 5.4.4 Weighting

It is shown in Chapter 3 that the optimum filter used for the extraction of a signal by cross-correlation techniques, $\tilde{q}(\mathrm{f})$, is given by,

$$\tilde{q}(\mathrm{f}) = \frac{\tilde{h}(\mathrm{f})}{S(\mathrm{f})}.$$

$S(\mathrm{f})$ has to be calculated separately for each length of data. Rather than divide each filter in turn by $S(\mathrm{f})$ before cross-correlation, the data itself is divided through by $S(\mathrm{f})$. This optimises the data


LOG(POWER SPECTRUM OF TIMESERIES.)
AMPLITUDE
0
5
10
0
2000
4000
6000
8000
$10^4$
FREQUENCY (Hz)

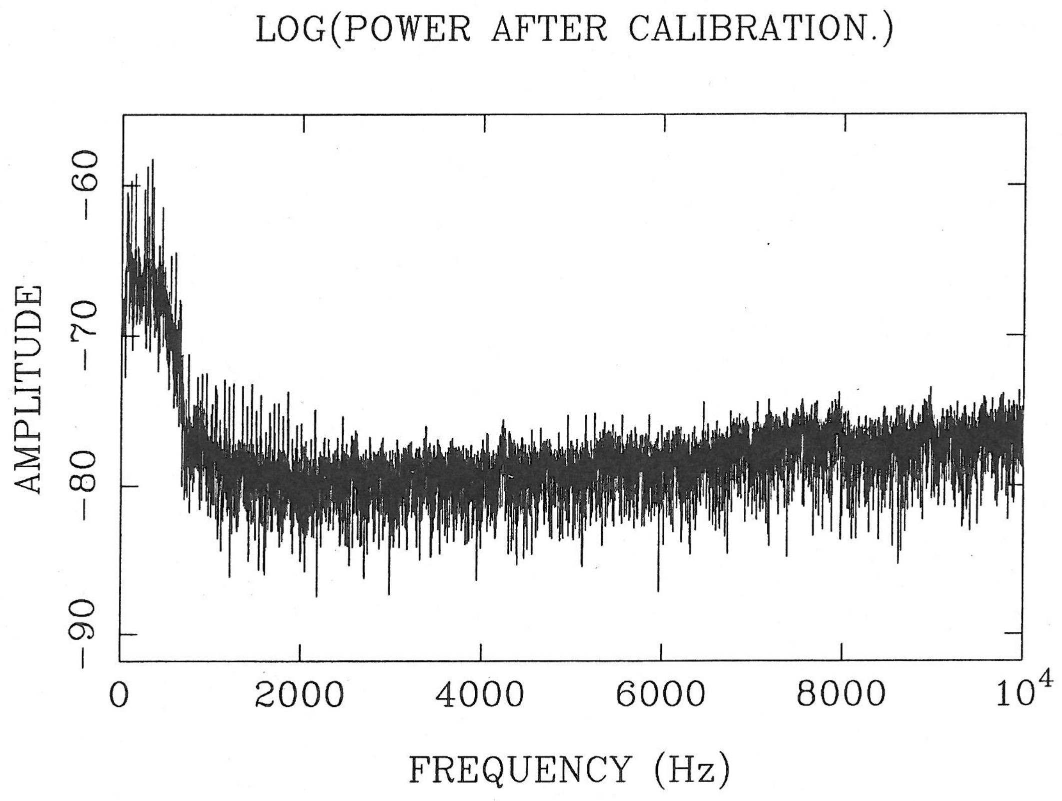

LOG(POWER AFTER CALIBRATION.)
AMPLITUDE
−90
−80
−70
−60
0
2000
4000
6000
8000
$10^4$
FREQUENCY (Hz)

FIGURE 5.10

for simple time series burst event searches as well as for Chirp searches because, just searching a data stream for points crossing some threshold is equivalent to cross-correlating the stream with a *delta* function acting as a filter. This procedure is carried out in the subroutine WEIGHT (lines 295-340).

$S$ (f) is assumed to vary both slowly and smoothly, which allows the data to be split up into regions, with one value of $S$ (f) calculated for each region. The 16384 complex data points are divided into 128 groups of 128 points. $S$ (f) is then calculated for each of these groups in turn, approximated as the variance $\sigma^2$(f), of the data in each group. For any group whose complex elements are $\mathrm{C}_{i1}$ to $\mathrm{C}_{i128}$ where $i = 1...128$ and $\bar{\mathrm{C}}$ its mean, its variance Var($i$) is given by,

$$\mathrm{Var}(i) = \frac{1}{128}\sum_{j=1}^{128}|\mathrm{C}_{ij}|^2 - \bar{\mathrm{C}}^2 \tag{5.9}$$

$$= \frac{1}{128}\sum_{j=1}^{128}\mathrm{Rl}[\mathrm{C}_{ij}]^2 + \mathrm{Im}[\mathrm{C}_{ij}]^2 - \bar{\mathrm{C}}^2. \tag{5.10}$$

These 128 values of $S$ (f) are then written to an array.

In a loop, every point is then replaced by its own value divided by the variance of the group in which it lies. Hence the value of $\mathrm{C}_{ij}$, with $i$ and $j$ defined as above, becomes,

$$\mathrm{C}_{ij} = \frac{\mathrm{C}_{ij}}{\mathrm{Var}(i)}.$$

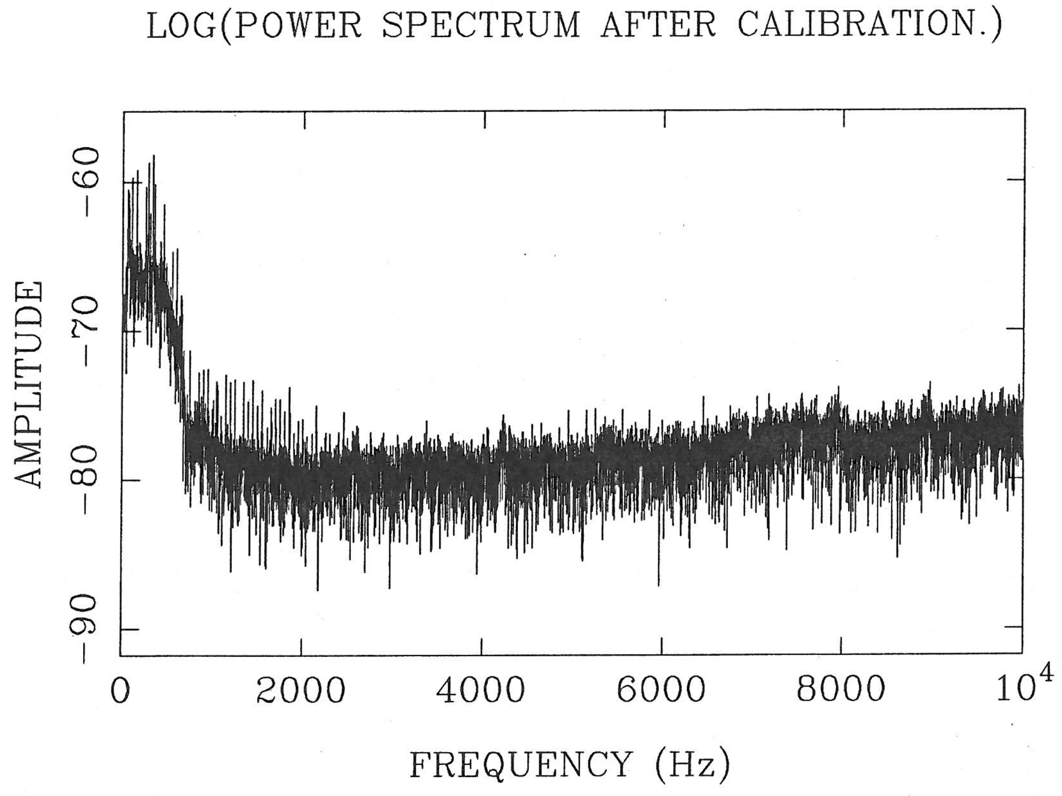
The effect of this is to optimise the filters, by reducing the influence of high noise at certain frequencies in the time series, after which the weighted data is then returned to the main routine.

The effect of this procedure can be seen in Figure 5.11 which shows calibrated data before and after this weighting procedure. Frequencies at which the noise is high (and hence undesirable) before weighting, show noise levels below the full spectrum mean afterwards. This graphically demonstrates the most serious potential drawback of this procedure, as the high noise levels tend to be at low frequencies, less than 1000Hz, which is also the region that any chirp present would occupy. Therefore in reducing the influence of the high noise, the influence of the chirp in the noise would also be diminished. Another potential problem could occur at any frequency, if some narrowband event stood out above the noise in frequency space to such an extent that it enhanced one or several values of $S$(f). The weighting procedure could then cause the strong signal to effectively cancel itself out.

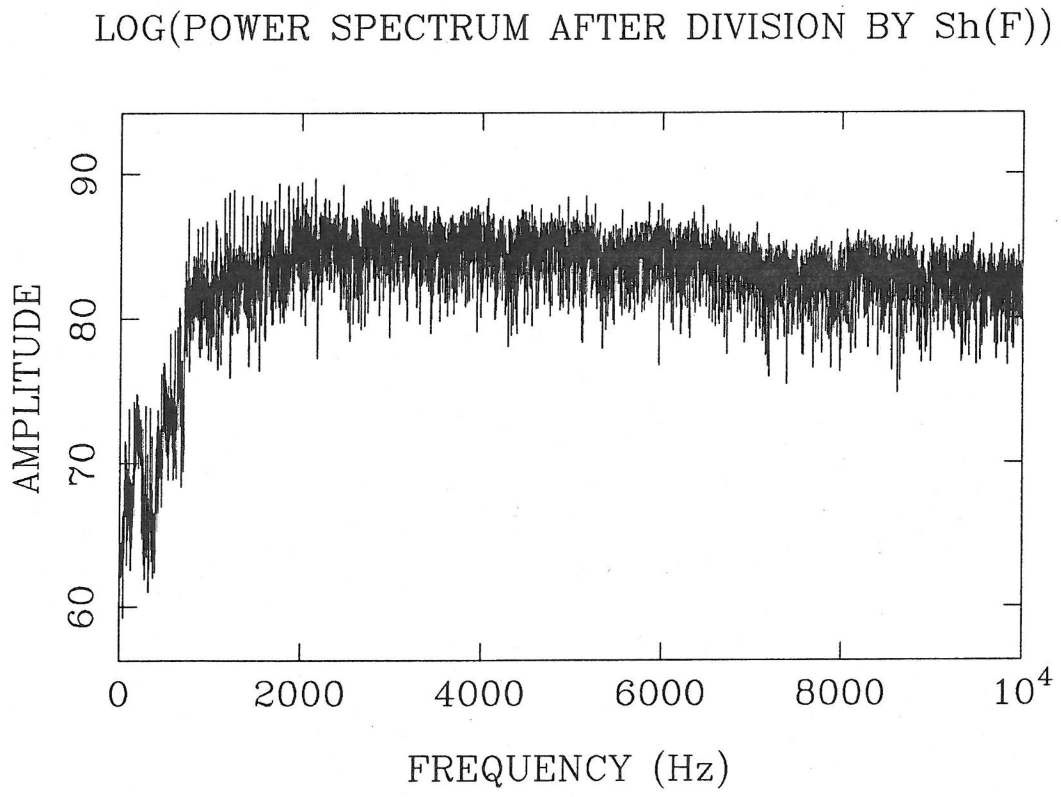
In Chapter 3 it is shown that the amplitude, $A$, of any wave that may be found in the data is given by,

$$A = \sigma(f)\left(\frac{S}{N}\right)\left(2\int_0^\infty |\tilde{s}(f)|^2\,df\right)^{-1/2}. \tag{5.11}$$

Here $\sigma$ is the standard deviation over the frequency range of the data over which the search was made. There are two frequency ranges involved, 0-10000Hz *ie* the whole stream for the



LOG(POWER SPECTRUM AFTER CALIBRATION.)
AMPLITUDE
−60
−70
−80
−90
0
2000
4000
6000
8000
$10^4$
FREQUENCY (Hz)



LOG(POWER SPECTRUM AFTER DIVISION BY Sh(F))
AMPLITUDE
90
80
70
60
0
2000
4000
6000
8000
$10^4$
FREQUENCY (Hz)

FIGURE 5.11

broad band search and also 300-1000Hz for the chirp search as these are the frequencies over which the filters are defined. These values are calculated in this routine, before the weighting, and are returned to the main program where they take up the last 2 elements (3 and 4) of the array SDT.

### 5.4.5 SECD2

#### Introduction

This routine runs in the third transputer. It receives 2 data messages from the second transputer in the analysis of a group of data. The first of these is the 32768 points of the FFT of the optimised time series and is held in SECERR. The second is the four element array SDT containing data calculated prior to the weighting operation. The 32761st point in the time series is used to pass certain instructions into the rest of the system. If point 32761 is set to -1 then the program is to be terminated and all results are returned to the root and the task halted. If point 32761 is set to 5555 then the current statistics histogram (explained below) is sent to the root.

The first 4096 points (representing 1250 Hz) are then transmitted to the next task in the next transputer along with an additional 4097 point which is normally set to 0 but when the termination or statistics return commands are set then point 4097 is set to -1 or 5555 respectively.

SECERR is then returned to its time series by finding its inverse fast Fourier transform using the same routines (REALFT and FOUR1) as used in the previous task. The data are then sent to the statistics routine explained below and from there on to the event search routine also explained below.

#### Statistics

The STATS (lines 83-127) routine is passed the group of weighted and calibrated time series data and returns its mean and standard deviation $\sigma$. It is also passed a 500 element array containing histogram data which is updated using data from the group and then returned to the main program along with a single number from which the extent to which the noise is Gaussian, can be assessed.

For each group the first 4 blocks or 26208 (= N) elements only are analysed. If an element is written as $x_i$, then the mean of the noise $\bar{x}$ is found along with the sum of the squares of each of the elements which gives the standard deviation by the following equation,

$$\sigma = \left( \frac{1}{N} \sum_{i=1}^{N} x_i^2 - \bar{x}^2 \right)^{1/2} . \tag{5.12}$$

The histogram array is a 500 element array that contains a record of the number of elements at various displacements from the mean accumulated over all the groups on that tape preceding the current group.

The distribution of data is assumed to symmetrical about its mean and all the displacements are taken as amplitude (positive) displacements starting at the mean and going up to 10 (which represents a displacement of $10\sigma$) in bins of width 1/50 ($\sigma/50$). Any points beyond 10 are counted in bin 500. The bin number $i$ to which a particular element $x_j$ should be added is given by,

$$i = \text{int}(|x_j - \bar{x}| \times 50/\sigma), \tag{5.13}$$

where int means "the integer part of". When this is established 1 is added to the number already counted in bin $i$. As a safety precaution the array is requested occasionally by the main program in the root from where it is written to the hard disk of the PC. The shape of the binned data as it should appear is shown in Figure 5.12 which is a theoretically generated histogram.

## The Gaussian Parameter

The numbers in the bins for each group separately, are used to provide a quick diagnostic test of the Gaussian extent of the noise in that group. If the 26208 points of noise considered in each group are Gaussian then the number of elements that *should* be in each of the bins can be found. The first 5 bins of displacements 0 to 1/10 (0 to $\sigma/10$) were considered. The number of points $N_{0,1/10}$ that should occupy these bins was found as follows,

$$\begin{aligned} \mathrm{N}_{0,1/10} &= A\int_0^{1/10} e^{-x^2/2}\mathrm{d}x \quad \text{where } A = 26208\sqrt{2/\pi} & (5.14)\\ &= A\int_0^{1/10}\left(1 - \frac{x^2}{2} + \frac{x^4}{8} - ....\right)\mathrm{d}x & (5.15)\\ &= A\left[x - \frac{x^3}{6} + \frac{x^5}{40} - ....\right]_0^{1/10} & (5.16)\\ &= 2088\,. & (5.17)\end{aligned}$$

By comparing this number with the number that were found to occupy the first five bins a measure of the extent to which the noise is Gaussian is possible. This can be quantified by defining the *Gaussian Parameter*, $\mathrm{N}_g$, by

$$\mathrm{N}_g = \frac{\mathrm{N}_{0,1/10}}{2088}. \tag{5.18}$$

The nearness of $\mathrm{N}_g$ to 1 can then be used as a guide to the *goodness* of the noise. It will be shown in the next Chapter where the results of the analysis are considered that this parameter is a good diagnostic. $\mathrm{N}_g$ is returned by the subroutine to the main task along with the updated histogram array as well as the mean and standard deviation of the group.

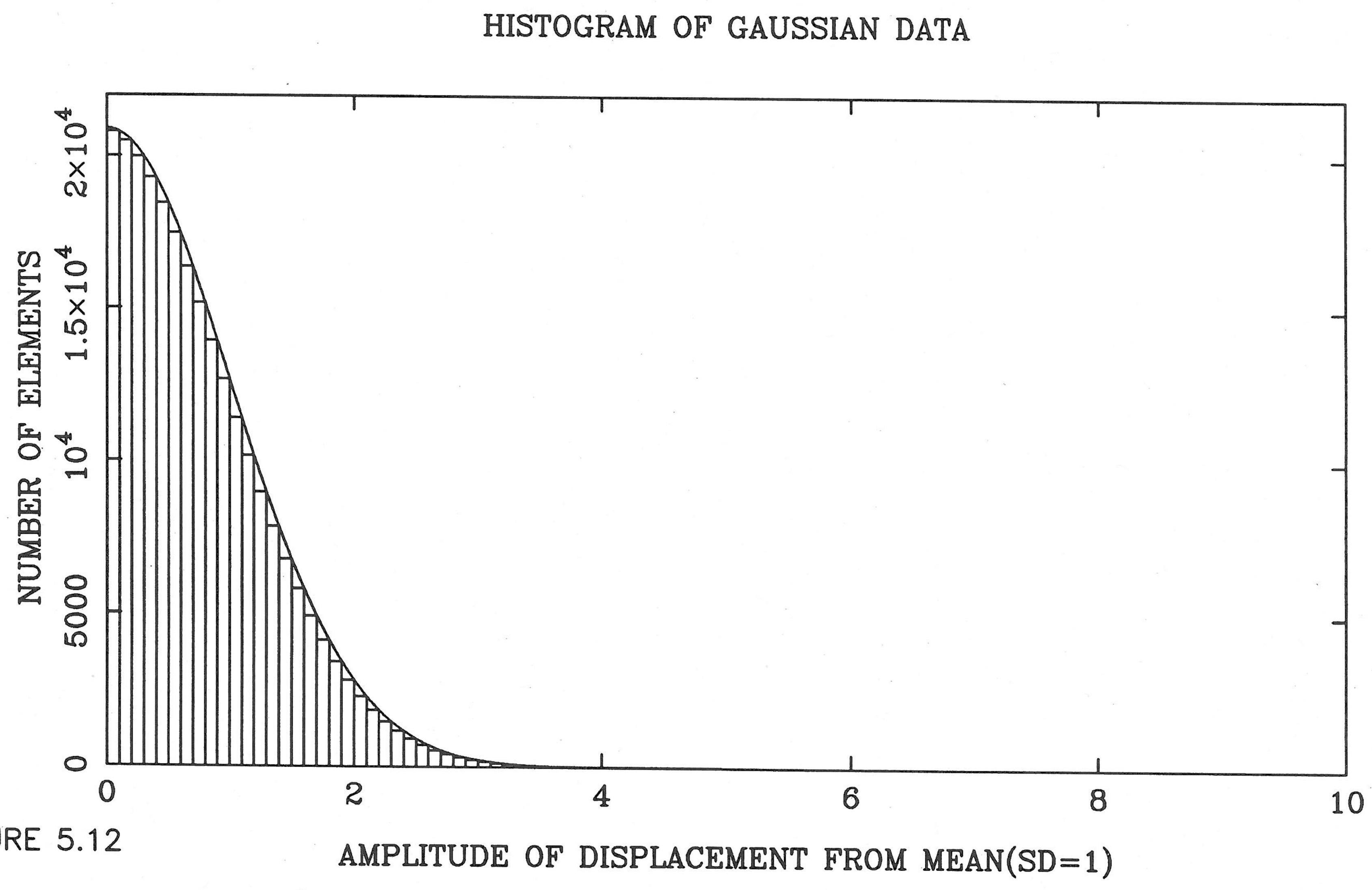

HISTOGRAM OF GAUSSIAN DATA
NUMBER OF ELEMENTS
0
5000
10^4
1.5×10^4
2×10^4
0
2
4
6
8
10
AMPLITUDE OF DISPLACEMENT FROM MEAN(SD=1)

FIGURE 5.12

### Event search

The search for threshold crossing events takes place in the subroutine called SEARCH (lines 134-214). It is given a stretch of time series data, held in the array SEGMNT along with the standard deviation and the mean of the data calculated by the previous statistics subroutine. It is also sent the four element array SDT that contains data used to attach a real value to the amplitude of an event. It returns the 200 hundred element array EVENT containing the results of the event search of the current group.

A loop is set up that goes through each of the 26208 points one by one, the first operation in each group establishing whether or not that particular point's absolute displacement from the data mean exceeds some preset multiple of the standard deviation. If it does then it is treated as an event.

The search is based around the possibility that events could be *multiple*, *ie* come in groups of two or more. No multiple events were seen in preliminary testing of the Glasgow data, however they were found to be quite common in the Garching data as well as the chirp search data from both data sets, hence it seemed reasonable to incorporate in the possibility of multiple events in the analysis. For this reason after a first event is found nothing is written to the array EVENT until the next point is also checked. If this point is also an event then its size is compared with the previous point and then again nothing is written to the results array until the next point is checked. When some point following a group of events is found not to be above threshold, then the results from the previous events are written to the results array. Associated with each set of events are 10 points in the results array. For this reason, with 200 points in total and the first 7 holding a *mini header*, data corresponding to only the first 19 events sets are written for each data group. This is reasonable on the basis of threshold chosen as it was found only non Gaussian noise actually produced this number of events. The multiple of the standard deviation that gives the event threshold was taken to be 4 as this was seen to produce sufficient events generally without exceeding the capacity of the results array. For perfectly Gaussian data one would expect only 2-3 events on the basis of this threshold for every 26208 points. But experimentation with the Glasgow data showed that on average more events were seen even in stretches of data made up from groups whose Gaussian assessment parameters were consistently very close to 1. For groups such as these, there were on average 5-7 events per group although there were also a significant number of groups that produced 15 or more events. For stretches of *bad*, non Gaussian data, 19 events were usually found long before the end of the group.

The first 7 points of EVENT are taken up with data pertaining to the whole 26208 point group. The first 3 of these points are written in this subroutine. They are, in order, the total number of events found, the groups calculated standard deviation and the groups calculated mean. The 4 other points are written when the data is returned to the main body of the task. They consist of 3 of the elements contained in the array SDT and also the Gaussian parameter $N_g$ returned by the statistics routine.

The first of the 10 element results strings starts at element 8 in EVENT. The first element of each string is its event group number, from 1 to 19, depending on the number of events previously found in the group. The next three elements give the position of the event within the group. These are numbers from 1 to 26208 and represent, in order, the position of the first threshold crossing event in the set, the position of the largest of the threshold crossing events and the position of the last of the events in the set. It is enough to use these positions only as the indicator of the position of the events in time as the time is calculated periodically throughout the analysis and the time from one of these points to an event can be calculated as the exact sampling frequency of the data is known. Should a point be an event independent of any other points then obviously all three of these position indicators will be the same.

The next 2 points written are concerned with the amplitude of the largest component of the multiple event. The first point is the ratio of its size to the standard deviation, *ie* the signal to noise, hence this is a number at least equal to four. The second point gives the calculated value for the actual amplitude of the gravitational wave that may have produced such an event. This was calculated using equation 5.11, where $\sigma(f)$ is given in SDT(4). As the filter is effectively a delta function, the integral in equation 5.11 was approximated as follows,

$$\int_0^\infty |\tilde{s}(f)|^2 \, \mathrm{d}f \approx \frac{10000}{16384} \sum_{i=1}^{16384} 1 = 10000, \tag{5.19}$$

where 10000/16384 is the frequency spacing between each sampled point.

The last four of the ten elements allocated to each event set are left blank (set to 0). These are used in the root transputer where relevant housekeeping data is added. The event array is shown diagramatically in Appendix 2.

The array EVENT is then returned to the root task along with the current histogram if requested and the task is ready to receive the next data group.

### 5.4.6 CORR1 and CORR2

#### Introduction

These routines ran simultaneously in both the fourth and fifth transputers respectively. They cross-correlated data received from task *secd2* with a suite of filters built up on the basis of the accepted mathematical model for the evolution of the gravitational radiation emitted from binary systems of varying mass parameter and phase just prior to coalescence. This characteristic shape will subsequently be referred to as a *chirp*. Unlike the 2 previous tasks these were not written as a sequence of subroutines but as continuous programs. This was because this task was not obviously divisible into separate sections that would facilitate the operation of the processes in a time efficient manner. This is essentially due to the nature of the problem. The same *extended* process involving IFFT's, statistics and event searches had to be repeated in cross-correlating the data with each of the filters in turn. If these lesser tasks were each allocated to a subroutine then the analysis of each group of data would involve many data transfers between routines which would certainly slow the program up.

Basically, each group was cross-correlated with each pair of filters (both based on the same mass parameter) in turn. The results from all of these correlations were statistically analysed. This was followed by an event search, the results from which were added to an overall 200 element results array associated with that group. This array contained the results from all the cross-correlations with the filters in that transputer. These arrays from both transputers were then returned to the Root task. The various procedures employed, in the correct order, are explained below.

Commands to terminate the tasks or return histogram statistics are sent in point 4097 of the data. This point plays no part in the correlation. The array SDT was also received from *secd2*.

### Filters

The suite of filters was built up at the start of each analysis run in both tasks. This takes place only while the Exabytes are undergoing their initiation procedure, Hence the program is not delayed by the filter formation. All the filters on each of the tasks are built into part of the same array called FILTERS. The filters are initially built up in the time domain as the chirp signals for particular mass parameters and phases. Each filter is 32768 points long with a relatively small calculated non zero part at the start and all other points equal to zero. The FFT of the filter is then found, of which the first 4096 points are written to the array FILTERS in the appropriate place. This represents the first 1250 Hz which includes all the significant power in the chirp transform. Ten filters are used in total in each task which means that FILTERS is 10 $\times$ 4096 = 40960 elements long.

The filters are built using equations given in function statements defined at the start of the program (lines 13-16). These statements are based on equation 2.7 to equation 2.9 given in Chapter 2. The central statement, defining the chirp is

```
CHIRP(TIM,PAR,PHASE)=A(TIM,PAR)*COS(B(TIM,PAR)+PHASE*1.570796)
```

where TIM is the time in seconds starting at zero when the frequency of the chirp reached 100 Hz.

PHASE is the phase of the wave at 100 Hz and PAR the mass parameter defined by,

$$\texttt{PAR} = (\text{reduced mass})(\text{total mass})^{2/3} = \frac{M_1 M_2}{(M_1 + M_2)^{1/3}}, \tag{5.20}$$

where $M_1$ and $M_2$ are the masses of the components of the binary system in units of solar mass. A(TIM,PAR) is defined by the statement,

```
A(TIM,PAR)=(1-0.34*TIM*PAR)**(-.25)
```

and B(TIM, PAR) by,

```
B(TIM,PAR)=(1005.31*(1-A(TIM,PAR)**(-5/2))/(0.34*PAR))
```

Thus for different masses and phases the chirp signal at different times after 100 Hz can be calculated. The frequency range considered here is from 300 Hz to 1000 Hz. It was hence necessary to know the time $t_{300}$ for each mass parameter at which the signal reached 300 Hz and also the time taken for the frequency of the wave to increase to 1000 Hz. From equation 2.8, the time $t_f$ taken for the chirp to reach a frequency f with $f \geq 100$Hz is given by,

$$t_f = \frac{[1 - (f/100)^{-8/3}]}{0.34 \times \texttt{PAR}}. \tag{5.21}$$

Hence the time taken to reach 300 Hz is $t_{300}$ given by,

$$t_{300} = \frac{[1 - 3^{-8/3}]}{0.34 \times \texttt{PAR}} \tag{5.22}$$

and the time taken, TD(PAR) for the frequency to go from 300 Hz to 1000 Hz is the fourth function statement and is given from equation 2.9,

```
TD(PAR)=(3.**(-8/3)-10.**(-8/3))/(.34*PAR)
```

The sampling rate of the data cross-correlated with the filters was 20 kHz hence this must also be the sampling rate at which each of the filters is formed. So the minimum number of points, NPTS, that must be calculated in a loop increasing with time by 0.0005 sec each loop and starting at $t_{300}$ is given by,

```
NPTS=INT(TD(PAR)*20000.)+1
```

The 10 filters in each task, have 5 different mass parameters and 2 phases, with the mass parameters representing the chirp obtained from coalescing binaries with equal components. Sathyaprakash, et al (1991) show that ideally mass parameters should increase by about 2% for each filter pair, starting at about 1.39 going up to about 120. With so few filters, covering this whole range of potential parameters is impossible. It was hence decided to cluster the parameters in one task around the value expected for most neutron stars, *ie* about 1.4 increasing by about 5% each step, and in the other to cover a wider range from 2 to 6 with variable percentage increases. The component masses ranged from 1.395 $\mathrm{M}_\odot$ to 1.412 $\mathrm{M}_\odot$ with a step size of 0.0583 in task *corr2* and 1.74 $\mathrm{M}_\odot$ to 3.017 $\mathrm{M}_\odot$ with a step size of 1.1487 in task *corr1*. The 2 phases used just needed to differ by $\pi/2$ hence 0 and $\pi/2$ were chosen. The filters are built up in two loops over mass parameter and phase shown in lines 43-90.

As in the previous task, to calculate the true amplitude of any event, the integral in equation 5.11 has to be calculated. For a filter, $\tilde{g}(f)$, sampled at points $\tilde{g}_i$, the integral can be approximated as follows,

$$\int_0^\infty |\tilde{g}(f)|^2 \, \mathrm{d}f \approx \frac{10000}{16384} \sum_{i=492}^{1639} \tilde{g}_i^2, \tag{5.23}$$

where 10000/16384 is the frequency gap between each sample. This was calculated for each filter in turn, at the start of the tasks, with the results stored in the array FILNORM.

## Correlation

When the 4096 points of the FFT of the data group (FDATA) were received by *corr1* it was immediately sent on to *corr2*. Simultaneously then, in both these tasks the data is cross-correlated with all the filters stored in the template array FILTERS. The data was correlated with each 4096 point filter within a DO loop that moves through FILTERS by the mass parameters one by one calculating in each case the cross-correlation for both phases at that parameter. To move through both filters of any one mass parameter, the position indicator within FILTERS needs to be incremented by 8192 in each loop.

The cross-correlations were calculated by application of Equation 5.24 originally derived in Chapter 3.

$$\text{cross} - \text{correlation} = \text{IFFT}\left[\texttt{FDATA} \times \texttt{FILTER}^*\right], \tag{5.24}$$

where IFFT is the inverse FFT, FILTER is one of the 4096 element segments of the template and $*$ implies the complex conjugate. Both FDATA and FILTER are complex and hence when multiplied, produce a complex result. So if element $(\mathrm{a} + i\mathrm{b})$ from FDATA is multiplied by element $(\mathrm{c} + i\mathrm{d})^*$ from FILTER the result is $((\mathrm{ac} + \mathrm{bd}) + i(\mathrm{bc} - \mathrm{ad})$. The IFFT of this stream produced the 4096 point real data stream, CORR, which was the time series cross-correlation representing the same time interval (1.6384 sec) as the original 32768 point group, with 1/8 of the sampling frequency.

In Chapter 3 the wraparound phenomenon that effects discrete cross-correlation is described. In this analysis its effect was mitigated by limiting the number of non zero points at the start of any of the filters, but still if a filter had NPTS non zero elements, then the last NPTS of CORR were meaningless and hence were ignored. The smallest mass used was 1.395 $M_\odot$ which provided the largest non zero part of any of the filters with a non zero length of 2162 points. This is less than the length of one data block and is the reason why the event search covers only the first 4 blocks in any group. This is also why, in order not to lose any data, the original version of the fifth block in any group becomes the first block in the next group.

### Statistics and event search

As explained above only the equivalent of the first four blocks or 26208 points were actually analysed in any group, hence as 4096 points of CORR represent the whole group, only the first 3276 (= 4096 × 26208/32768) points were actually studied in the event search. The event search itself was carried out in a similar way to the search in *secd2* producing a 200 element results array EVENT for each group in both of the tasks. Whereas however, in *secd2* there are 10 elements associated with each event, here there are 12. This means a maximum of only 16 events for each group from all the filters was recorded. Again this is not overly worrying as it was found that even in badly behaved data the number of events rarely exceeded 6 or 7.

Firstly the means and standard deviations, $\sigma_0$ and $\sigma_{\pi/2}$, of the two cross-correlation streams $\mathtt{CORR}_0$ and $\mathtt{CORR}_{\pi/2}$ associated with any one mass parameter are found. Histogram statistics are again compiled over the correlation data up to $10\sigma$ with the data being written to 100 bins. These are again returned, when requested, to the main task where they are written to the hard disc of the PC.

Any incoming wave would not have been likely to have had the same phase as either of the filters. Hence the true cross-correlation, CORR, would have to have been a phase dependant, linear combination of the correlations with both filters. Hence, for correlation element $i$, $\mathtt{CORR}(i)$ is given by,

$$\mathtt{CORR}(i) \;=\; \cos(\theta)\mathtt{CORR}_0(i) \,+\, \sin(\theta)\mathtt{CORR}_{\pi/2}(i) \tag{5.25}$$

where both $\mathtt{CORR}_0(i)$ and $\mathtt{CORR}_{\pi/2}(i)$ here represent the deviations of these points from their respective means. This is a function of $\theta$ and is maximised when $\theta$ is the actual phase of the correlation. Hence by differentiating Equation 5.25 and setting the resulting expression to 0, $\theta$ can be found as follows,

$$\frac{\mathrm{d}\mathtt{CORR}(i)}{\mathrm{d}\theta} \;=\; -\sin(\theta)\mathtt{CORR}_0(i) \,+\, \cos(\theta)\mathtt{CORR}_{\pi/2}(i) \;=\; 0\,. \tag{5.26}$$

This leads to the following expression for phase,

$$\theta \;=\; \arctan\left(\frac{\mathtt{CORR}_{\pi/2}(i)}{\mathtt{CORR}_0(i)}\right). \tag{5.27}$$

$\pi$ or $2\pi$ are added to the value of $\theta$ where appropriate to make the angle positive and less than $2\pi$. This is decided on the basis of the signs of $\mathtt{CORR}_0(i)$ and $\mathtt{CORR}_{\pi/2}(i)$ that establish the quadrant in which $\theta$ lies.

If $\theta$ is the true phase then $\mathtt{CORR}_0(i) = \cos(\theta)\mathtt{CORR}(i)$ and $\mathtt{CORR}_{\pi/2}(i) = \sin(\theta)\mathtt{CORR}(i)$. Hence it was possible to find a phase independent value for the amplitude of the true cross-correlation given by,

$$|\mathtt{CORR}(i)| = ((\mathtt{CORR}_0(i))^2 + (\mathtt{CORR}_{\pi/2}(i))^2)^{1/2}. \tag{5.28}$$

It is this amplitude stream over which the event search was carried out.

Any point of CORR whose amplitude deviation exceeds the allocated threshold THR is considered to be an event. The deviation from the mean is already incorporated within CORR. The standard deviations $\sigma_0$ and $\sigma_{\pi/2}$ are used to establish the standard deviation $\sigma$ of the stream CORR where $\sigma$ is given by,

$$\sigma = (\sigma_0^2 + \sigma_{\pi/2}^2)^{1/2}. \tag{5.29}$$

THR is then, just a fixed multiple of $\sigma$. The multiple was chosen to be 3.5 by essentially the same criteria as the threshold in *secd2*, *ie* to ensure a reasonable number of events for *good* data.

Tests carried out showed that this software was able to correctly locate various artificial chirps buried in Gaussian noise giving an accurate estimate of the mass parameter but only accurately finding the phase when it was close to some integer multiple of $\pi/2$. This is in accordance with work done by Dhurandar and others (private communication). Details of the tests are given in Appendix 1.

Again any event found was treated as though it was the first of an unbroken set of events. The event in the set with the highest amplitude was taken to be the main event whose amplitude, mass parameter, and phase were written to EVENT. Also calculated was the true amplitude of each event, using equation 5.11. The standard deviation of the noise over the frequency range was taken from the array element SDT(3) and the value of the integral for that filter, from FILNORM. This too is written to EVENT.

As there are only an 8th as many points to search for events here as in *secd2* the accuracy to which events can be located is also only an 8th as great. To improve the accuracy the values of the correlations either side of the main event were used in conjunction with the main event to predict the position of the peak correlation for that filter.

Two arrays were set up, one Y, containing the amplitudes of the 3 points and the other, X, the three values -1,0 and +1, representing the positions of the points. These arrays were sent to the routine POLCOE taken from Numerical Recipes by Press *et al* (1986) which returns the coefficients of a quadratic polynomial that gives the equation of the best curve through the points, plotting the correlations against the positions. The true positions of the points were not used as they could be extremely large which could lead to coefficients that exceed the transputer capacity.

If the coefficients returned are $A, B$ and $C$ then the curve can be written in terms of $x$ and $y$ by,

$$y = Ax^2 + Bx + C. \tag{5.30}$$

The correlation is maximised at the point where the above Equation is maximised which can be found differentiating and setting the result to 0. This point is given by,

$$\frac{dy}{dx} = 2Ax + B = 0 . \tag{5.31}$$

This gives the position of peak correlation, $x$ given by,

$$x = \frac{-B}{2A}$$

where $x$ is guaranteed to be within the range (-1,+1) as `Y(2)` was the maximum correlation found and as such, at least as large as both `Y(1)` and `Y(3)`.

The actual position, $x_{max}$, of the peak correlation in the data, is given by,

$$x_{max} = x + x_{0max},$$

where $x_{0max}$ is the actual position of `Y(2)` in the data.

This number $x_{max}$ produced was then made equivalent to the positions given by the event search in the time series data by the following equation,

$$P_{max} = \text{int}(8 \times x_{max} + 0.49), \tag{5.32}$$

where $P_{max}$ is the event's best position in a stream 26206 elements long. So if, for example, $x_{0max} =$ 2013 and $x_{max} = 2012.84$ then $P_{max} = 16103$. By experimentation this method was, in general, found to improve the accuracy of the event's positions. This is described further in Appendix 1.

$P_{max}$ is also written to the results array along with the positions of the first and last events in the set of events that surround the peak event. The results arrays from both tasks are then returned to the root where the relevant housekeeping data is added to the 4 remaining unfilled elements associated with each event. All the results array for that group are then combined into a larger array which is at the appropriate time written to tape. The structure of this array is explained in greater detail in Appendix 2.

# Chapter 6

# RESULTS.

## 6.1 Introduction

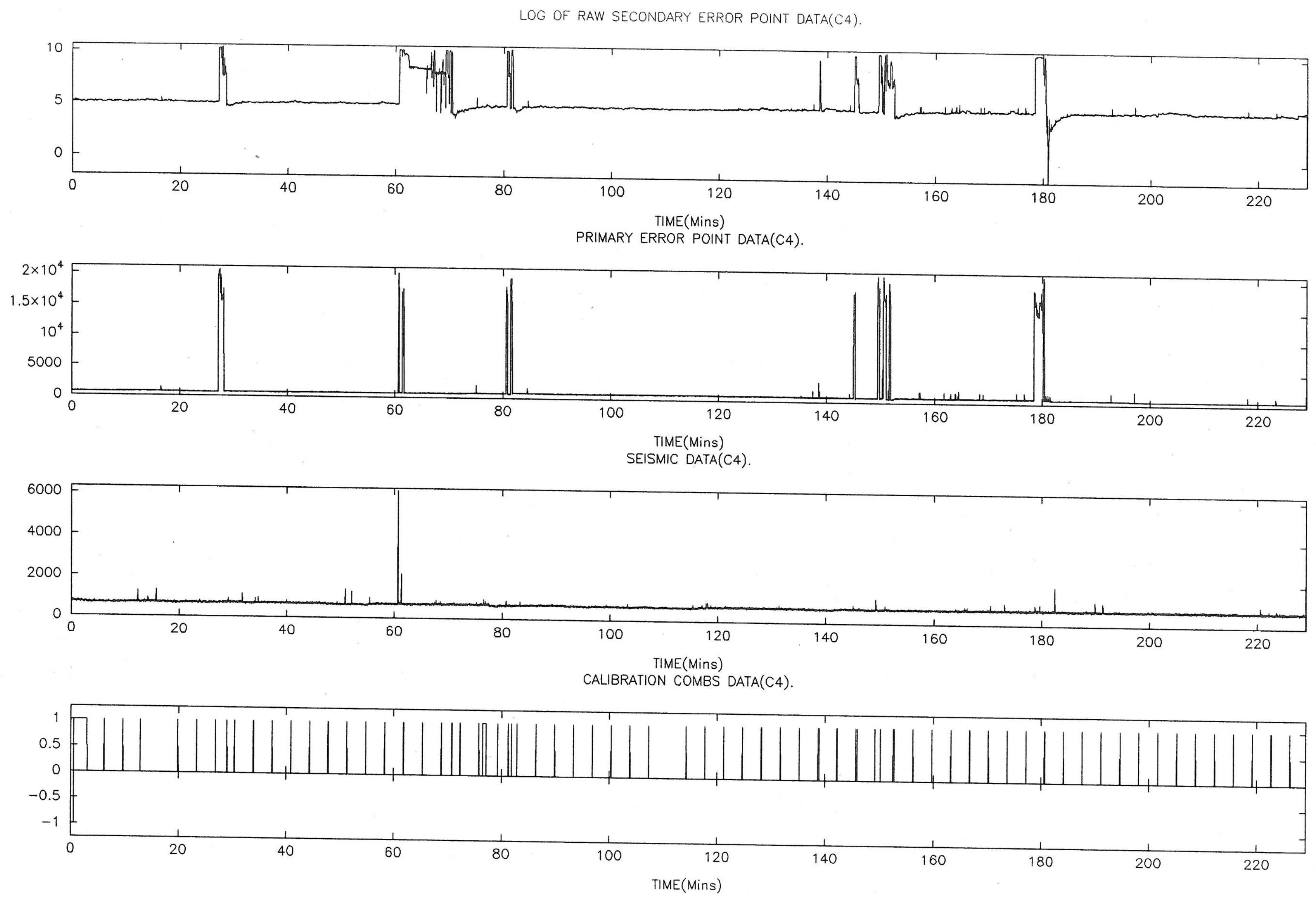
The analysis run carried out using the software developed in the last Chapter began on the 19th November 1990. Initially 6 weeks seemed a reasonable estimate for its duration based on the knowledge that the analysis ran at roughly 8 times the data taking rate of the detector over its 100 hour run. This estimate turned out to be rather optimistic as the exabyte machines seemed to be incapable of maintaining sustained usage and continually jammed or crashed or whatever exabytes do when they fail to work. The analysis was finally completed on the 11th February 1991 after 13 weeks. This problem was then further compounded by the exabyte's inability to write to certain points of some of the tapes, and then to read data just written which led to the results of the analysis initially being spread over smaller parts of a number of different tapes.

In this Chapter I discuss the results of the analysis, describing the noise statistics of both the gravitational wave (secondary error point) data, and the various streams of housekeeping data, as well as any common features between the 2. I also discuss the threshold crossing events found from the analysis.

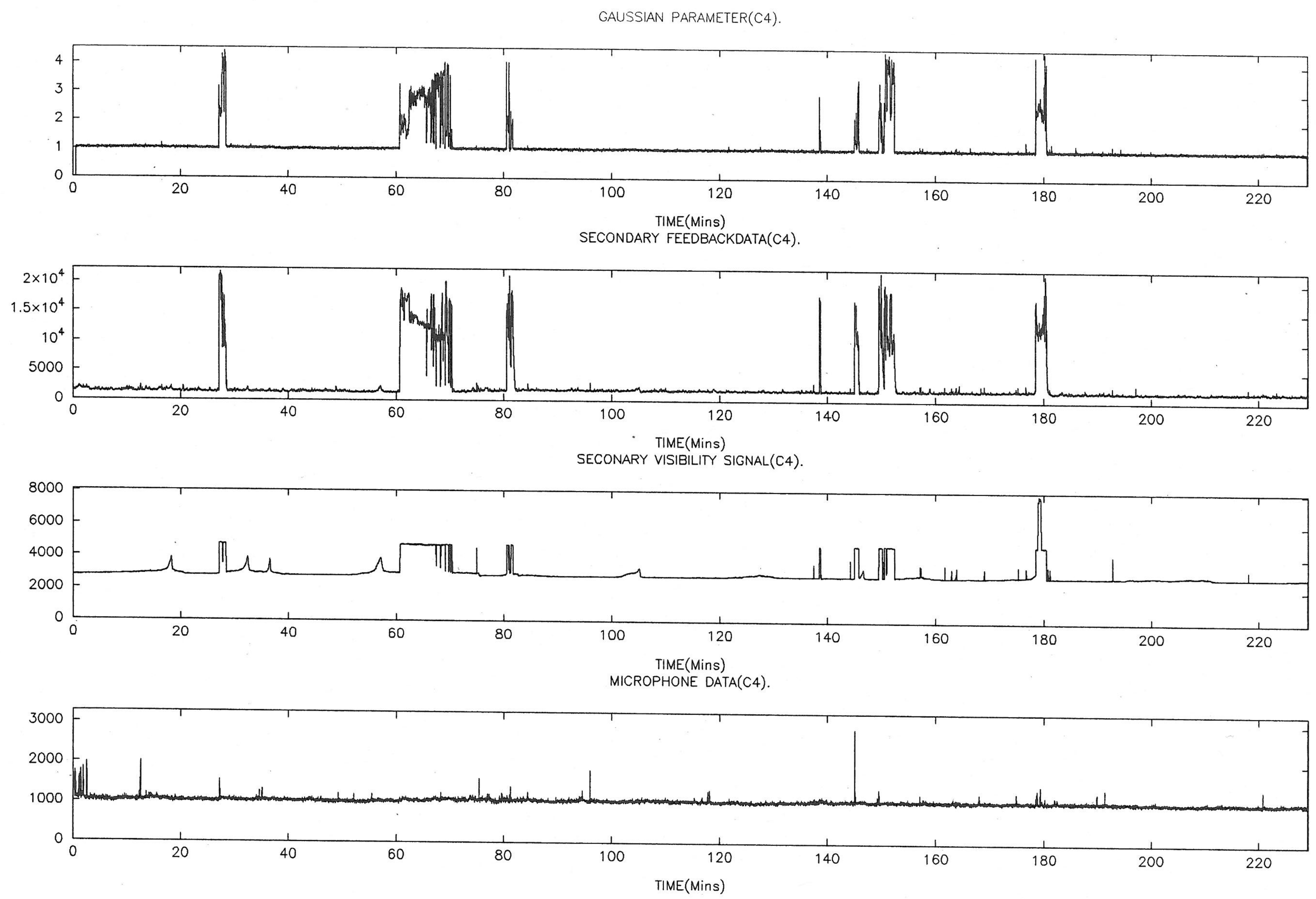
## 6.2 Details of results obtained from tapes B6 and C4

### 6.2.1 Introduction

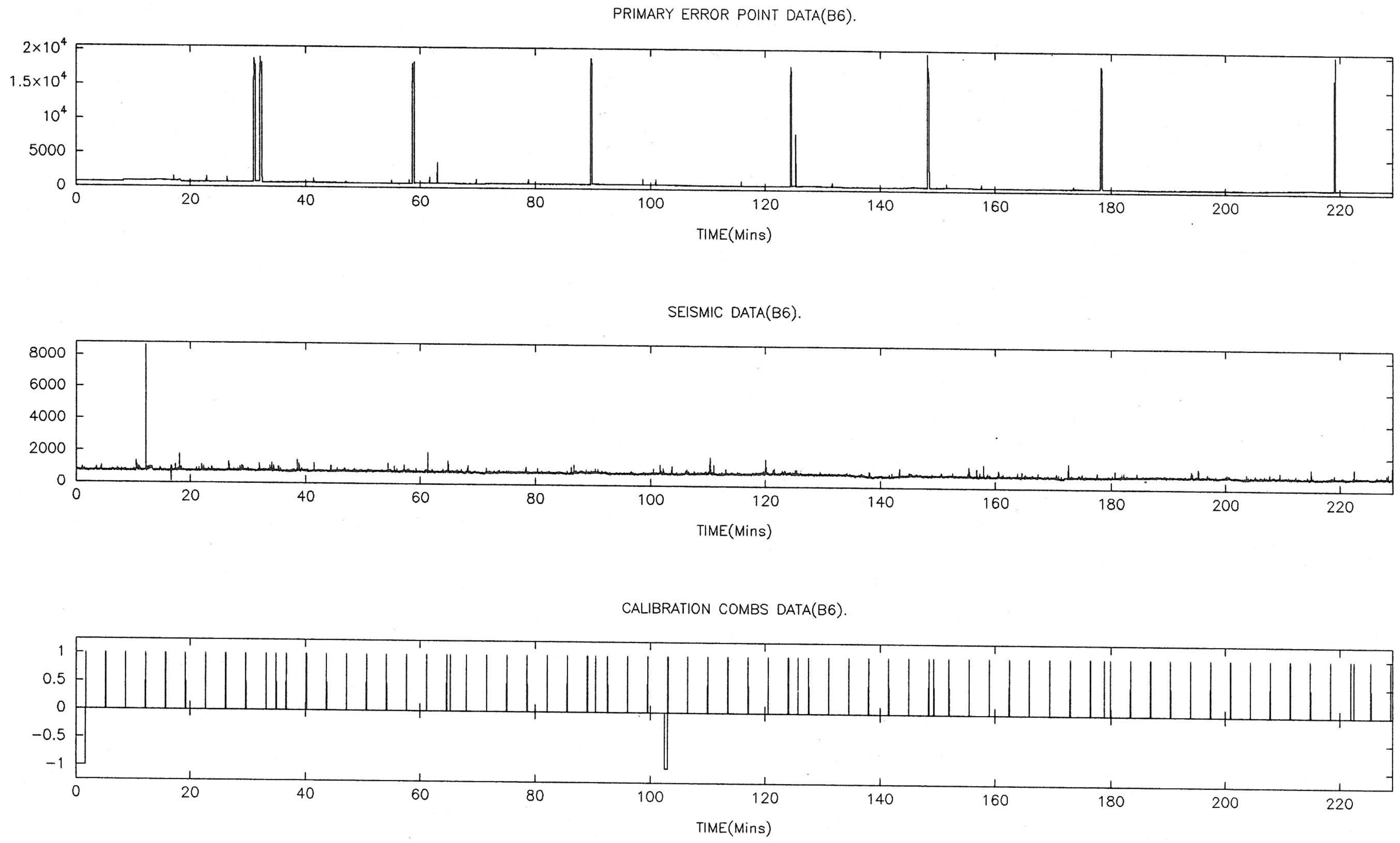
Tapes B6 and C4 began recording data on Friday, 3rd March 1989 at 10:29 and Saturday, 5th March at 22:42 respectively. They both ended approximately 229 minutes later at 14:18 and 2:31. Upon investigation the results obtained from these tapes were found to be typical of the overall results obtained from the whole run. Hence here I use the results obtained from these tape to demonstrate the relationships



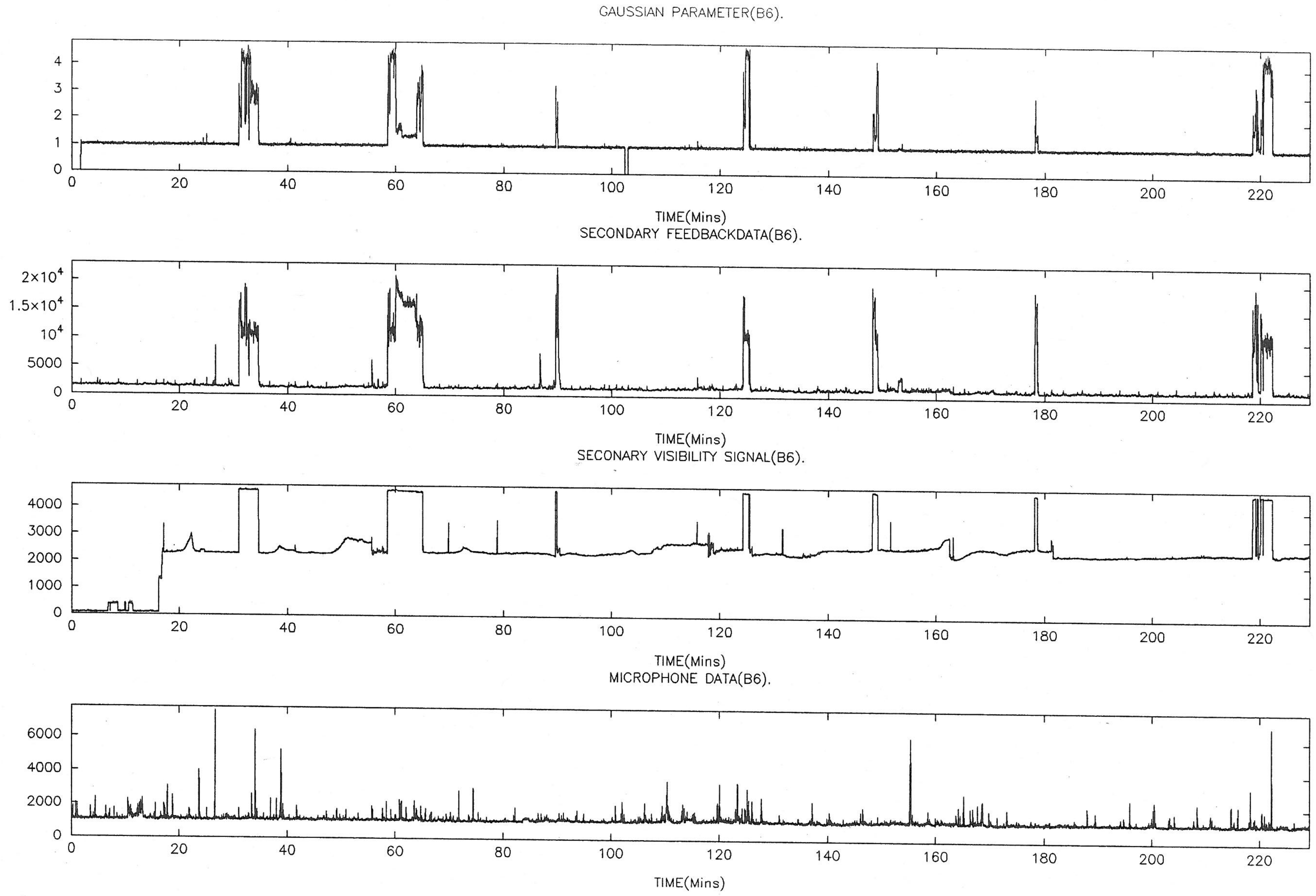
**EVOLUTION OF STREAMS WRITTEN TO TAPE C4:– (FIGURE6.1)**



EVOLUTION OF STREAMS WRITTEN TO TAPE C4:– (FIGURE6.2)



EVOLUTION OF STREAMS WRITTEN TO TAPE B6:- (FIGURE6.3)



EVOLUTION OF STREAMS WRITTEN TO TAPE B6:- (FIGURE6.4) between the housekeeping streams, the secondary error point data and also the Gaussian parameter which was described in the previous Chapter and which here I show to be a good diagnostic. For each group, the means of each of the housekeeping streams were found and written to the results tape. The group means of all these streams, for tapes B6 and C4, are shown in Figures 6.1-6.4.

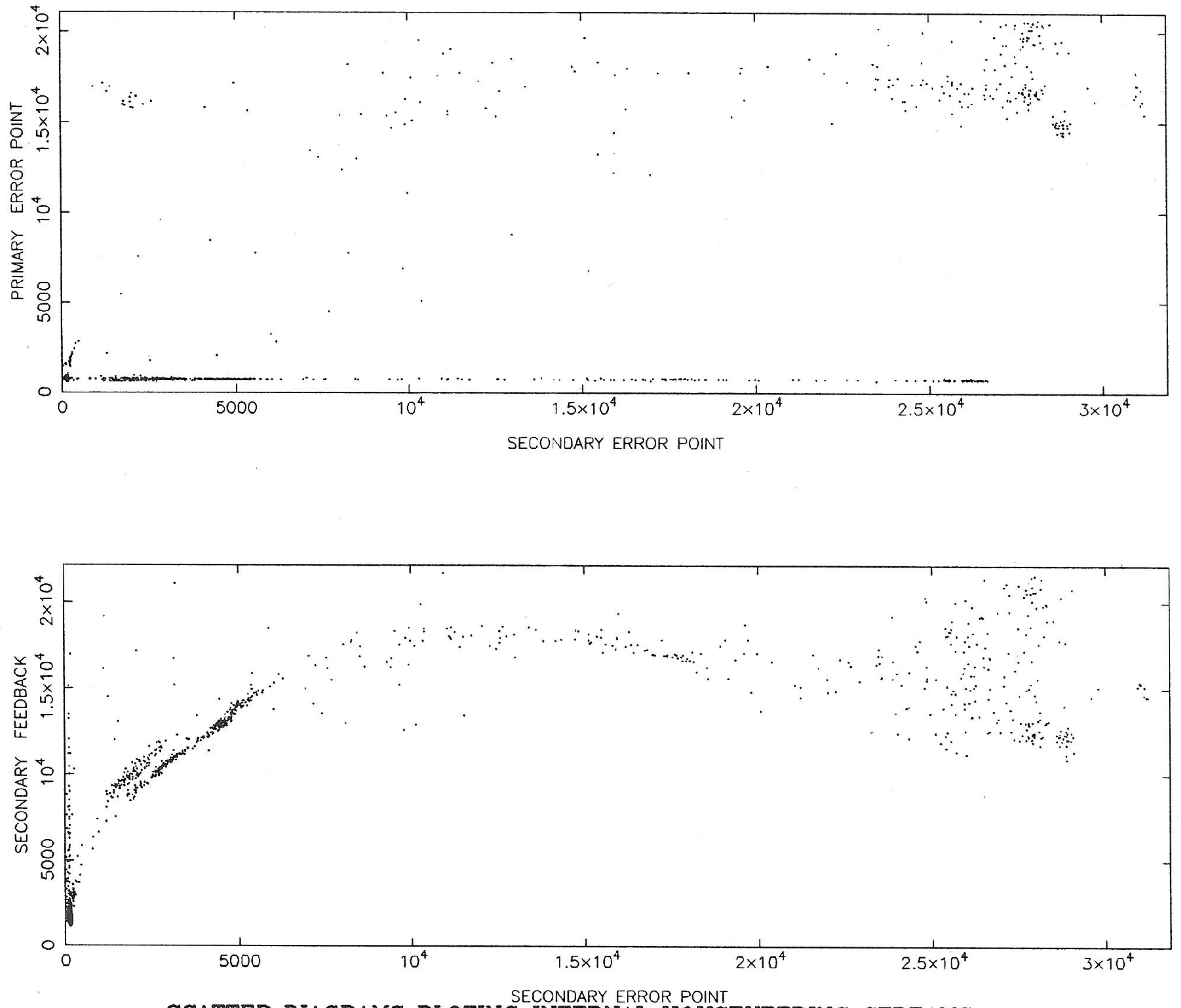
### 6.2.2 Internal data

The internal data streams are those obtained from sensors reading directly from the detector. These are the primary error point signal, read from the primary cavity and both the secondary feedback and the secondary visibility signals read from the secondary cavity as well as the secondary error point signal upon which the event search is carried out. Figures 6.1-6.4 show that these streams maintain a relatively fixed value most of the time, with any significant change in one appearing in the others as well. By consultation with the diary kept by the Glasgow group during the 100 hour run, it is clear that by far the most significant of these changes occurred when the one or other, of the cavities went out of lock. When this occurred the noise levels in each of these streams jumped up to several times its *in lock* level.

When the noise was seen to go bad it would do so abruptly with the means of both the secondary error point and the secondary feedback data suddenly jumping to a far higher level. The primary data also shows this but frequently a second or so after it is seen in the secondary signals which *is* surprising as often the source of the bad noise is the primary cavity when it goes out of lock. In analysis terms however the visibility signals may turn out to be more significant as the value of the secondary visibility signal sometimes jumps in the analysis group, (of four blocks), before the group that shows the bad noise in the other secondary streams. This could represent recognition of as much as an additional 1.3 seconds of bad noise. In determining the end of the bad data stretch the visibility data is not so helpful as it returns to its *in lock* state several seconds before the end of the bad data according to the other secondary streams. The primary data shows correlation only at the start of the bad stretches returning to its good data mean long before the secondary data.

Scatter diagrams showing the extent of the correlations between high noise in both the secondary feedback and primary error point signals and the secondary error point signal is shown in Figure 6.5A and Figure 6.5B. This data represents the whole of tape C4.

Figure 6.5A shows the primary error point data plotted against the secondary error point data. The abrupt distinction between low and high noise in both streams is noticeable in that the middle region of the Figure contains very few points,with concentrations in the bottom left corner (low noise in both) and also in the top right which is high noise in both. The line of points along the bottom of



SCATTER DIAGRAMS PLOTING INTERNAL HOUSEKEEPING STREAMS

AGAINST RAW SECONDARY ERRORPOINT DATA: FIGURE6.5

the Figure results from the tendency (mentioned above) for the primary cavity to regain lock before the secondary during a stretch of high noise.

Essentially the same correlation is seen with the secondary feedback signal, which is not surprising as the two signals are basically different frequency range samples of the same signal, with low and high noise regions common to both.

### 6.2.3 Gaussian considerations

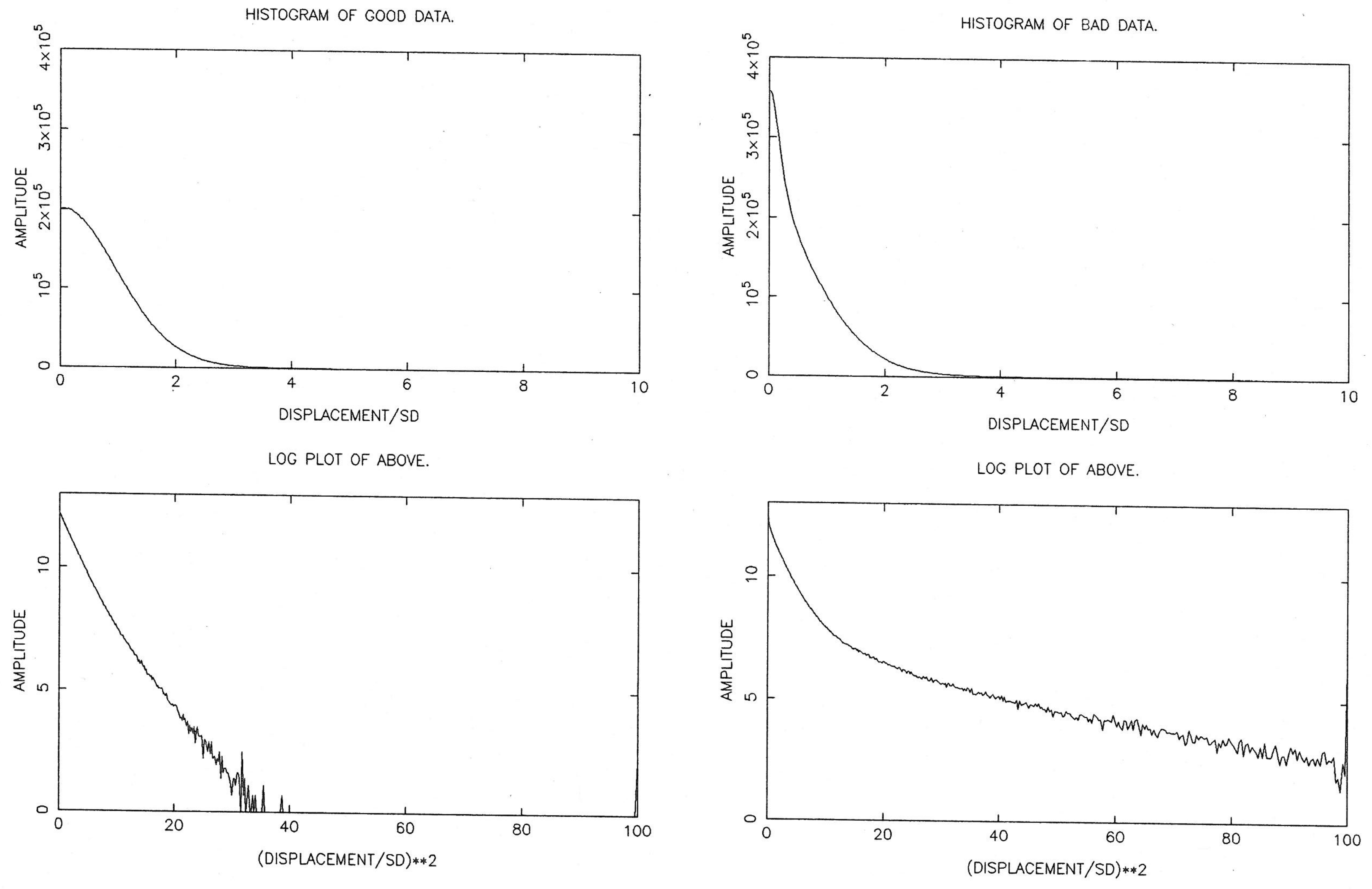
The major consequence deducible from the above section is that it is necessary only to look at the secondary error point noise to determine when the detector is operating properly, with *well behaved*, low level noise produced in both cavities. When the detector is well behaved, the noise produced is declared to be *good* and when the detector is not well behaved, with the cavities out of lock, the noise is declared to be *bad*. Then when analysing these regions of both good and bad noise, it was found that good noise is essentially Gaussian and the bad noise not. Figure 6.6 shows graphically the distinctions between good and bad noise. The diagrams are built up from part of the histogram data taken during the run. The left hand plots are formed from a histogram built up over 500 groups where the noise was known to be good and similarly the right hand side where the noise is bad. The top plots show the histogram data plotted against displacement (which is the actual displacement divided by the standard deviation) from the mean.

The top left hand plot associated with the good data clearly shows the traditional bell like Gaussian shape. The traditional Gaussian shape is given by the equation $Ae^{-x^2/2}$ where $A$ is a constant and $x$ is the displacement in terms of $\sigma$. Obviously then if the logarithm of a Gaussian is plotted against $x^2$ then the resulting curve would be a straight line of slope $-\frac{1}{2}$. This is seen to be roughly the case with the good data as shown in the bottom left hand plot but not with the bad data where the slope of the curve changes sharply at roughly $3\sigma$ splitting the diagram up into 2 regions both containing extended straight line portions. This lead to the idea that in bad noise two separate Gaussian statistics were present, one dominant in the region before $3\sigma$ and the other in the region after $3\sigma$.

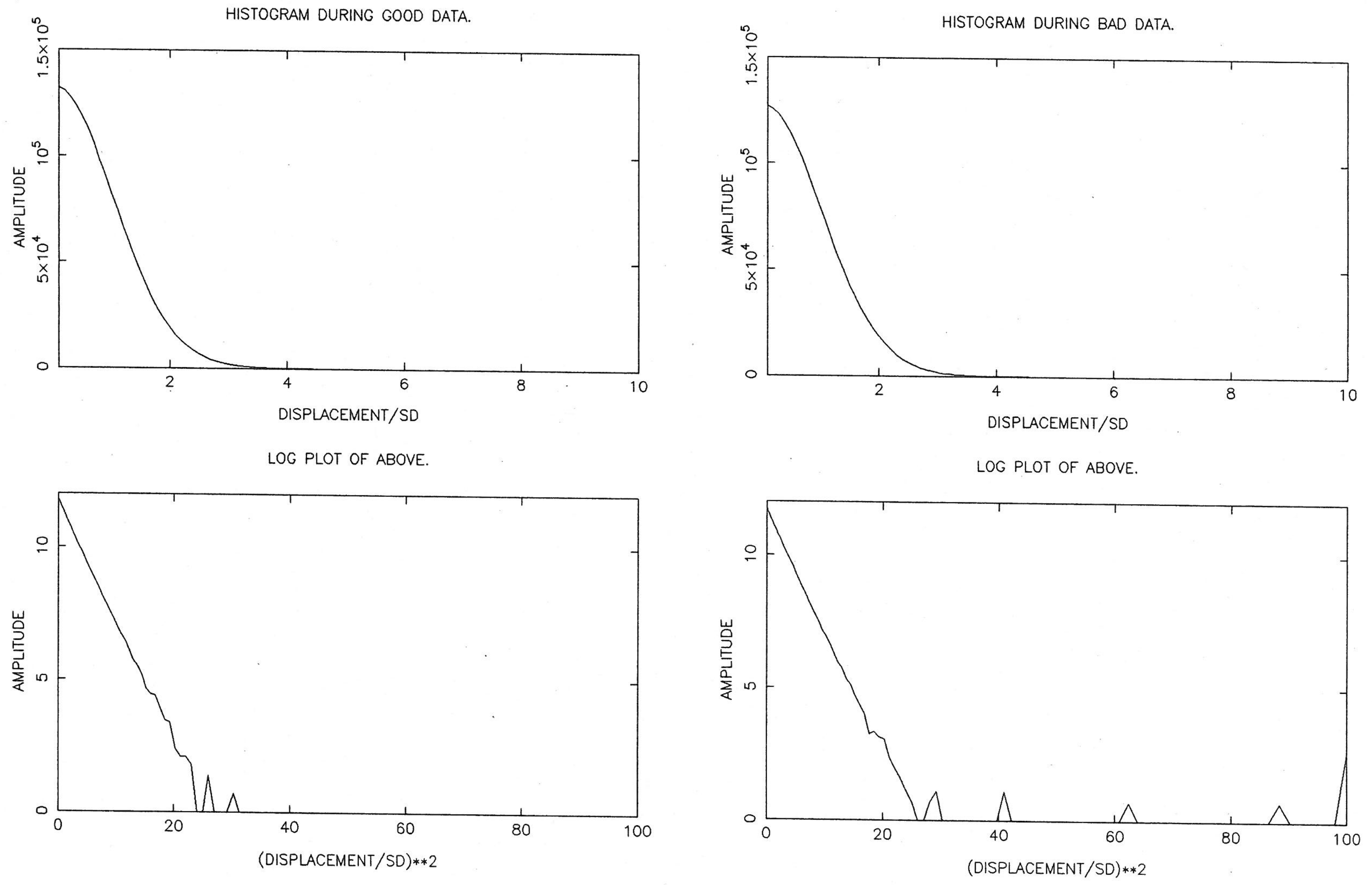
Obviously, as the diagram is a logplot, there are far fewer elements concerned with the region after $3\sigma$ than with the region before. If the two statistics are defined on the basis of two standard deviations $\sigma_1$ and $\sigma_2$ and if when combined, the number of elements controlled by statistic 1 are given by $n$ and the total number of elements is $N$ then the standard deviation, $\sigma_t$, of the whole is given by,

$$\sigma_t = \sqrt{\frac{n}{N}\sigma_1^2 + \left(1 - \frac{n}{N}\right)\sigma_2^2}. \tag{6.1}$$

TYPICAL AMPLITUDE DISTRIBUTIONS:- Glasgow 10m prototype. (FIGURE6.6)



TYPICAL AMPLITUDE DISTRIBUTIONS OF FILTERED TIMESERIES:- Glasgow 10m prototype. (FIGURE6.7)



From this it can be seen, not surprisingly, that if the elements obeying the statistics with the lower $\sigma_1$ far out number the others then $\sigma_t$ is roughly $\sigma_1$, provided $\sigma_2$ is not very much greater (*ie* an order of magnitude or so) than $\sigma_1$. For example if $\sigma_1$ is half $\sigma_2$ and if $\sigma_1$ comprises 98% of the total then,

$$\sigma_t = 1.03\,\sigma_1.$$

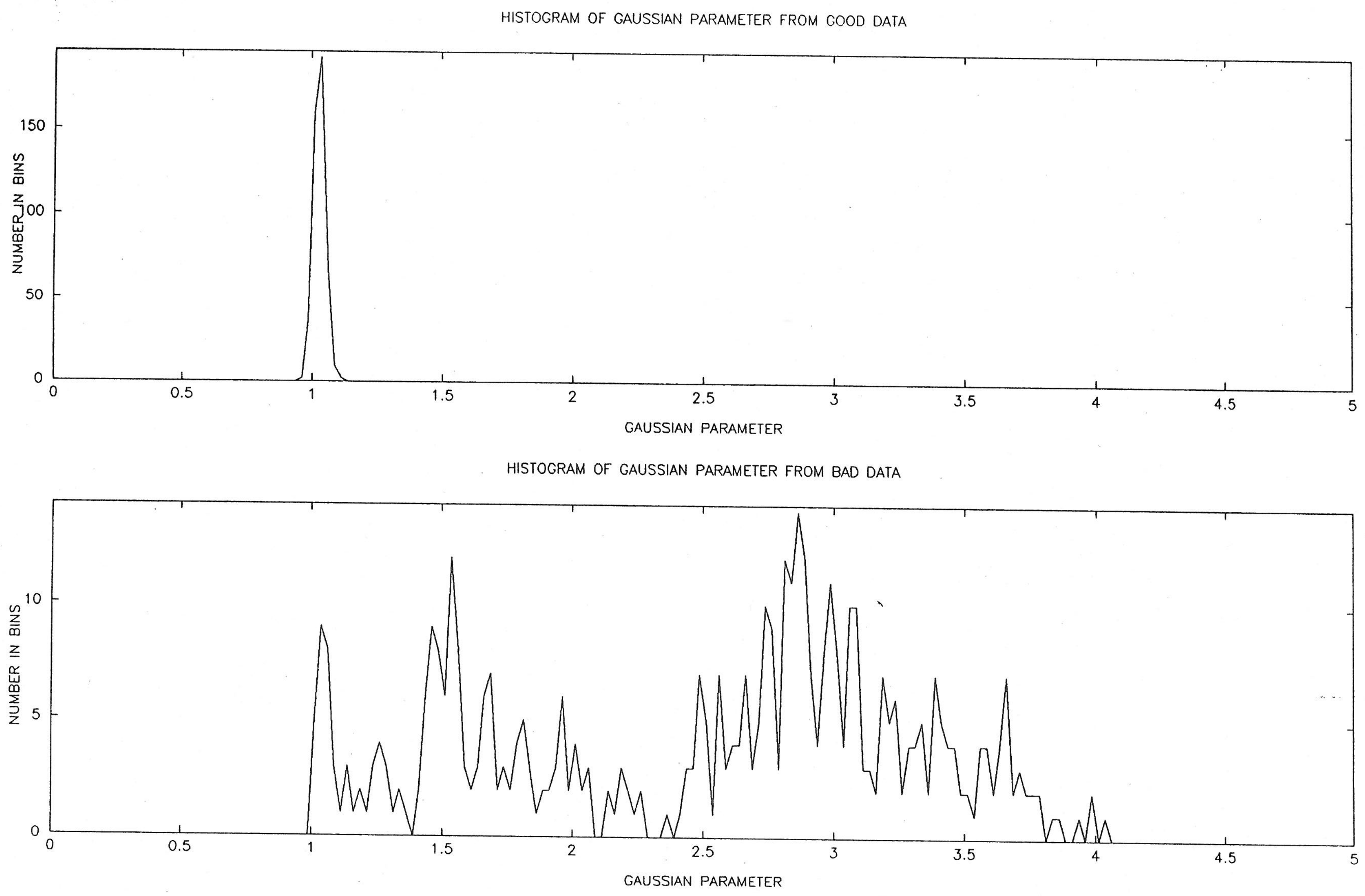
This is significant when examining the slope of the regions. The first region has a slope close to, but slightly less than, $-\frac{1}{2}$ which suggests that it is primarily *normal* Gaussian noise polluted by a small amount of noise with a higher $\sigma$ that becomes dominant in the higher displacement bins. In this instance the first region would inevitable have a slope less than $-\frac{1}{2}$ as $\sigma_t > \sigma_1$ and hence each bin interval, $\sigma_t/50$, would be slightly greater than the interval for purely Gaussian data, subsequently the first bins would inevitably collect more points than when just considering gaussian data, with each bin increased by more than the next, leading to a severer slope.

It should be noted that even the *good* data plots in Figure 6.6 seem to show some kind of contamination, as the slope of the logplot is not quite straight being slightly less than -1/2 before $3\sigma$ and slightly greater afterwards. This was seen to be true even for the very best data stretches seen. This contamination seems to be fundamentally different from that seen in the *bad* data, in at least extent and probably cause, as there is no apparent middle ground between good and bad data. Either the data is roughly Gaussian or strongly non Gaussian.

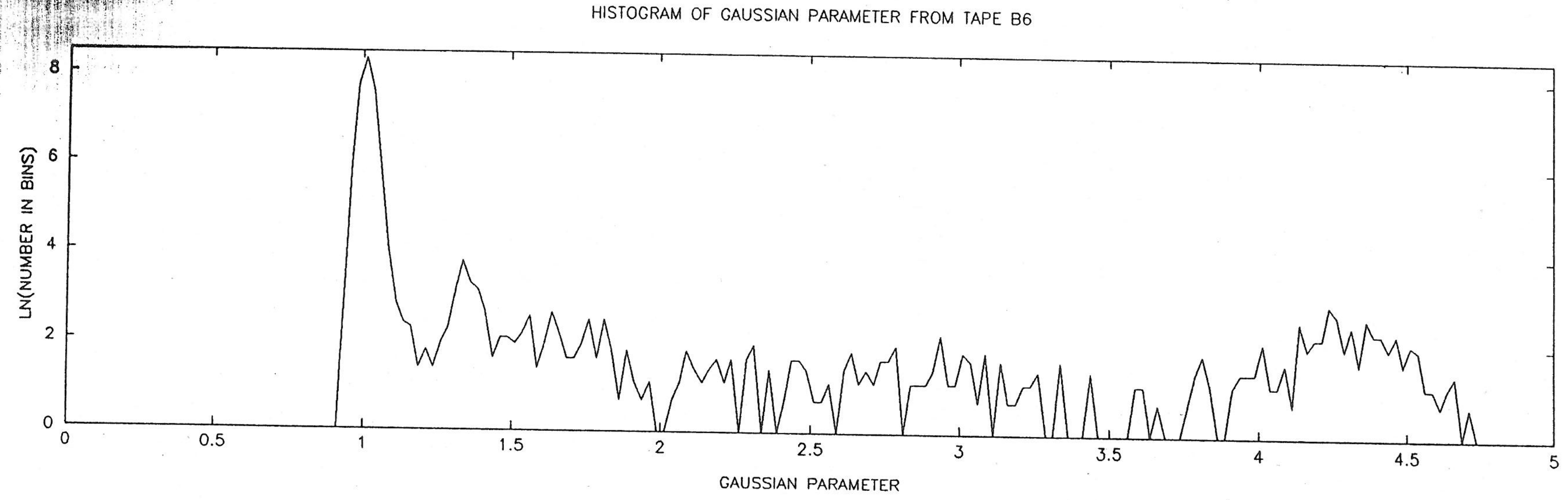
Figure 6.7 shows similar diagrams built up for filtered data, *ie* time-series data produced by the cross-correlation of the original time-series with one of the chirp filters. The left hand diagrams show a region in which the detector was well behaved, producing good noise, whereas the right hand diagrams cover a region where the original time series noise was known to be bad. Clearly there was a lot more high amplitude activity during the bad noise, but it shows no sign of the glitch at $3\sigma$ seen in Figure 6.6. This has some fairly significant implications, as the filtered data contains no frequencies above 1250Hz. This strongly suggests that the source of the contamination in the full time series operates at some higher frequency than this.

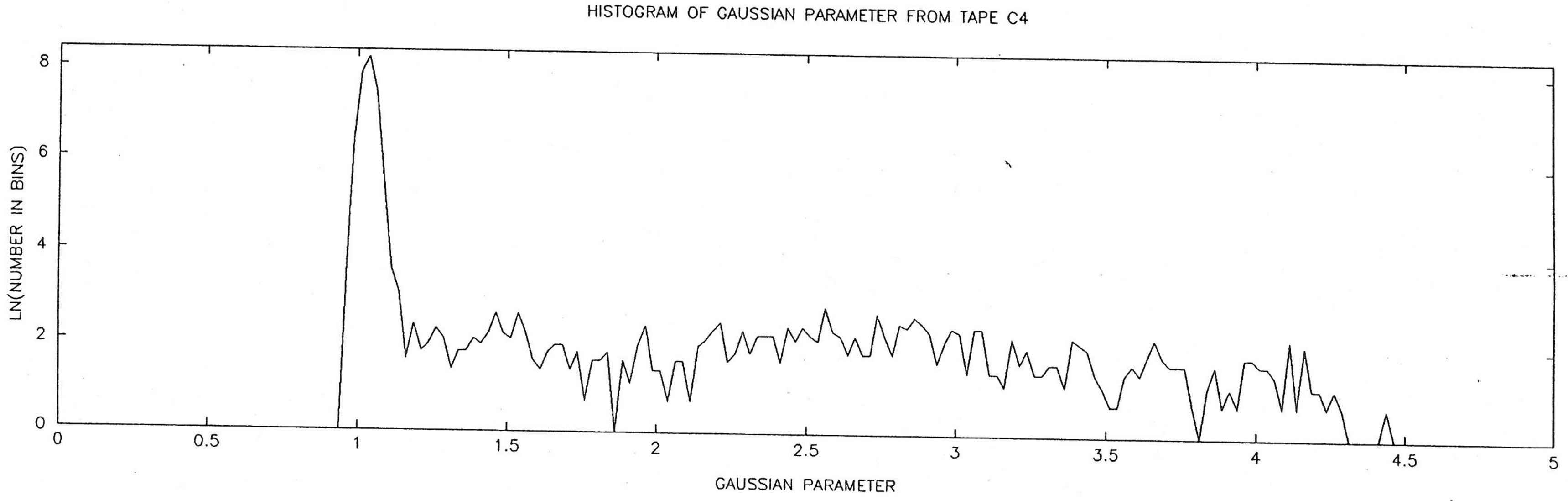
### Gaussian parameter

In order to assess the noise, *ie* deem it to be either good or bad, the Gaussian parameter was developed. As described in the previous Chapter, the parameter is calculated for each group on the basis of the first 5 histogram bins calculated for that group. The parameter is given by the total number of elements found to occupy these bins divided by the total number that should if the noise is Gaussian. Hence the nearness of the parameter to 1 would allow a decision about the *goodness* of



DISTRIBUTION OF GAUSSIAN PARAMETER FOR BOTH GOOD AND BAD DATA FOR TAPE C4:FIGURE6.8





DISTRIBUTION OF GAUSSIAN PARAMETER FOR BOTH TAPES B6 AND C4:FIGURE6.9

the noise.

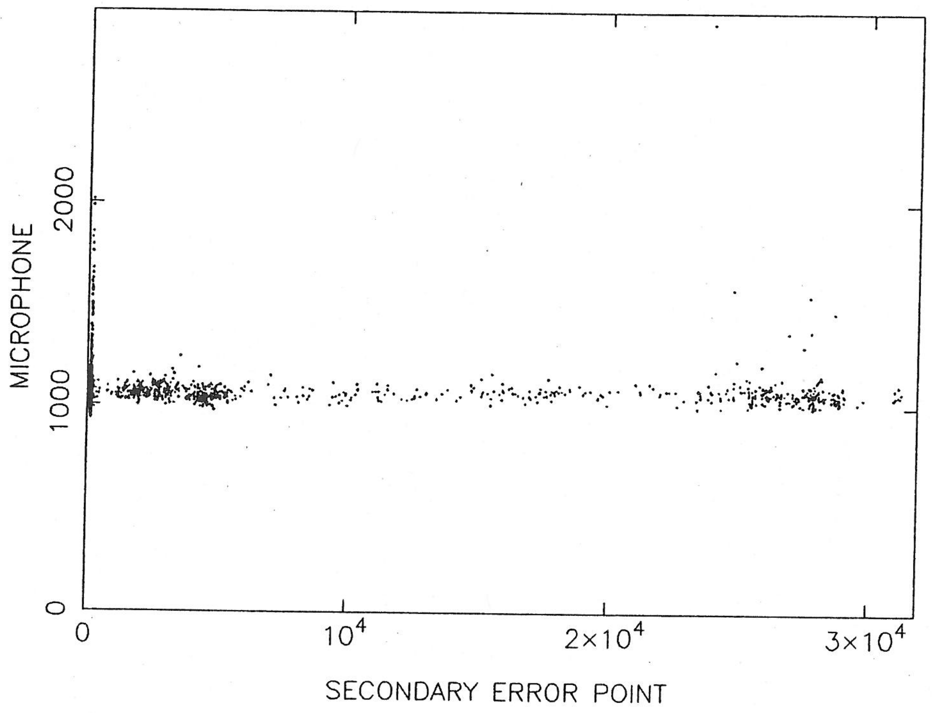
Figure 6.8 shows a histogram displaying the spread of the Gaussian parameter over both a region where the noise is known to be low (and hence good) and also a region where the noise is high (and hence bad). Each number on the y axis represents the sum of the number of parameters found to have values given by the numbers on the x axis. It is seen that for good data the parameter is always relatively close to 1, whereas for bad data it varies from just below 1 to beyond 4. This is consistent with Figures 6.1-6.4, where any significant changes in the Gaussian parameter occur abruptly, straddling areas of high noise in the internal housekeeping channels. These general characteristics were seen over all relevant stretches of good and bad data, from the whole 100 hour run. The Gaussian parameter could therefore be used, on its own, as a quick diagnostic assessment of the noise within the detector, and hence an immediate veto on events found in regions of bad noise within the detector.

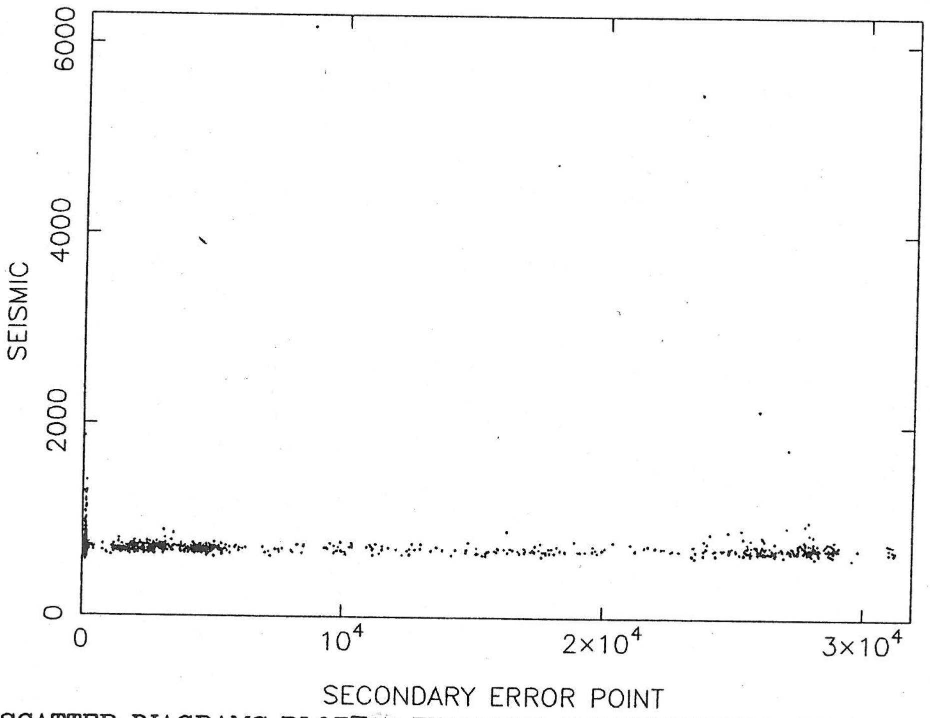
In Figure 6.8, it is obvious that the mean of the parameter is slightly greater than 1 for the good noise. This is consistent with the high $\sigma$ contamination speculated upon above. The first 5 bins would contain more than they should. Hence the parameter would be consistently forced above 1. Where the cutoff in the parameter between acceptable and unacceptable noise should be set is never going to be certain. However from studying the evolution of the parameter over the whole run it seems resonable to treat groups with parameters between 0.9 and 1.15 as good Gaussian groups. These estimates may well, on further analysis need to be expanded or contracted or even graduated fixing various levels of *goodness* within the upper and lower bounds. Figure 6.9 shows the histogram variations of the Gaussian parameter over the whole of both tapes B6 and C4.

## 6.2.4 External housekeeping

These streams are those tracking the detector's external environment, *ie* the microphone and seismic streams. From Figure 6.1-6.4 and indeed the rest of the tapes as well, it is fairly clear that neither seismic or audio activity have any significant influence on the large scale structure of the noise within the detector. This can also seen in Figure 6.10 that shows scatter plots of group standard deviations of both streams against the group standard deviation of the raw secondary error point data, taken from the whole length of tape C4. The streams vary along lines parallel to each axis, largely independent of each other. There are a few points high in both streams in each plot that suggest, that occasionally, high noise in the detector may be due to some external factor but this is certainly not the norm.

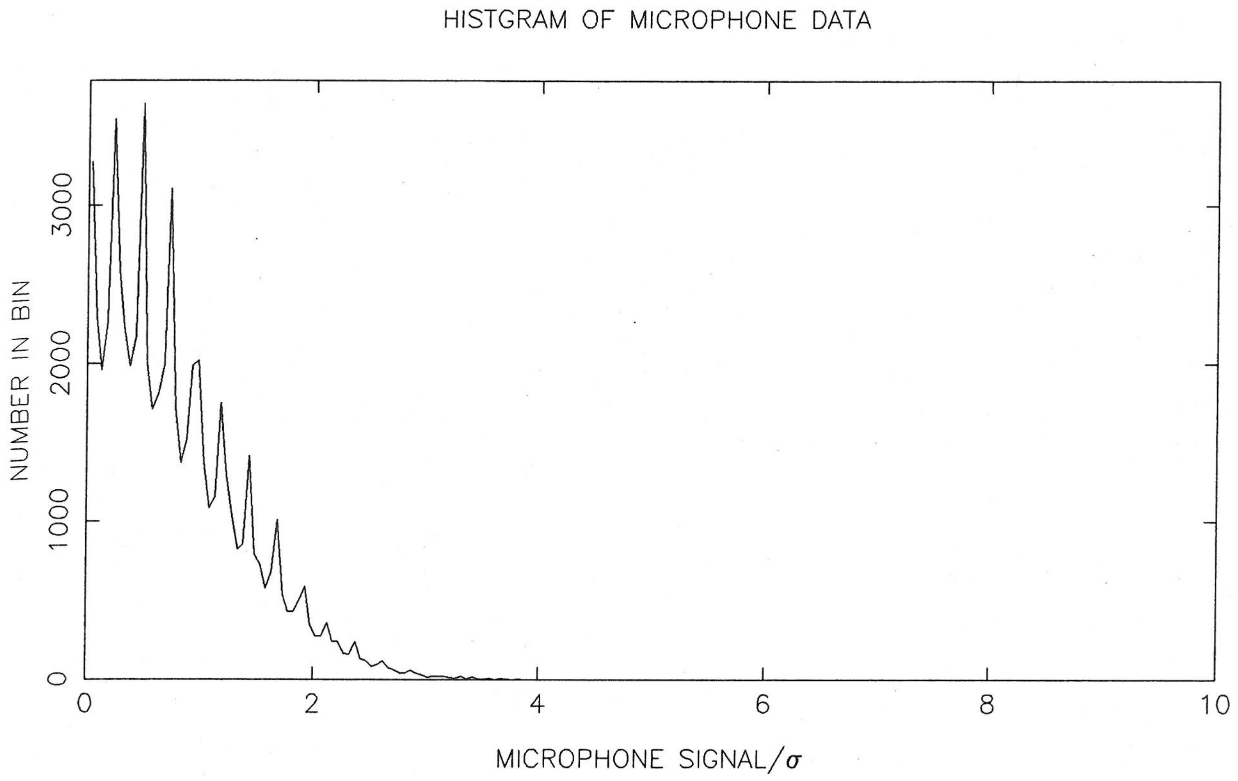
Figure 6.11 shows 2 plots concerning the microphone data. The first is a histogram that plots the

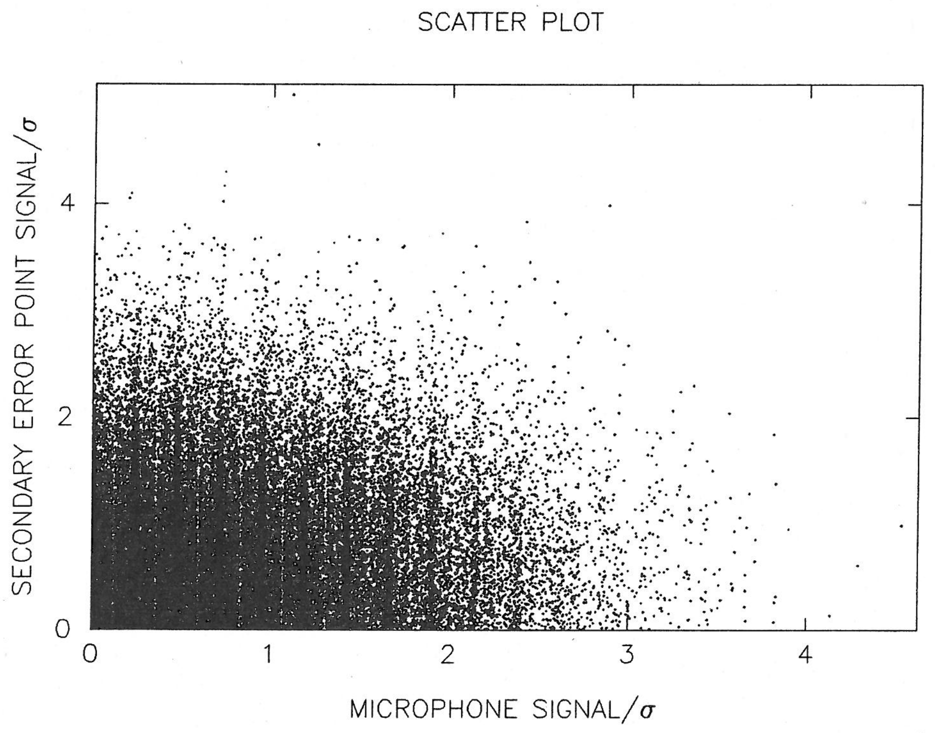




SCATTER DIAGRAMS PLOTTING EXTERNAL HOUSEKEEPING STREAMS

AGAINST RAW SECONDARY ERRORPOINT DATA: FIGURE6.10





**MICROPHONE SIGNAL: FIGURE6.11**

displacement of randomly selected microphone data points from its group mean along the abscissa against the total number of points found to be at certain displacements along the y axis. The plot is quite standard, with the envelope appearing roughly gaussian. The rapid oscillation from bin to bin results from the nature of the microphone signal which takes only a limited number of integer values, which artificially biases certain displacement bins. The second plot is a scatter diagram, plotting the displacement of raw secondary error point data from its mean divided by its group $\sigma$ against the simultaneous microphone data plotted divided by its group $\sigma$. This plot shows no obvious pattern of high amplitude events simultaneous in both streams.

Figure 6.12 shows similar plots based on the seismic data. Again the histogram appears roughly Gaussian and similarly the scatter diagram shows no obvious correlation between the two streams.

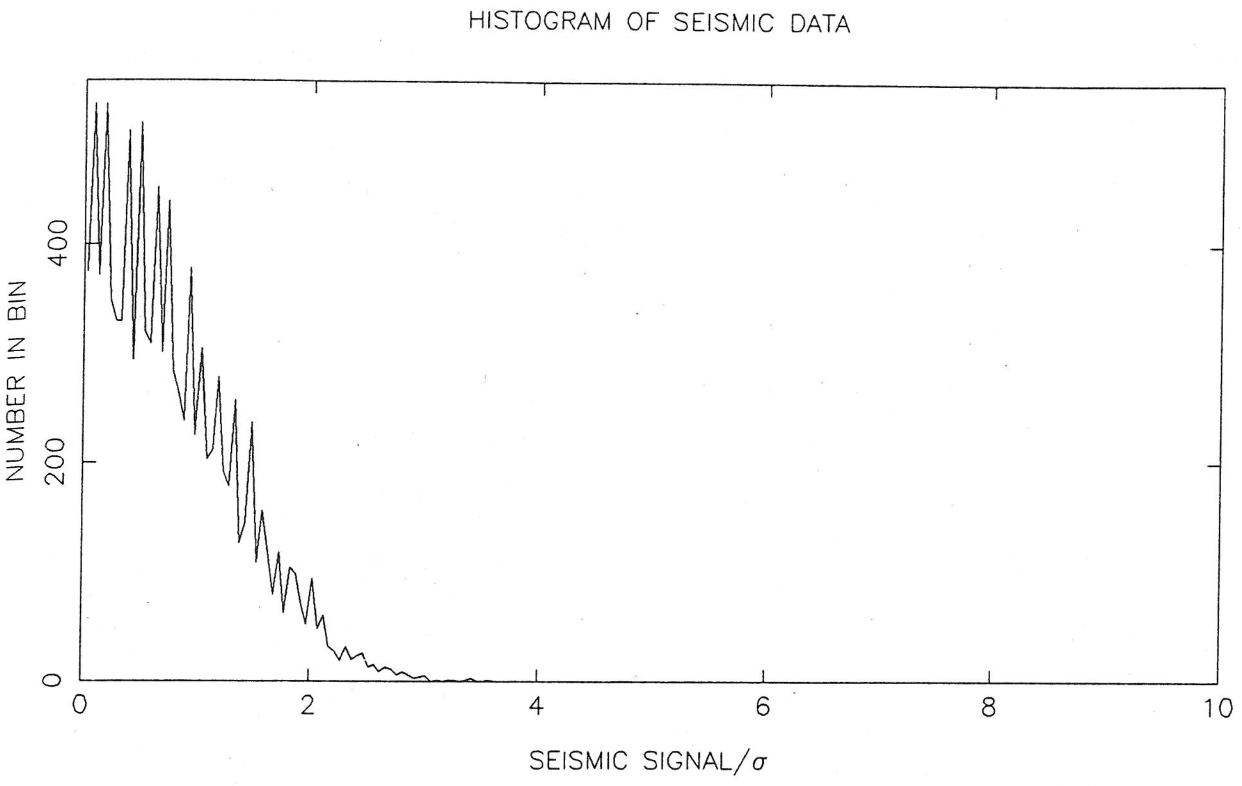
### Microphone analysis

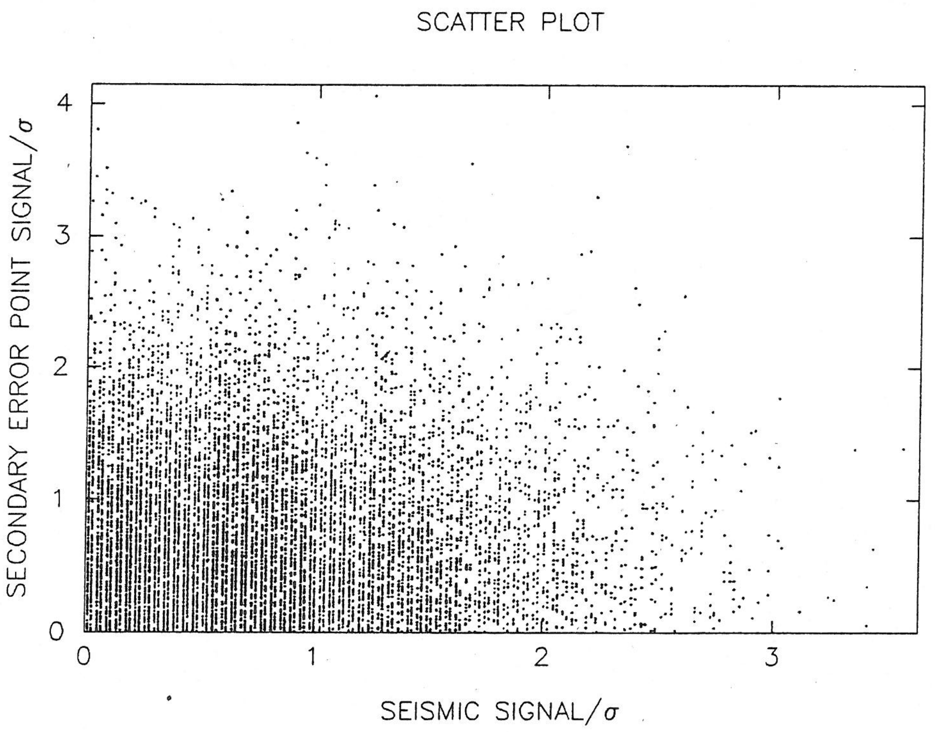
It was possible, using the Glasgow log of the 100 hour run, to attribute some of the peaks in the microphone signal to noises within the lab. This is demonstrated in Figure 6.13 which shows the microphone data taken from tape B6. The letters above some of the peaks indicate the source of the noise as follows. B:-computer beep within the lab, P:-phone in lab, D:-door in lab, DC:-distant crash, L:-lorry.

## 6.2.5 Events

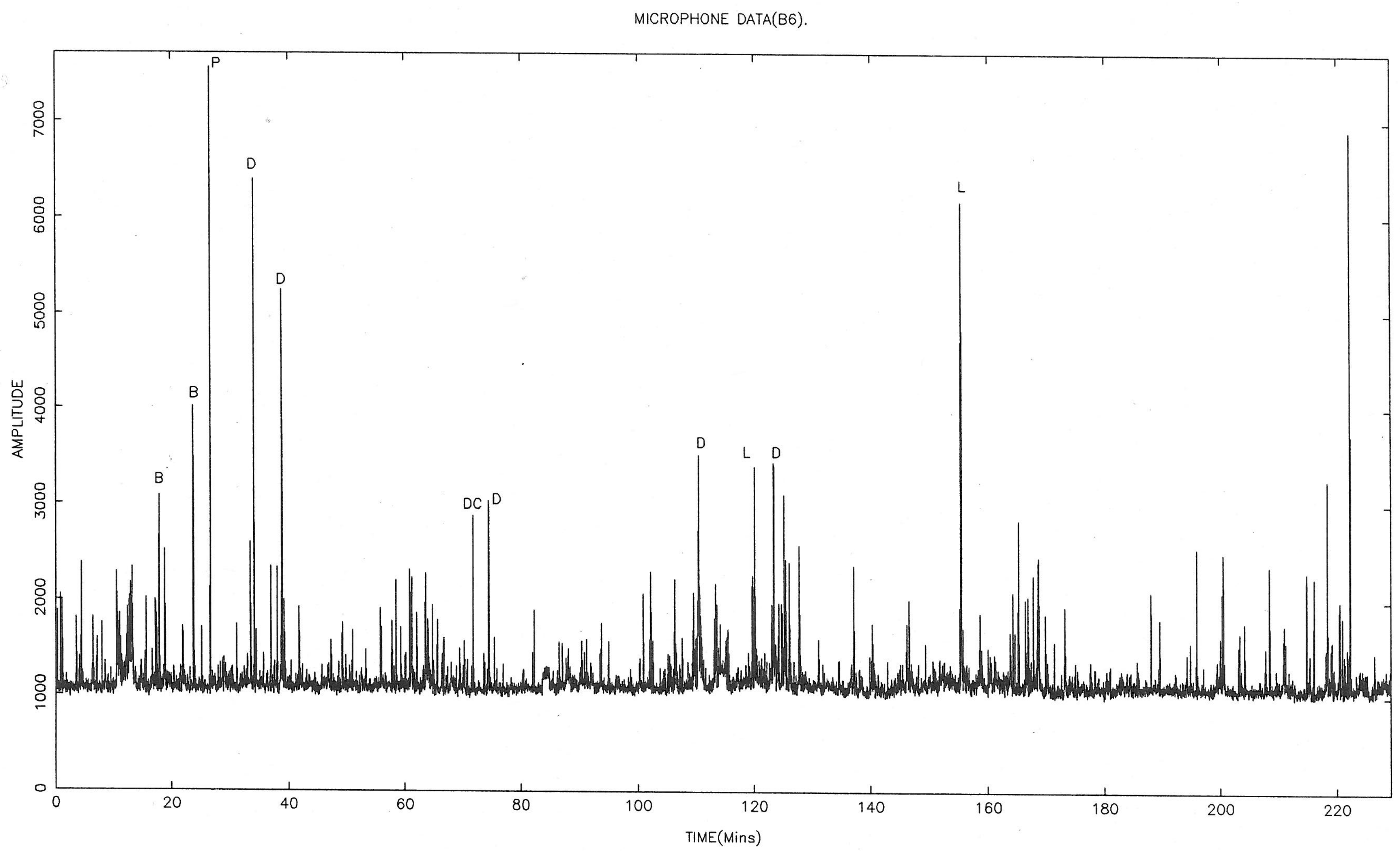
Events were found in the vast majority of the groups, more normally in the time series data where the events tended to be single point as opposed to the cross-correlation results that often came in adjacent groups of 2 points or more, but were considered all to be associated with the same event. The number of threshold crossing events found in each group was found to vary quite drastically with the noise. Figure 6.14 and Figure 6.15 are both histogram plots of the number of groups (of four blocks) against the number of events found from analysing the data within each of the groups. The plots given in Figure 6.14 show the results from a stretch of good data. The y axis represents the number of groups giving a particular number of events. The abscissa represents the number of events. The top graph gives the results of the time series analysis where the maximum number of events is only 19. The lower graph combines the results from both cross-correlation results set for each group. Therefore the maximum number of events possible is 32. The final bin in both graphs (the 19th and the 32nd) obviously represent saturation and means only that the number of events found reached or exceeded the maximum number possible to write to tape.

It can be seen from this Figure that, for good data, the majority of the groups, each 1.3104 seconds

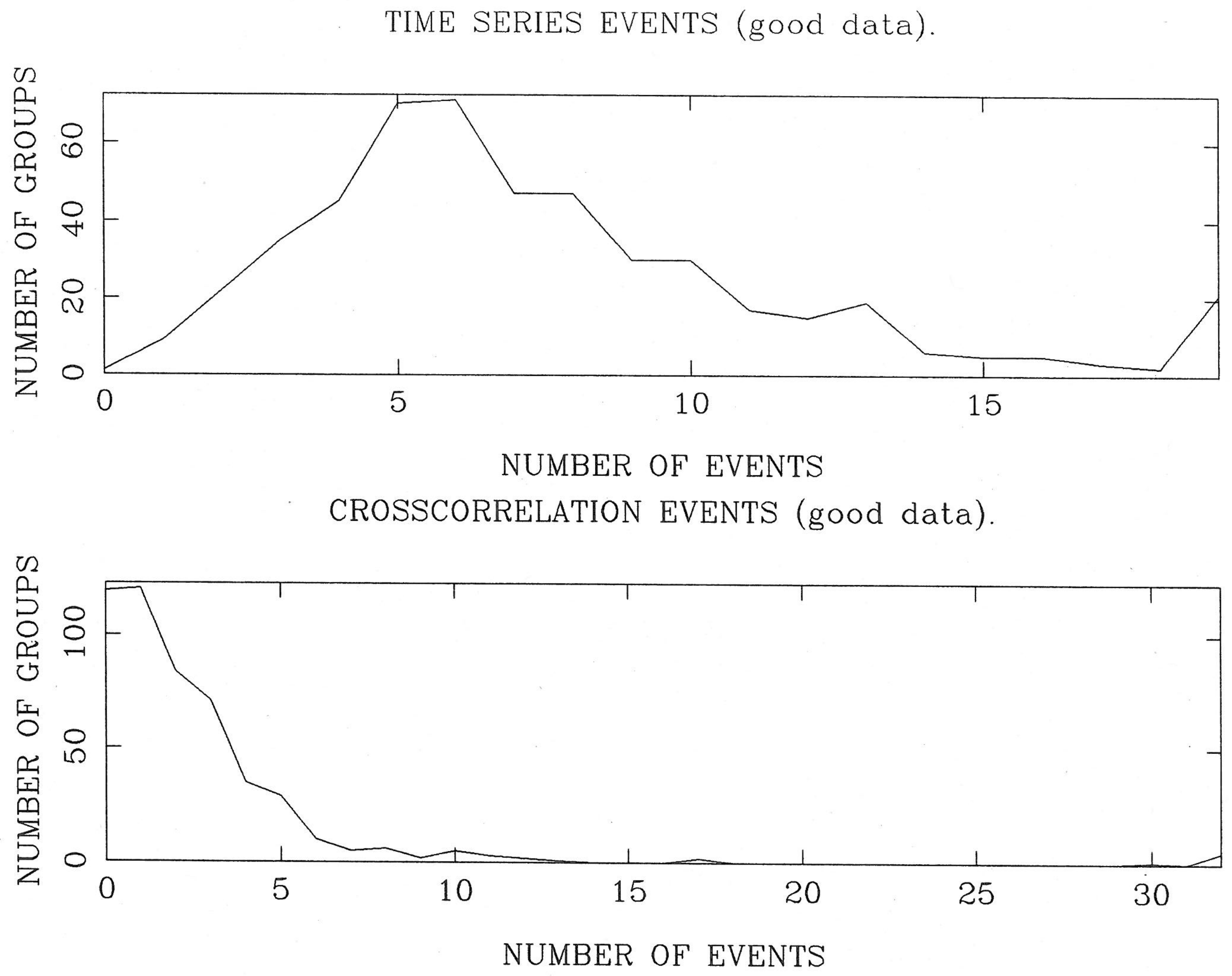




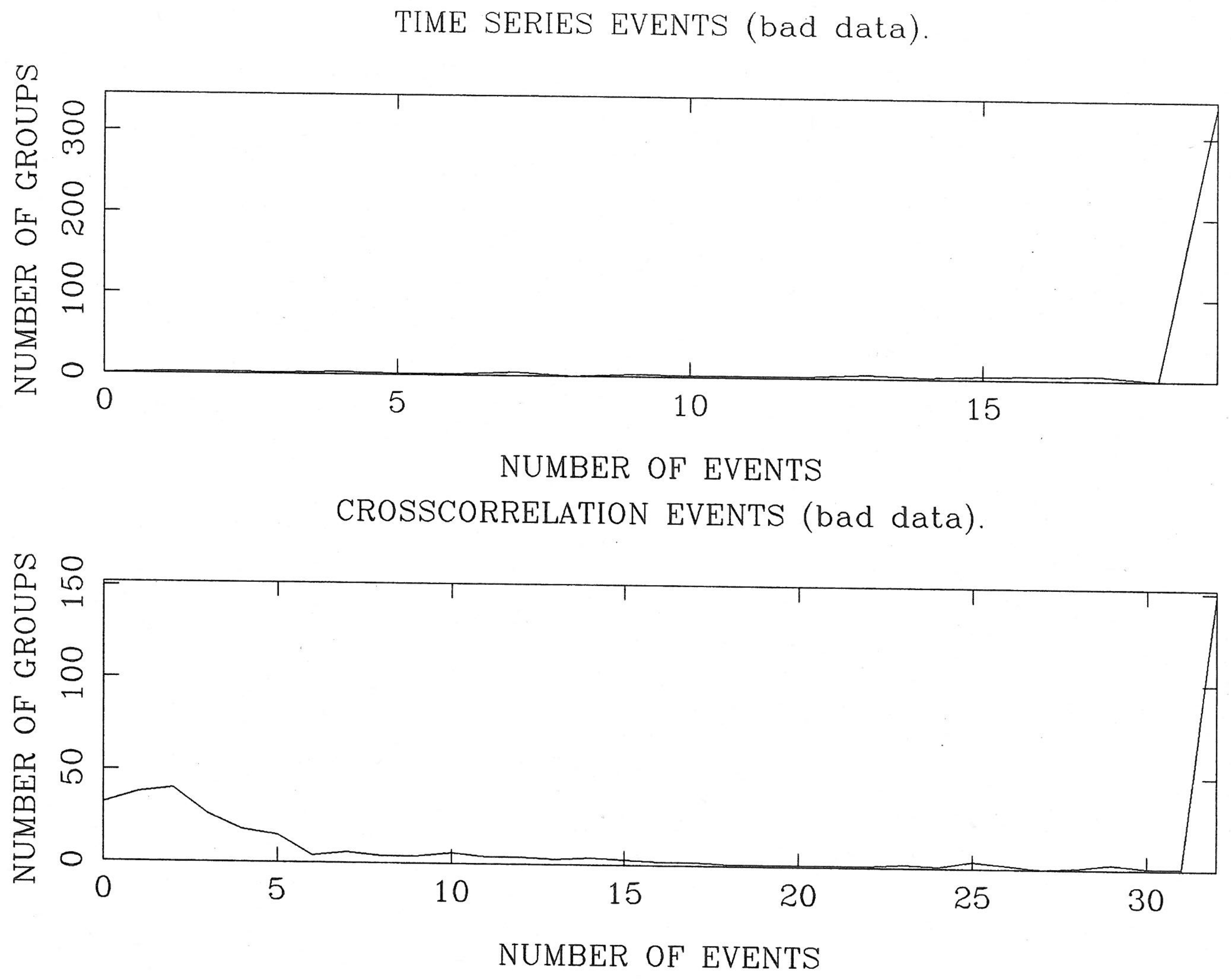
SEISMIC SIGNAL: FIGURE6.12



EVOLUTION OF MICROPHONE SIGNAL OVER TAPE B6:- (FIGURE6.13)



**HISTOGRAM OF THE NUMBER OF EVENTS FOR GOOD NOISE:- FIGURE6.14**



**HISTOGRAM OF THE NUMBER OF EVENTS FOR BAD NOISE:– FIGURE6.15**

long, contains roughly 5-6 threshold crossing events in the time series and 0-3 in the cross-correlation data. Figure 6.15 shows the equivalent graphs for a stretch of bad data. Here the average number of events in each group is much higher with 30 – 60% of the groups saturating their results array. It should be noted that even data that appears to be highly Gaussian produces more threshold crossings than true Gaussian data which on average would produce slightly less than 2.4 events per events group.

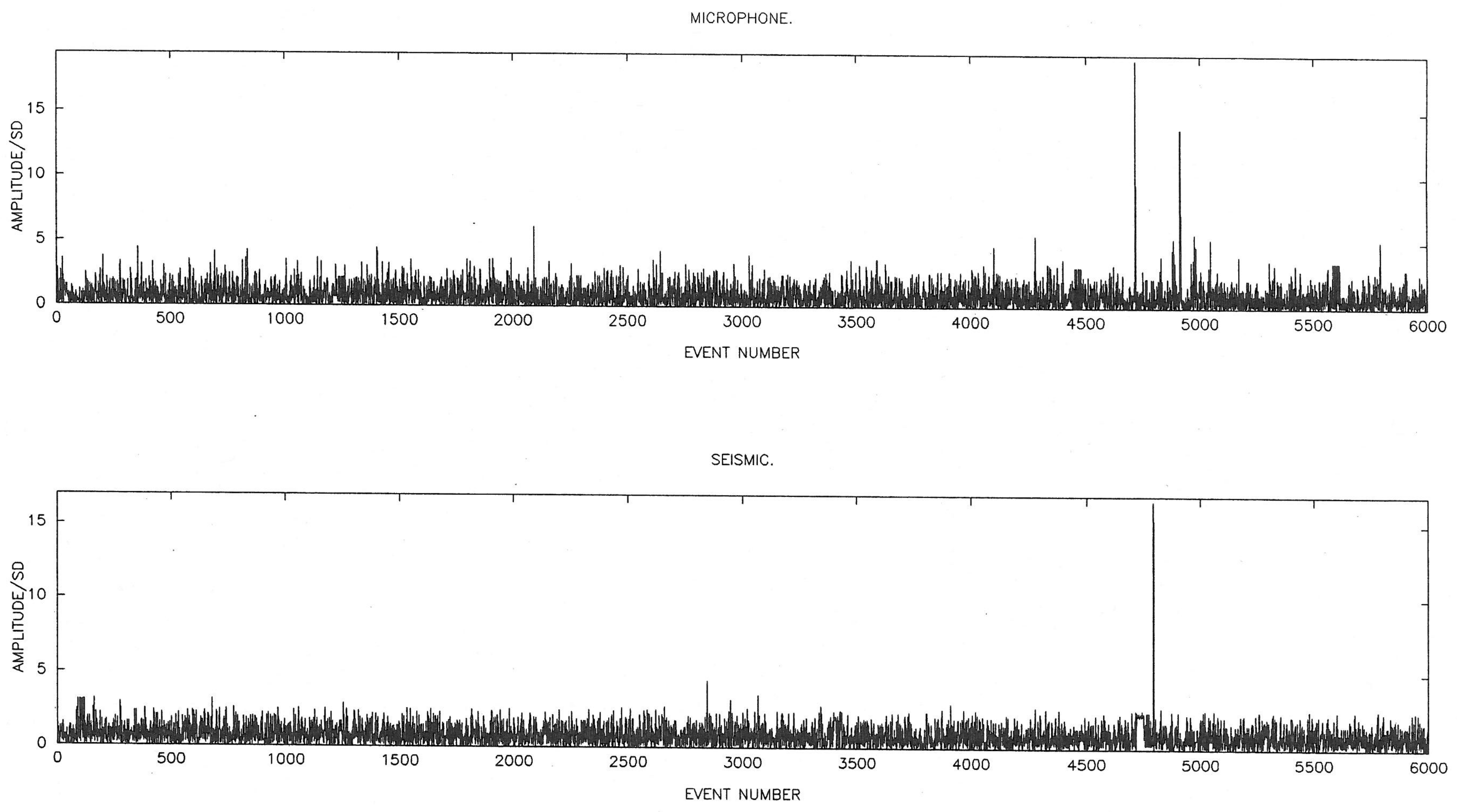
Figure 6.16 shows the relative heights of housekeeping streams at confirmed events in the secondary error point stream. The values displayed at each point are the amplitude of the point minus the mean of the stream and then divided by the stream's standard deviation, $\sigma$. It can be seen in both stream's that the amplitudes are primarily distributed fairly evenly around 1. However it can also be seen, particularly in the microphone stream, that occasionally the amplitudes do deviate considerably more than $\sigma$ from the mean. Both streams are also seen in Figure 6.17 in histogram form.

Working on the basis that the data that produced Figure 6.11 is typical microphone data, its examination revealed that of the 65536 samples in the data, only 3 exceeded $4\sigma$ which is less than 0.005% of the total. The microphone data in Figure 6.17, although largely similar in distribution to the data in Figure 6.11, showed 21 points in total from the 6000 where the deviation exceeds $4\sigma$. This represents 0.35%, from which it seems clear that occasionally events are caused by audio activity.

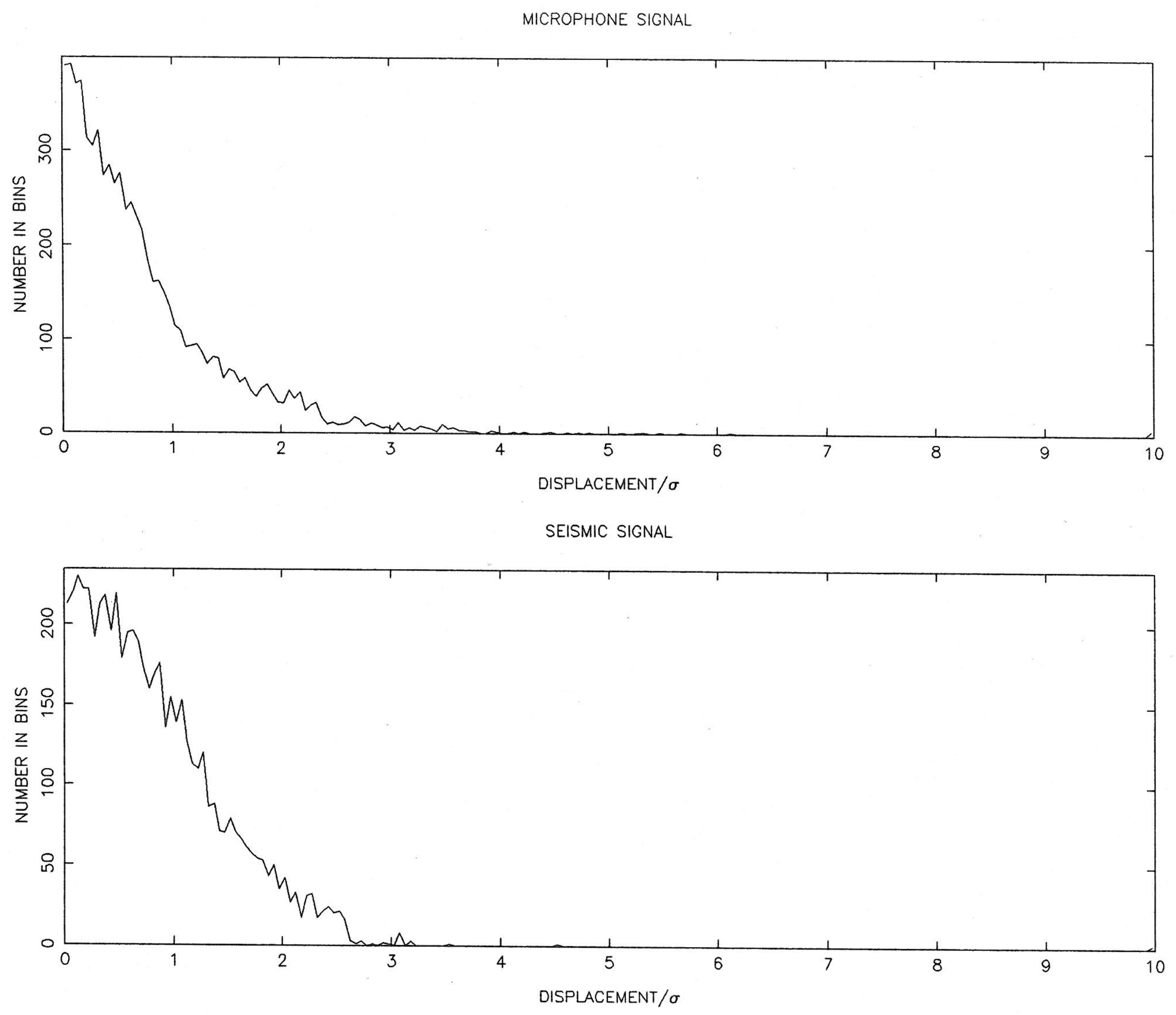
Similarly with the seismic data, Figure 6.12 shows no samples at all that exceed $4\sigma$, whereas the data plotted in Figure 6.17 shows 3. Of these, 2 points are adjacent in the stream and are preceded by an unusual region that casts some doubt on the integrity of the seismic data in this area and the third is only just above $4\sigma$. It seems unlikely then that seismic activity is systematically responsible for events.

### 6.2.6 Sensitivity of the detector

The dimensionless strain sensitivity $h$, corresponding to a level of $1\sigma$, in the detector noise was calculated individually for each group within the analysis. For Gaussian data, $h$ varied from about $0.8 \times 10^{-19} - 1.2 \times 10^{-19}/\sqrt{Hz}$.



RELATIVE HEIGHTS OF HOUSEKEEPING SAMPLED AT EVENTS. (FIGURE6.16)



HISTOGRAMS OF EXTERNAL HOUSEKEEPING SAMPLED AT EVENTS. (FIGURE6.17)

# Chapter 7

# SUMMARY AND CONCLUSIONS

In this thesis, I have described a system that analysed data collected by the 10 m prototype interferometer in the Department of Physics and Astronomy at the University of Glasgow. It searched this data, looking for several specific forms of gravitational wave, as well as compiling noise statistics. I discussed these potential sources in Chapter 2 and described the main mathematical techniques used in the analysis in Chapter 3. The data under examination here was briefly described in Chapter 4. In the rest of this Chapter I describe the main consequences of the software development described in Chapter 5 and the results of the analysis given in Chapter 6.

The gravitational wave data analysis system was developed to run automatically, without human interference, on a small parallel system. With its limited ambitions and negligible chance of actually finding any real event, it was intended only as a prototype from which those designing future systems may benefit. It was designed on a network of 5 transputers, to read the data directly from tape, analyse it and then write the results back to tape in as time efficient a manner as possible. This essentially meant ensuring that the duration of the tasks on the 4 Slave transputers, that did the actual analysis, did not exceed that of the task in the Root that had to both read the data from the tape and unpack it into its various constituent streams. Each group of data, of duration 1.3104 seconds, took about 30 seconds to pass through the whole network which, as the network was three tier, implied that the average time between completing one group and completing the next was about 10 seconds. This is roughly 7.5 times real time. This time was dictated by the Root task which spent some 80% of its time just separating the secondary error point data and the various housekeeping streams. This represented quite a significant overhead in the overall analysis. It was essentially a consequence of the fairly complex way the channels were multiplexed together as the de-multiplexing routines involved many complicated and time consuming bit operations. The Glasgow data is currently being read from the existing tapes and written to others, in a simpler format more conducive to rapid recovery. It is also likely future data streams sampled in and around gravitational wave detectors will not be written in this way.

The transputers themselves performed admirably, the transfer of data between them producing no significant or indeed measurable overhead even when quite large amounts of data were being passed. Their main drawback was their lack of memory, which meant that the de-multiplexing routines mentioned above were all effectively confined to the Root transputer as only the Root had both access to the de-multiplexing library and sufficient memory to contain all the housekeeping data.

Before the noise was searched in any way for the presence of a signal, it underwent two procedures in frequency space. The first removed any frequency bias within the detector and the second optimised the noise for cross-correlation with the coalescing binary filters. To allow an assessment of the extent of the frequency bias in the detector, calibration combs were periodically applied to the timeseries every 210 seconds. These combs revealed themselves in frequency space as a row of teeth of varying amplitudes at various equally spaced frequencies. The teeth were all originally applied with the same amplitude (except for the first which was 1000 times stronger). Knowing these amplitudes, the observed amplitudes of the teeth in the data were used to produce an inverse comb that when multiplied through the data removed any bias. This method was not wholly satisfactory mainly because of the nature of the noise. The noise was found to be very high at low frequencies, up to about 500-600 Hz, which contained the the first tooth, but then to drop quite abruptly to the region containing the second tooth. Subsequently in the inverse comb the amplitudes of the first two teeth differ quite considerably which, despite the linear interpolation between the two teeth, when applied to the noise, results in the connection between the two regions not appearing smooth. This is not as significant over the rest of the range as the noise changes only slowly. This situation could be improved by the addition of more closely spaced combs, particularly over the lower frequency regions.

The second procedure in which the data was optimised for cross-correlation, was a consequence of the matched filtering theorem. It required finding an average value for $S_h(f)$, calculated as a variance, over 128 narrow bandwidths each containing 128 complex points. Each point across the full bandwidth was then divided by the value of $S_h(f)$ found for the group in which it lies. This procedure is potentially destructive if there is some kind of structure in the noise on the same scale as the narrow bandwidth. This is distinctly possible as 128 points represents a bandwidth of nearly 80Hz. This method could be improved by not just building up the spectra over just one group, but many. This increases the number of points representing the same frequency range, allowing a significant decrease in the bandwidth over which the variance is calculated, while maintaining a reasonable number of points within each bandwidth. After this, of course, rather than dividing through the whole data stream by an abruptly changing series of histograms, a smooth curve could be fitted to the points.

After the calibration and weighting, noise statistics were compiled for the time series data. It was observed that the group root mean square value of the secondary error point data remained steady and low for long periods, occasionally abruptly leaping in value and becoming erratic. These periods always coincided with similar periods in those housekeeping channels reading directly from the detector and they were invariably found to be due to one of the laser cavities going out of lock.

The noise in the detector should ideally be Gaussian. I found, by studying histograms of amplitude statistics that in the steady low noise regions mentioned above, the noise was roughly but not quite Gaussian, with a few points showing some high amplitude deviation from the Gaussian the rest of the points seemed to describe. In the high noise regions however, the noise took on a different character, with many points making up a large non Gaussian tail beginning at about $3\sigma$. The straightness of both regions, before and after $3\sigma$, in the logplot suggested the presence of two Gaussian distributions in the noise, one with a significantly higher standard deviation than the other. An investigation of the possible sources of this tail is currently underway.

A single number was calculated for each group from that group's histogram data, by summing the number of points that lie between the mean and $\sigma/10$ and then dividing the result by the number of points that would occupy this region if the noise is Gaussian, the nearness of this number to one allowed an assessment of the extent of the Gaussian nature of the noise. This number, named the *Gaussian parameter* was shown to be a good diagnostic, correlating strongly with the secondary error point data as well as the relevant internal housekeeping data, giving numbers very close to, if slightly over, one for data that is roughly Gaussian and jumping to 3 or 4 for non Gaussian stretches. The analysis tended to show that the noise in the detector was either in one of these states or the other. The states were given as *good* when the detector noise was low and gaussian and *bad* when the noise was high and non-Gaussian.

Following on from the above, another important consequence of the gaussian parameter, is its implications for the housekeeping data. Significant variations in the gross structure of both the principle internal housekeeping streams, *ie* the primary error point and the secondary feedback data, showed up simultaneously or even before in the Gaussian parameter. This effectively renders these housekeeping streams obsolete as indicators of misbehavior in the detector. The same also appears to be basically true of the external housekeeping, notably the seismic and microphone signals, for completely the opposite reason. Fluctuations in these streams showed no definite correlation with either the internal signals or the Gaussian parameter. The possible exceptions to the above are the primary and secondary visibility signals that may possibly reveal bad noise even before the Gaussian parameter. This requires further investigation.

Each group was searched for time series wideband events that exceeded the group's mean by a preset multiple, in this case 4, of the group's standard deviation. The number of events found in each group tended to correlate strongly with the Gaussian parameter, with good noise producing an average of about 5-6 events per group and bad usually completely saturating the event array, *ie* producing more events than space was allocated to record. It should be noted that in the good data twice as many events were found as would have been the case for pure Gaussian noise, presumably due to whatever caused the slight deviation from Gaussian noise noted above. Each event was treated as being potentially part of a larger group of events. This turned out to be largely unnecessary as, in contrast to the German data, very few events were seen to be consecutive.

The first 1250 Hz were correlated with 2 templates of coalescing binary filters. Both of these templates were built up at the initiation of the analysis and stored in the transputer memory throughout. The resulting cross-correlation data was statistically analysed and found to be Gaussian over stretches of known good data, not showing the deviation from pure Gaussian found in the equivalent time series. This leads to the suggestion that the non-Gaussian component in the time series may be at higher frequencies than those present in the correlation, *ie* greater than 1000 Hz. As with the time series the number of events found varied with the Gaussian parameter. For good noise the most common number of events was in fact none at all, tapering off rapidly after about 3 or 4. For bad data, the event arrays were most frequently saturated. Again each event was treated as though it were part of a larger group of events and this time this was found, in general, to be the case, with far more multiple events of 2,3,4 or more points in a row than single threshold crossers. It also transpired that in both sets of events, a small proportion of the total number, were attributable to something that also showed up in one of the housekeeping streams, primarily the microphone stream.

Ultimately the routines analysed and reduced the data to $\frac{1}{50}$ th of its original volume. The results occupied approximately half of one exabyte tape, which represents about 0.7 Gbytes. Within this, there is undoubtedly some information that is, in retrospect, superfluous as well as other information that is inefficiently recorded. Doubts have to be raised over the usefulness of a large portion of the housekeeping data, indeed in future detectors there may be little point in recording some of these streams, particularly the internal housekeeping. It seems clear that a larger network of transputers or some similar parallel assembly could have completely analysed the data and indeed some parallel element is certain to be present in the final analysis system that reads data directly from the full sized detector. This analysis system will, hopefully in the not too distant future, enable the announcement of the first detection of gravitational waves.

# Bibliography

[1] Bondi,H. (1957), *Nature*, **179**, p1072

[2] Bondi,H. (1960), *Nature*, **186**, p535

[3] Bourzeix,S.,Boursin,G. and Linet,B. (1990), *Explicit determination of the distance to a coalescing binary from gravitational wave observations*, Physics letters A, **151**, no 8, p371-4.

[4] Bracewell,R. (1986), *The Fourier Transform and its applications*, McGraw-Hill, London.

[5] Clark,J.P.A. and Eardley,D.M. (1977), *Astrophysical Journal*, **215**, p315.

[6] Chandrasekhar,S and Detweiler,S.L. (1975), *Proceedings of the Royal Society of London*, **344**, p165.

[7] Davies,P.C.W. (1980), *The Search for Gravity Waves*, Cambridge University Press.

[8] Drever,R.W.P. (1983), in *Gravitational Radiation* ed. N.Deruelle and T.Piran, North-Holland, Amsterdam.

[9] Eddington,A.S. (1924), *The Mathematical Theory of Relativity*, 2nd edition, Cambridge University Press.

[10] Einstein,A. (1916), *Preuss. Akad. Wiss. Berlin, Sitzungsberichte der Physikalischmathematischen Klasse,* p688

[11] Einstein,A. (1918), *Preuss. Akad. Wiss. Berlin, Sitzungsberichte der Physikalischmathematischen Klasse,* p154

[12] Finn,L.S. (1989), in *Gravitational Wave Data Analysis*, editor B.F.Schutz, Kluwer Academic Publishers, Dordrecht.

[13] Forward,R.L. (1971), *Appl.Opt.*, **10**, p2495

[14] Hough,J. *et al* (1989) *Proposal for a joint German-British Interferometric Gravitational Wave Detector* Max-Planck-Institute für Quantenoptik

[15] Isaacson,R.A. (1968), *Physical Review*, **166**, p1263

[16] Kawamura,S. *et al* 1987, *10m Prototype for the Laser Interferometer Gravitational Wave Antenna*, ISAS Report No.637, p8.

[17] Krolak,A. 1989, in *Gravitational Wave Data Analysis*, ed, B.F.Schutz, Kluwer Academic Publishers, Dordrecht.

[18] Landau,L.D. and Lifshitz,E.M. (1941), *Teoriya Polya,* Nanka:Moscow

[19] Livas,J.C. (1987), Ph.D. Thesis, Massachusetts Institute of Technology, Cambridge, Mass.

[20] Meers,B.J. (1988), *Recycling in laser interferometric gravitational wave detectors,* Physical Review D, **38**, p2317-2326.

[21] Michelson,P.F. *et al* (1991), *Proceedings of the Elizabeth and Frederick White Research Conference*, ed. D.F.McClelland and H.-A.Bachor.

[22] Michelson,P.F. *et al* (1987), MonthlyNotices RAS, **227** p933.

[23] Misner,C.W., Thorne,K.S and Wheeler,J.A. (1973), *Gravitation*, W.H.Freeman and Co.:San Francisco.

[24] Muller,E. (1982), Astronomy and Astrophysics, **114**, p53.

[25] Muller,E. (1991), *Multidimensional hydrodynamical simulation of supernova explosions*, preprint.

[26] Newton,I. (1687), *Principia.*

[27] Penrose,R. (1963), Physical Review Letters, **10**, p66.

[28] Press,W.H., Flannery,B.P., Teukolsky,S.A. and Vetterling,W.T. (1986), *Numerical Recipes*, Cambridge University Press, Cambridge.

[29] Piran,T.H. and Stark,R.F. (1986), in *Dynamical Spacetimes and Numerical Relativity*, ed J.M.Centrella, p40 Cambridge University Press.

[30] Riemann,B. (1854), *The Habilitationsvorlesung.*

[31] Sathyaprakash,B.S. and Dhurandhar,S.V. (1991), Gravitational Wave Data Analysis of the Coalescing Binary Signal: A Criterion for Choosing Filters, to be published in Phys Rev.

[32] Schutz,B.F. (1984), *Am. J. Phys.* **52**, p412

[33] Schutz,B.F. (1985), *A First Course in General Relativity*, Cambridge University Press.

[34] Schutz,B.F. (1986a), in *Gravitational Collapse and Relativity*, ed. H.Sato and T.Nakamura, pp350-368, World Scientific, Singapore.

[35] Schutz,B.F. (1986b), *Determining the Hubble Constant from Gravitational Wave Observations*, Nature, **323**, p310-311.

[36] Schutz,B.F. (1990), in *The Detection of Gravitational Radiation*, editor D.G.Blair, Cambridge University Press.

[37] Strain,K.A. and Meers,B.J. (1991), *Wave-front Distortion in Laser Interferometric Gravitational Wave Detectors,* Physical Review D, **43**, p3117.

[38] Taylor,J.H. (1987), in *General Relativity and Gravitation,* editor M.A.H.MacCallum, Cambridge University Press.

[39] Taylor,J.H. and Weisberg,J.M. (1989), Ap.J. **345** p434

[40] Thorne,K.S. (1978), in *Theoretical Principles in Astrophysics and Relativity*, ed. N.R.Lebovitz, W.H.Reid and P.O.Vandervoort, p149, University of Chicago Press, Chicago.

[41] Wagoner,R.V. (1984), Astrophysical Journal, **278** p345

[42] Weber,J. (1960), *Physical Review* **117**, p306.

[43] Weisburg,J.M. and Taylor,J.H. (1984), *Physical Review Letters*, **52**, p1348.

[44] Weyl,H. (1922), *Space-Time-Matter,* Methen:London.

[45] Owa,S. etal (1986), Proceedings of the 4th Marcel Grossman meeting on General Relativity, ed. R.Ruffini, p571, North Holland, Amsterdam.

[46] Caves,C.M. etal (1980) Phys. Rev. Lett. 45, 75-8

## Appendix 1:- Testing the cross-correlation routines

### A.1 Introduction

In this appendix I briefly describe the diagnostic tests carried out on the routines running in two of the tasks, CORR1 and CORR2 that cross-correlate 1.6 seconds of secondary error point data with a template of filters built up and held within each of the tasks. In order to test the task routines I buried a variety of *artificial* chirps in Gaussian noise and then supplied the resulting data to a routine based around the relevant source code taken from the tasks.

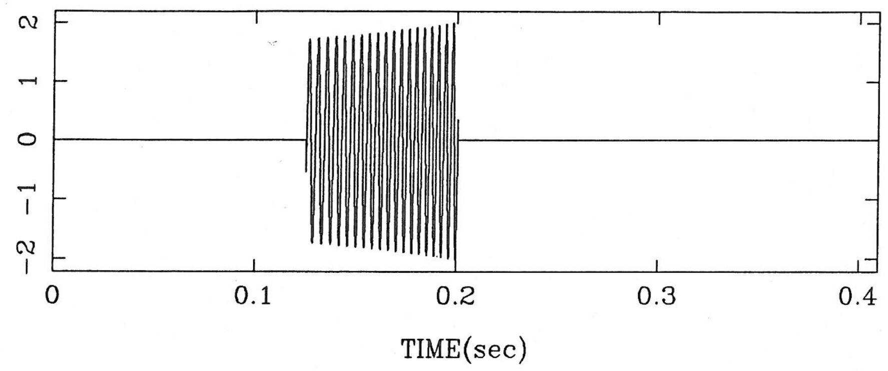
### A.2 Case 1:- signal directly proportional to filter

In this case, the filter $q_\alpha$ and the artificial signal, given by $Hq_\alpha$ are defined on the basis of the same mass parameters and initial phases, where $H$ is simply a linear factor representative of and proportional to the amplitude of the signal. This would certainly be very unlikely in any real observations but serves as a means of comparison when considering the various cases when the signal and filters are not defined on the basis of the same parameters.

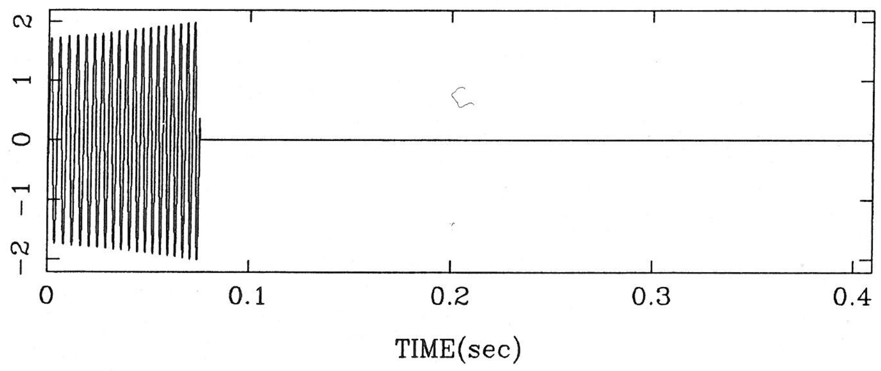
Figure A.1 shows a chirp and filter respectively. Both were constructed on the basis of a mass parameter of 2 and zero phase. The non zero parts of both therefore are identical apart from some multiplicative factor starting at 300 Hz and climbing to 1000 Hz. When the filter and chirp are cross-correlated the result, which in this instance is effectively an auto-correlation also shown in Figure A.1, gives a distinct peak at the time where the chirp starts.

When the same chirp is buried in noise, cross-correlation becomes the only means of its detection. Figure A.2 shows the chirp starting at 0.65 seconds. The Gaussian noise was artificially generated by routines found in Numerical Recipes by Press *et al*(1986). When the chirp is added to noise whose standard deviation equals the chirp's maximum value and is sampled at the same rate, the chirp is still just about noticeable within the noise to the unaided eye but would probably not distinguish itself in anyway to a simple threshold crossing search routine. When the data, (signal + noise), was cross-correlated with the filter template a number of the filters gave peak threshold crossings at the right time with the highest for the filters with mass parameter 2 as expected which gave a peak S/N of 8.89. Figure A.2 also shows the cross-correlation data for each of the filters with a mass parameter of 2. Both give very obvious peaks at the correct time with the correlation with the filter of phase 0, as would be expected, being slightly higher than the correlation with the filter of phase $\pi/2$. The deepest that any chirp was buried in noise and still picked up by cross-correlation giving a S/N ratio of 3.5 (the ratio chosen in the analysis), was for a chirp based on a mass parameter of 1 with 0 phase whose maximum amplitude was only 9% of that of the noise.

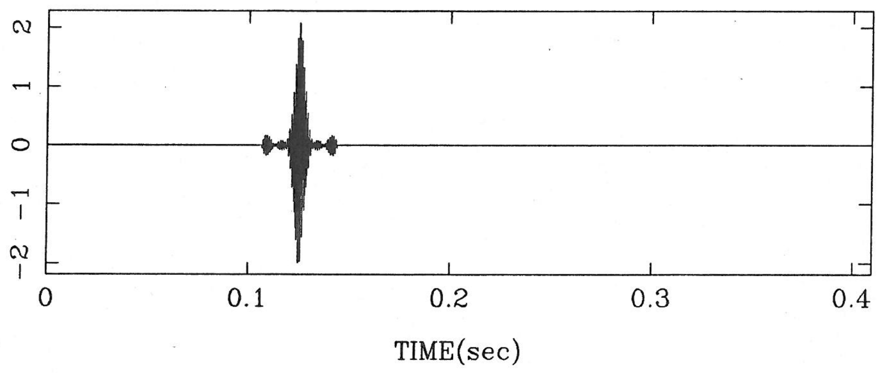
CHIRP. (Mpar=2.0, Phase=0.0)



FILTER. (Mpar=2.0, Phase=0.0)



CORRELATION OF NOISE WITH FILTER.



CLOSE UP OF ABOVE.

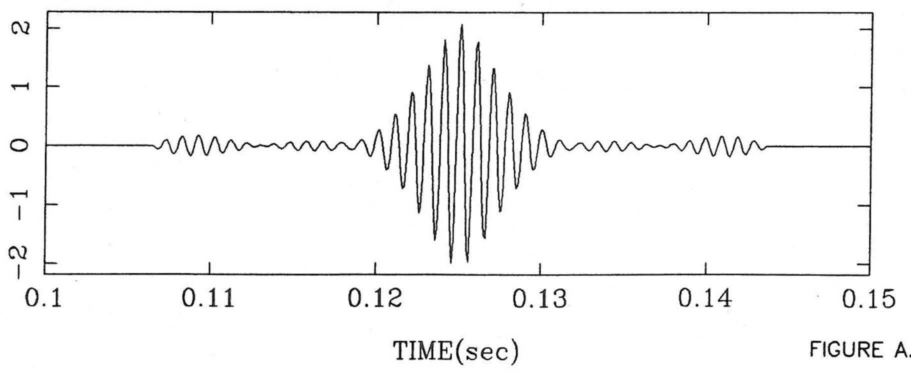
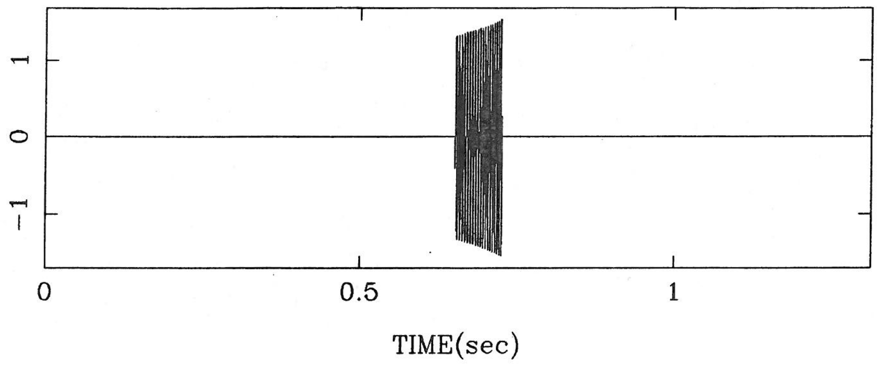
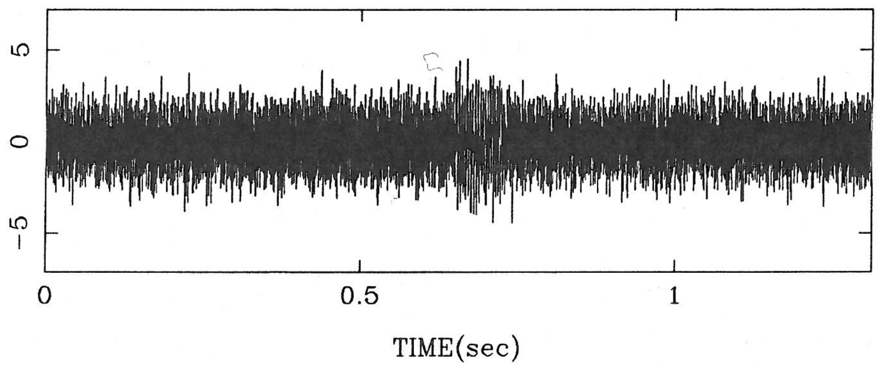


FIGURE A.1

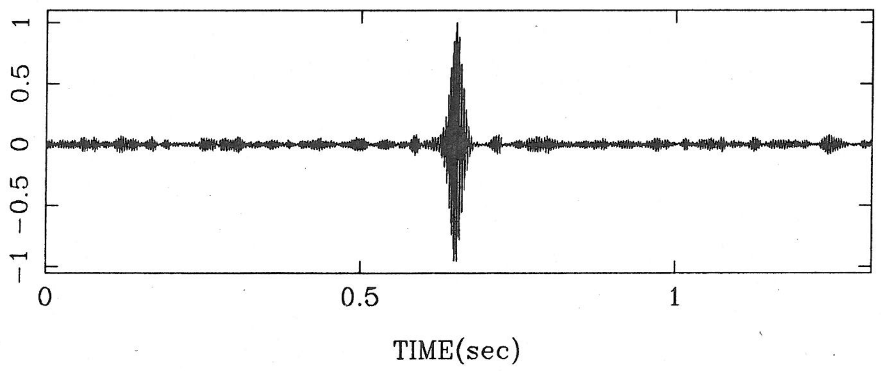
CHIRP. (Mpar=2.0, Phase=0.0)



CHIRP IN NOISE. ($\sigma_{noise}$ = chirp max)



CORRELATION OF NOISE WITH FILTERS OF Mpar=2.0, Phase=0.0



CORRELATION OF NOISE WITH FILTERS OF Mpar=2.0, Phase=1.57049

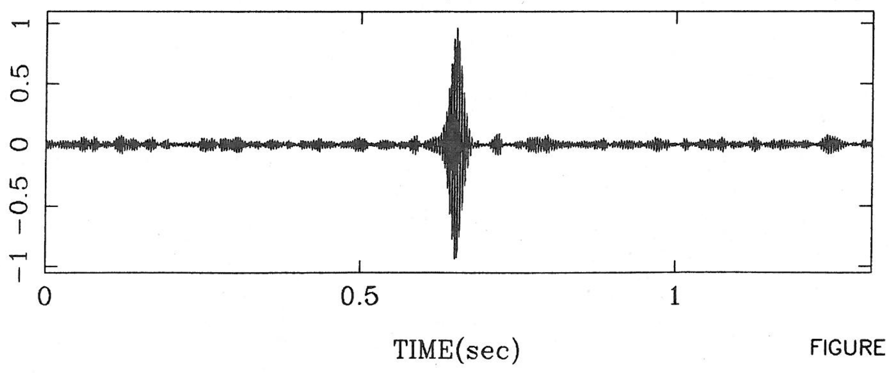


FIGURE A.2

## A.3 Case 2:- General chirp in noise.

Any chirp picked up in a data group could not be expected to be an exact match with any of the filters built up and held in the template. They would differ in both mass parameter and intial phase, the most significant consequence of which being the resultant uncertainty in the position of the event in the group from which is calculated the event's time of arrival.

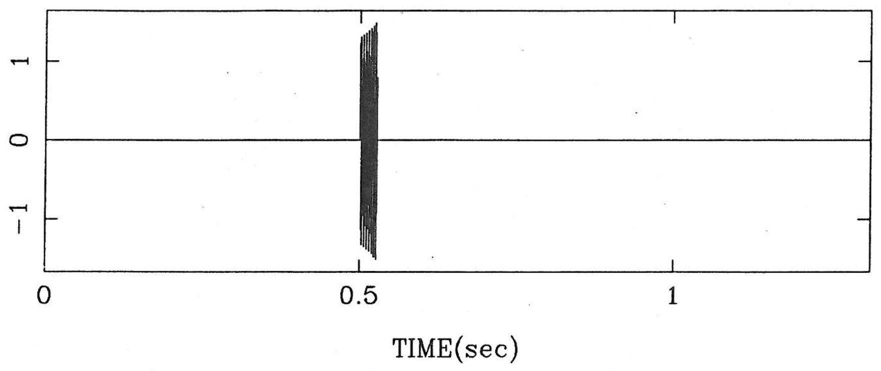
### Mass parameter.

Figure A.3 shows a chirp based on the mass parameter 5.5, 0.5 seconds into the group, which was then buried in Gaussian noise with the standard deviation of the noise 4 times the maximum value of the chirp. When correlated with the filter template the correlation streams gave peak S/N of 3.82 and 3.74 for the filters based on the mass parameters 5 and 6 respectively. These correlation streams are also shown in figure A.3. It can be seen that the two peaks are displaced slightly either side of the correct time. This displacement in the time of arrival was first demonstrated by Schutz(1986a). It is a consequence of the lengths of the filters. The non-zero part of the filter based on the smaller mass parameter is longer than the actual chirp in the noise whilst covering the same frequency range hence the whole correlation is spread out as the higher frequency regions meet and correlate before the lower frequency areas whereas with identical chirps the peak correlation occurs instantaneously as the same frequency regions in either chirp meet at the same time. The peak is shifted below the true event position because the higher frequency, higher amplitude region contributes more to the correlation than the lower frequency and amplitude regions hence as the higher region correlations occur first, the peak is shifted.

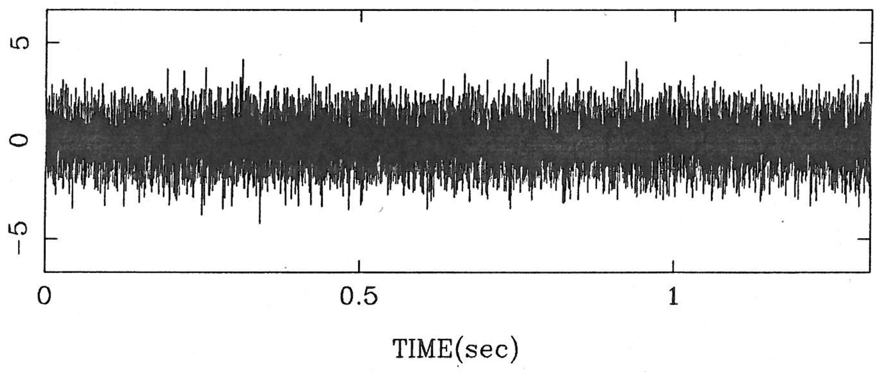
The opposite case occurs with the higher mass parameter. The non zero part of the filter is shorter than the chirp in the noise hence the peak correlation produced when the high frequency regions pass over each other occurs after the lower frequency correlation causing the peak correlation to be shifted beyond the true event position.

As the mass parameter of the signal was varied, the signal's general detectability was seen to worsen the further the mass parameter of the signal got from a mass parameter on which a filter was based. This is shown in figure A.4 where the peak S/N in the correlation streams for 5 signals is shown. The mass parameters vary from 3 to 4 as shown on the abscissa and their maximum values are 18% of the standard deviations of the noise. It can be seen that the peak S/N at both 3 and 4 exceed 3.5 and hence would be picked up by the analysis routines but at a mass parameter of 3.5 the peak S/N, (*ie* the larger of the peaks obtained from the correlations with filters 3 and 4), dips slightly below 3.5 and so would be missed. In general the peak correlation obtained for signals that

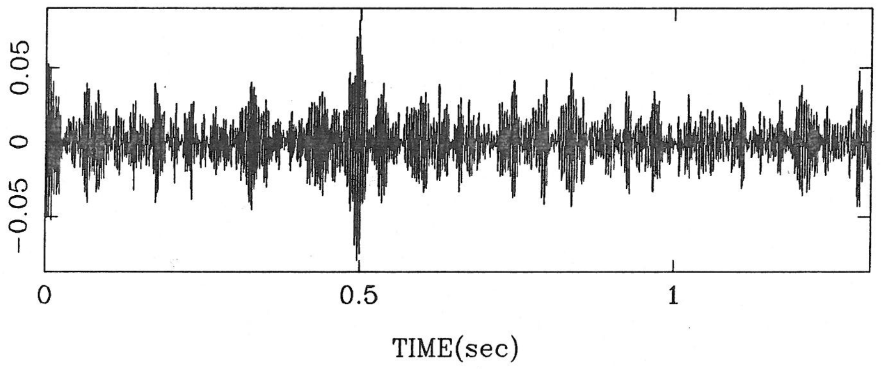
CHIRP. (Mpar=5.5, Phase=0.0)



CHIRP IN NOISE. ($\sigma_{noise}$ =4 × chirp max)



CORRELATION OF NOISE WITH FILTERS OF Mpar=5.0, Phase=0.0



CORRELATION OF NOISE WITH FILTERS OF Mpar=6.0, Phase=0.0

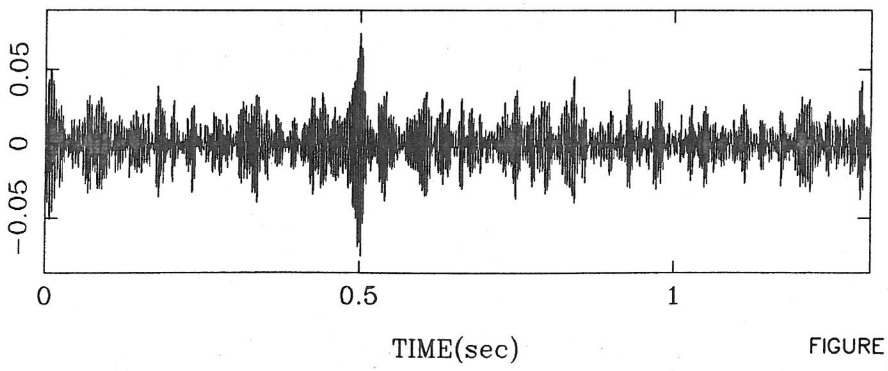


FIGURE A.3

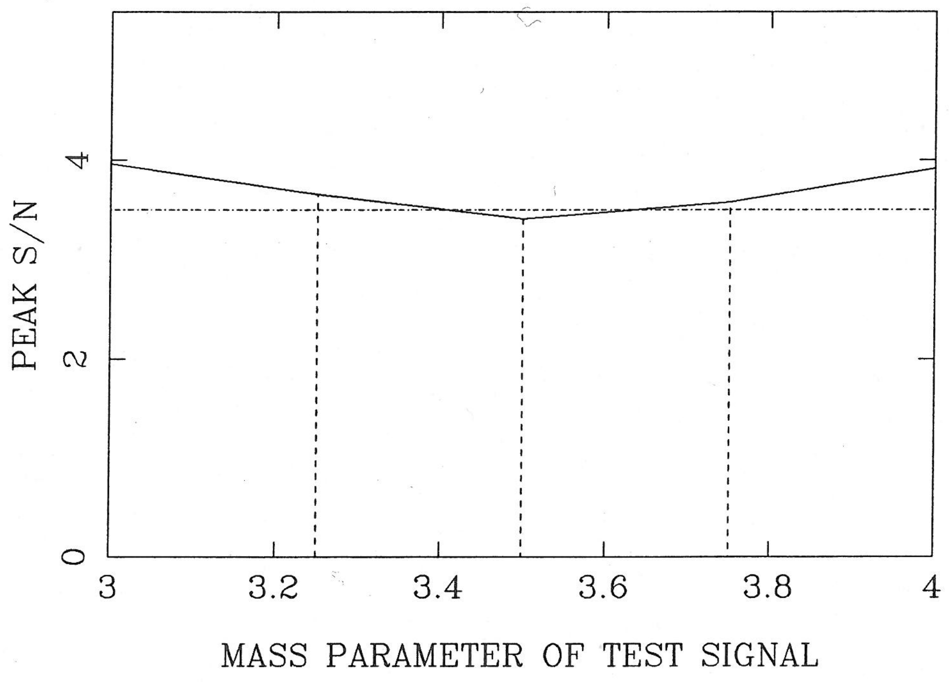


FIGURE A.4

fall half way between two filters is 70 – 95% of that obtained when the signal falls directly on one of the filters. The higher the mass parameter the smaller the difference.

In the tests carried out, the largest displacement of a maximum peak from the true position of the event when the signal and filter that gave the peak differed by a mass parameter of 0.5 was 225 points which represents 0.01125 seconds which is over twice the size of the light travel time, of 0.0045 seconds, between Glasgow and Garching that defines the coincidence window. This, of course, could lead to coincident events being missed if each event from each filter is treated separately but if events in successive filters that arereasonable close together are regarded as due to the same event then plotting a best fit curve through several of the associated peaks could enable *true* positions and maximum S/N's to be found. This is shown in figure A.5 which shows a quadratic curve fitted through the three peak correlations for one particular signal in noise It is seen that the curve peaks just beyond point 10000 which is where the chirp was buried in the noise. It should be noted that in tests, a curve through 3 points occasionally did not improve the estimate of the events position however, in these cases, fitting a quartic or an even higher order polynomial (if possible) frequently did.

Figure A.6 shows two related plots essentially both showing how the peak S/N varies with mass parameter. The upper plot shows the change in peak S/N with mass parameter for signals buried equally deep in the same stretch of noise. Its seen that higher the mass parameter the higher the peak S/N. This follows from theory outlined in Chapter 3 that relates the S/N to the number of cycles in the signal and hence also its time duration and total energy. The relationship can be expressed as follows,

$$S/N \propto \sqrt{\text{number of cycles}} \propto \sqrt{\text{time}} \propto \sqrt{\text{energy}}.$$

This is shown in the lower plot.

**Phase.**

The analysis failed to correctly estimate the phase of any of the signals other than those that were defined on almost precisely the same phase and mass parameters as those that defined the filters. This is shown diagrammatically in Figure A.7 which shows the results of cross-correlating 9 chirps with the same mass parameter but varying phase with a pair of filters with the same mass parameter. The plot shows the phase of the chirps calculated by the analysis against their actual phases. This *error* in the calculated phase is minimal only when the signal is a linear multiple of one of the filters. In the future if it is possible at all to accurately estimate the phase of any coalescing binary event it is going to have to be by a far more sophisticated method than employed in this work.

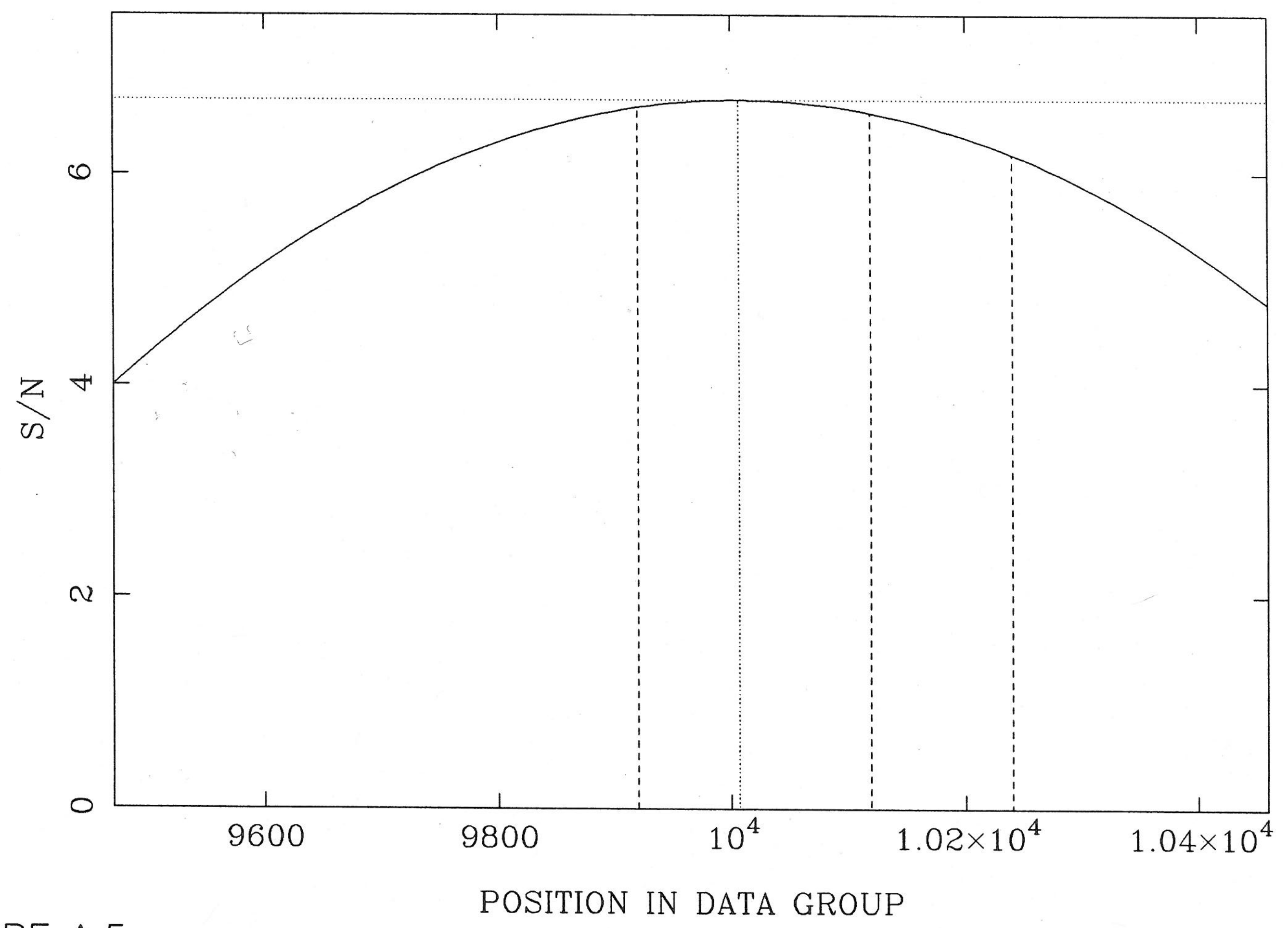
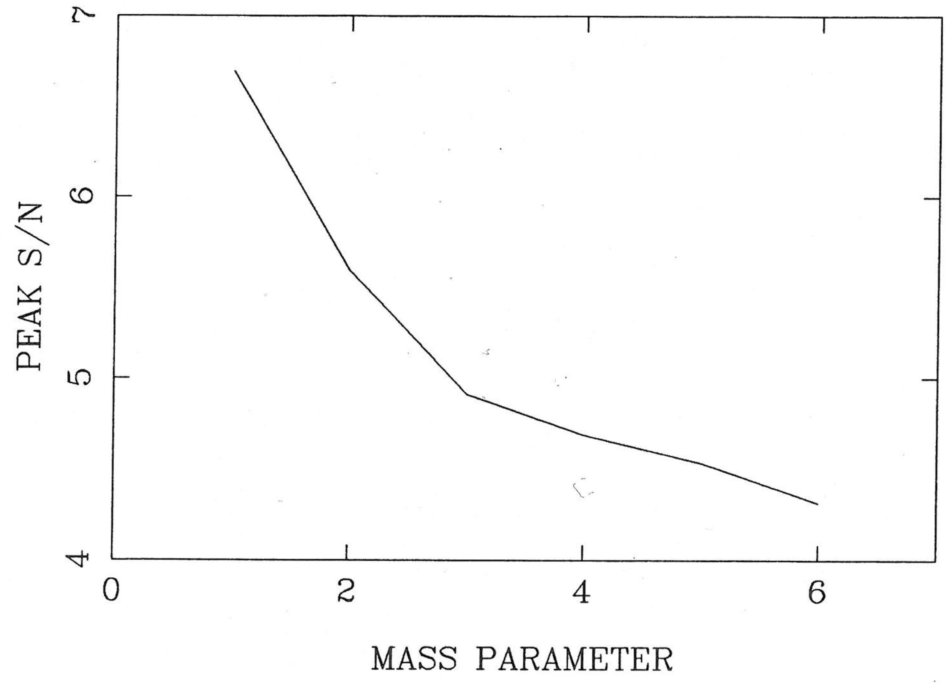

S/N
0
2
4
6
9600
9800
$10^4$
$1.02\times10^4$
$1.04\times10^4$
POSITION IN DATA GROUP

FIGURE A.5


PEAK S/N
4
5
6
7
0
2
4
6
MASS PARAMETER

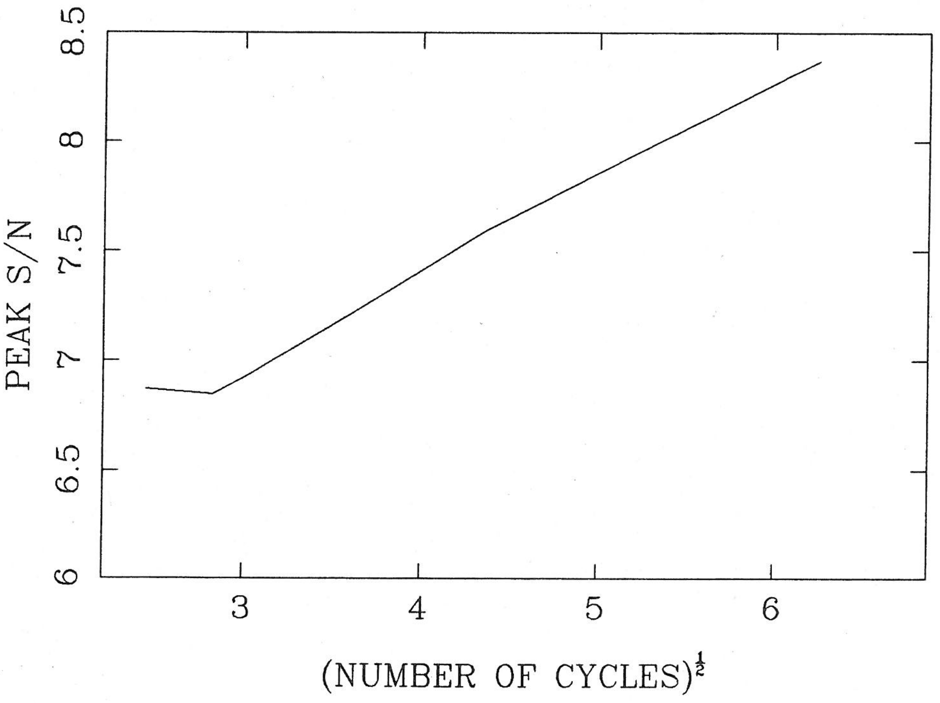
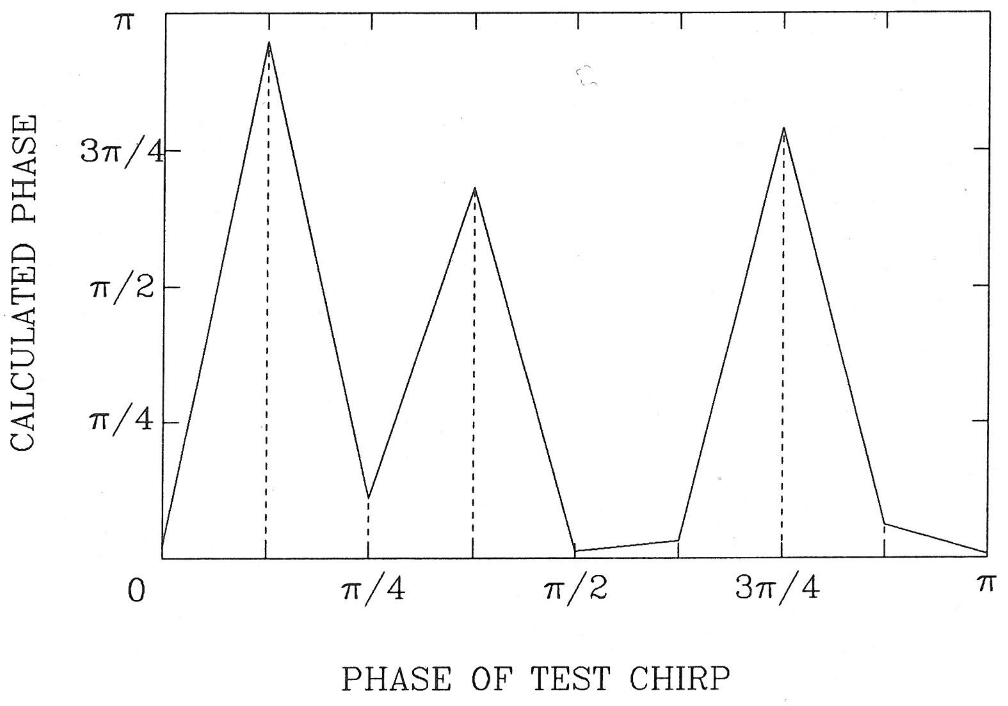

PEAK S/N
6
6.5
7
7.5
8
8.5
3
4
5
6
(NUMBER OF CYCLES)$^{\frac{1}{2}}$

FIGURE A.6

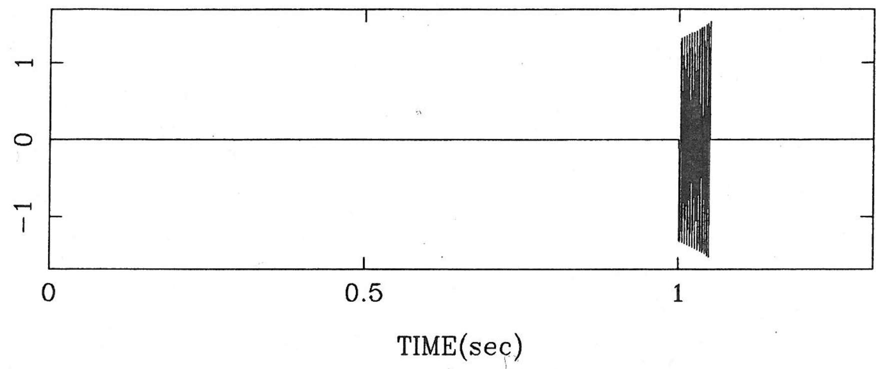


$\pi$
$3\pi/4$
$\pi/2$
$\pi/4$
0
$\pi/4$
$\pi/2$
$3\pi/4$
$\pi$
CALCULATED PHASE
PHASE OF TEST CHIRP

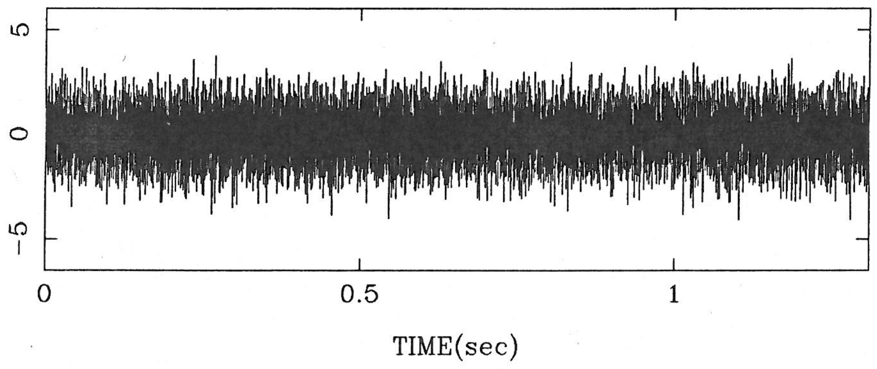
FIGURE A.7

The four plots given in Figure A.8 show a chirp built up with a phase of $\pi/4$ which is then buried in noise whose $\sigma$ is 2.5 times that of the chirp and also the correlation of this chirp in noise with the two filters with the same mass parameters. It is seen that the correlations are virtually identical in form with very similar amplitude peaks.

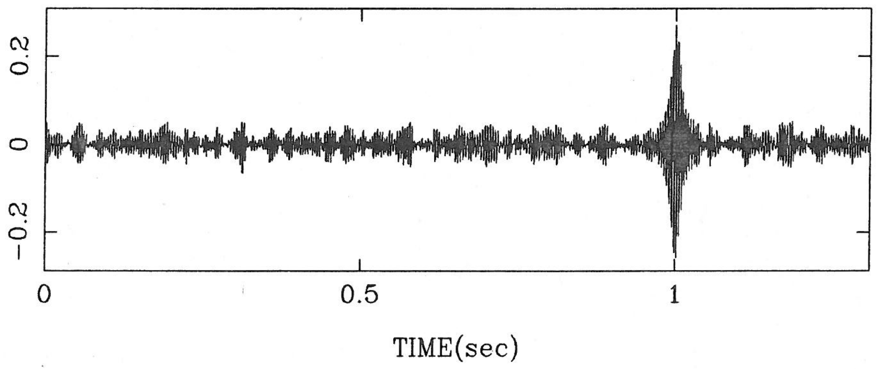
CHIRP. (Mpar=3.0, Phase=0.785245)



CHIRP IN NOISE. ($\sigma_{noise}$ =2.5 × chirp max)



CORRELATION OF NOISE WITH FILTERS OF Mpar=3.0, Phase=0.0



CORRELATION OF NOISE WITH FILTERS OF Mpar=3.0, Phase=1.57049

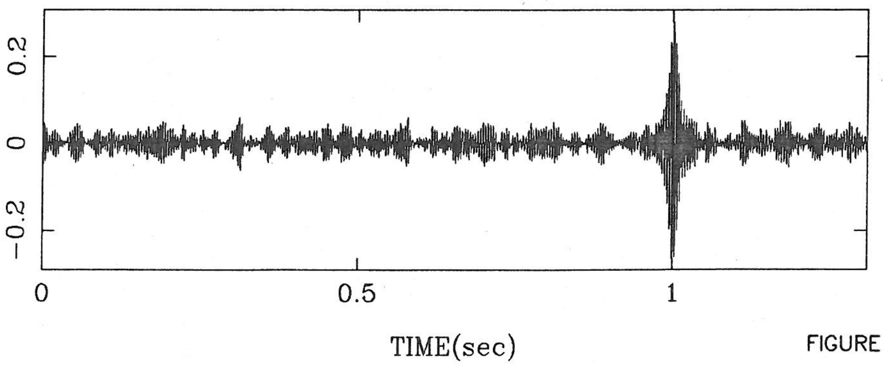
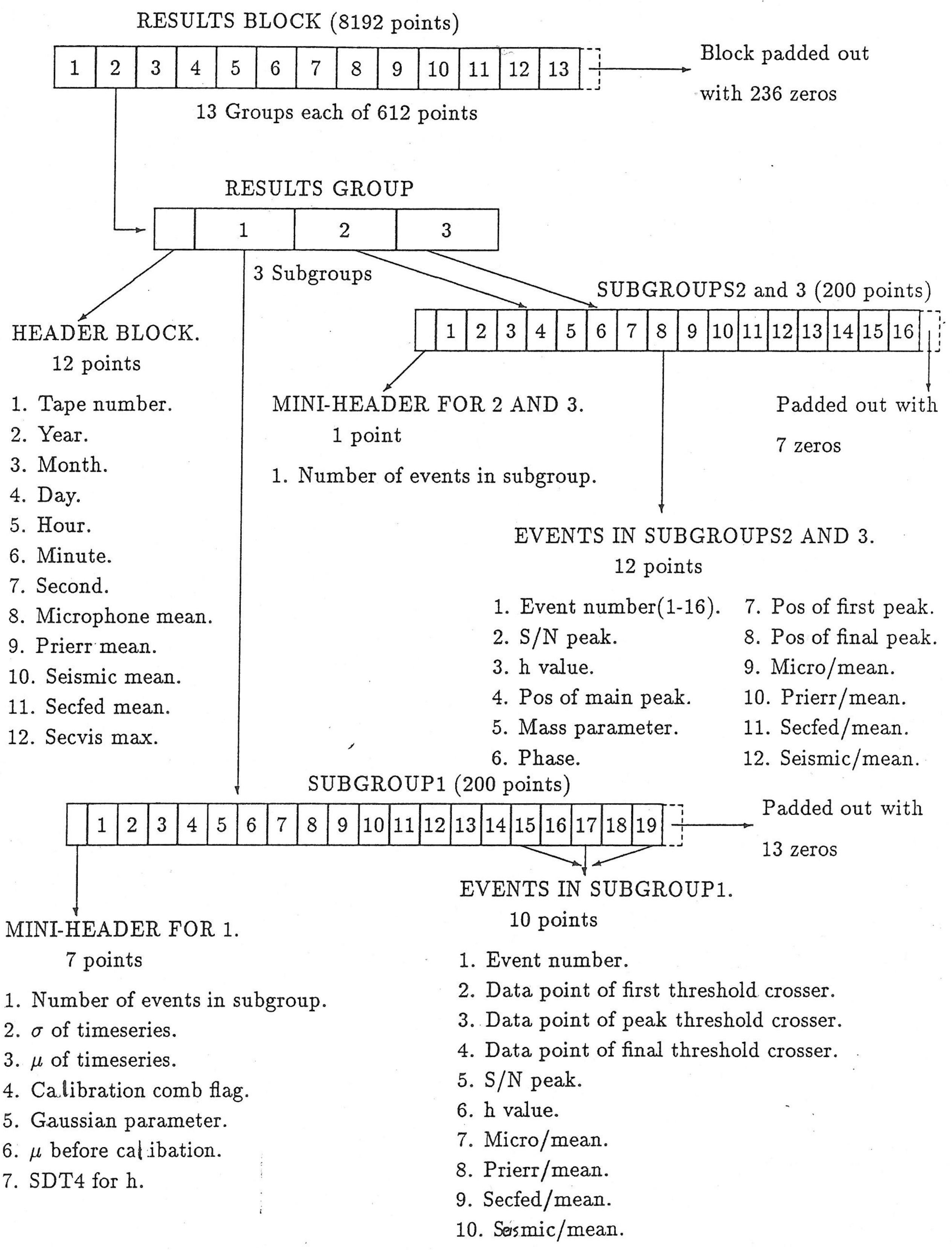


FIGURE A.8

## APPENDIX 2:THE STRUCTURE OF A RESULTS BLOCK



HEADER BLOCK.
12 points

1. Tape number.
2. Year.
3. Month.
4. Day.
5. Hour.
6. Minute.
7. Second.
8. Microphone mean.
9. Prierr mean.
10. Seismic mean.
11. Secfed mean.
12. Secvis max.

MINI-HEADER FOR 2 AND 3.
1 point

1. Number of events in subgroup.

EVENTS IN SUBGROUPS2 AND 3.
12 points

1. Event number(1-16).
2. S/N peak.
3. h value.
4. Pos of main peak.
5. Mass parameter.
6. Phase.
7. Pos of first peak.
8. Pos of final peak.
9. Micro/mean.
10. Prierr/mean.
11. Secfed/mean.
12. Seismic/mean.

MINI-HEADER FOR 1.
7 points

1. Number of events in subgroup.
2. $\sigma$ of timeseries.
3. $\mu$ of timeseries.
4. Calibration comb flag.
5. Gaussian parameter.
6. $\mu$ before calibation.
7. SDT4 for h.

EVENTS IN SUBGROUP1.
10 points

1. Event number.
2. Data point of first threshold crosser.
3. Data point of peak threshold crosser.
4. Data point of final threshold crosser.
5. S/N peak.
6. h value.
7. Micro/mean.
8. Prierr/mean.
9. Secfed/mean.
10. Seismic/mean.

## Appendix 3

This Appendix contains a complete listing of the software described in this thesis. Each task is distinct and the lines are numbered. Also included is a copy of the listing s of those routines taken from Numerical Recipes (Press *et al*(1986)).

```
processor host
processor root
wire ? host[0] root[0]
processor addon
wire ? root[2] addon[1]
processor addon1
wire ? addon[2] addon1[1]
wire ? root[3] addon1[3]
processor addon2
wire ? addon1[2] addon2[1]
processor addon3
wire ? addon2[2] addon3[1]
wire ? addon3[2] root[1]

task afserver           ins=1 outs=1
task try                ins=5 outs=5
task secd1              ins=2 outs=2
task secd2              ins=3 outs=3
task filt4              ins=2 outs=2
task filt5              ins=2 outs=2

place afserver host
place secd1  addon
place secd2 addon1
place filt4 addon2
place filt5 addon3
place try root

connect ? try[1] afserver[0]
connect ? afserver[0] try[1]
connect ? try[2] secd1[0]
connect ? secd1[0] try[2]
connect ? secd1[1] secd2[0]
connect ? secd2[0] secd1[1]
connect ? secd2[1] try[3]
connect ? try[3] secd2[1]
connect ? secd2[2] filt4[0]
connect ? filt4[0] secd2[2]
connect ? filt5[1] try[4]
connect ? try[4] filt5[1]
connect ? filt4[1] filt5[0]
connect ? filt5[0] filt4[1]
```

```
      PROGRAM ROOT
      INCLUDE 'CHAN.INC'
      INTEGER INCHAN,OUTCHAN,IN,OUT,ITIME(6),NBLK,SECERR(6552),
     +MICRO1(3276),MICRO2(3276),MICRO3(3276),MICRO4(3276),MICRO(3276),
     +MICRO5(3276),MICRO6(3276),MICRO7(3276),MICRO8(3276),MICROP(39312),
     +MICRO9(3276),MICRO10(3276),MICRO11(3276),MICRO12(3276),
     +PRIM1(3276),PRIM2(3276),PRIM3(3276),PRIM4(3276),PRIM(3276),
     +PRIM5(3276),PRIM6(3276),PRIM7(3276),PRIM8(3276),PRIERR(39312),
     +PRIM9(3276),PRIM10(3276),PRIM11(3276),PRIM12(3276),
     +SEIS1(546),SEIS2(546),SEIS3(546),SEIS4(546),SEIS(546),
     +SEIS5(546),SEIS6(546),SEIS7(546),SEIS8(546),SEISMO(6552),
     +SEIS9(546),SEIS10(546),SEIS11(546),SEIS12(546),
     +SECV1(546),SECV2(546),SECV3(546),SECV4(546),SECV(546),
     +SECV5(546),SECV6(546),SECV7(546),SECV8(546),SECVIS(6552),
     +SECV9(546),SECV10(546),SECV11(546),SECV12(546),
     +SFED1(3276),SFED2(3276),SFED3(3276),SFED4(3276),SFED(3276),
     +SFED5(3276),SFED6(3276),SFED7(3276),SFED8(3276),SECFED(39312),
     +SFED9(3276),SFED10(3276),SFED11(3276),SFED12(3276),NEMPTY1(2),
     +SEC1,SEC2,SAVE,RFLAG1,HFLAG
      DIMENSION RESULT(200),SECER1(6553),SECER2(6553),CORRES(400),
     +SECER3(6553),SECER4(6553),Z(500),Z5(500),Z6(100),SRES(100),
     +TAPRES(8192)
      CHARACTER INDATA(32768),DIGITS(3276),CHRES(32768),CONVET(39312)
      REAL MEANMC1,MEANMC2,MEANMC3,MEANPM1,MEANPM2,MEANPM3,
     +MEANSF1,MEANSF2,MEANSF3,MEANSE1,MEANSE2,MEANSE3
      LOGICAL GOOD,EOTAPE,FILMRK,SVHEAD,CALIB,MINMRK,CONFID,VETO,RES
C
C       The equivalence functions are used with the housekeeping data
C       to set up the 3 tier structure of the analysis, eg the first
C       third of MICROP(1,13104) is occupied by the first tier, the
C       second third (13105,26208) is occupied by the second tier and
C       the final third (26209,39312) by the third tier.
C       The seismic data was sampled at only 1/6th of the frequency of
C       the other streams hence the numbers are 6 times smaller.
C
      EQUIVALENCE (MICROP(1),MICRO(1)),(MICROP(3277),MICRO1(1)),
     +(MICROP(6553),MICRO2(1)),(MICROP(9829),MICRO3(1)),
     +(MICROP(13105),MICRO4(1)),(MICROP(16381),MICRO5(1)),
     +(MICROP(19657),MICRO6(1)),(MICROP(22933),MICRO7(1)),
     +(MICROP(26209),MICRO8(1)),(MICROP(29485),MICRO9(1)),
     +(MICROP(32761),MICRO10(1)),(MICROP(36037),MICRO11(1)),
     +(PRIERR(1),PRIM(1)),(PRIERR(3277),PRIM1(1)),
     +(PRIERR(6553),PRIM2(1)),(PRIERR(9829),PRIM3(1)),
     +(PRIERR(13105),PRIM4(1)),(PRIERR(16381),PRIM5(1)),
     +(PRIERR(19657),PRIM6(1)),(PRIERR(22933),PRIM7(1)),
     +(PRIERR(26209),PRIM8(1)),(PRIERR(29485),PRIM9(1)),
     +(PRIERR(32761),PRIM10(1)),(PRIERR(36037),PRIM11(1))
      EQUIVALENCE (SEISMO(1),SEIS(1)),(SEISMO(547),SEIS1(1)),
     +(SEISMO(1093),SEIS2(1)),(SEISMO(1639),SEIS3(1)),
     +(SEISMO(2185),SEIS4(1)),(SEISMO(2731),SEIS5(1)),
     +(SEISMO(3277),SEIS6(1)),(SEISMO(3823),SEIS7(1)),
     +(SEISMO(4369),SEIS8(1)),(SEISMO(4915),SEIS9(1)),
     +(SEISMO(5461),SEIS10(1)),(SEISMO(6007),SEIS11(1)),
     +(SECVIS(1),SECV(1)),(SECVIS(547),SECV1(1)),
     +(SECVIS(1093),SECV2(1)),(SECVIS(1639),SECV3(1)),
     +(SECVIS(2185),SECV4(1)),(SECVIS(2731),SECV5(1)),
     +(SECVIS(3277),SECV6(1)),(SECVIS(3823),SECV7(1)),
     +(SECVIS(4369),SECV8(1)),(SECVIS(4915),SECV9(1)),
     +(SECVIS(5461),SECV10(1)),(SECVIS(6007),SECV11(1)),
     +(SECFED(1),SFED(1)),(SECFED(3277),SFED1(1)),
     +(SECFED(6553),SFED2(1)),(SECFED(9829),SFED3(1)),
     +(SECFED(13105),SFED4(1)),(SECFED(16381),SFED5(1)),
     +(SECFED(19657),SFED6(1)),(SECFED(22933),SFED7(1)),
     +(SECFED(26209),SFED8(1)),(SECFED(29485),SFED9(1)),
     +(SECFED(32761),SFED10(1)),(SECFED(36037),SFED11(1)),
     +(TAPRES,CHRES)
C
C       Software connections to 3 slave transputers.
C       Channels to and from defined separately, as given in the
C       .CFG file, they must be integer.
C             PORT(2) connects to task SECD1
C             PORT(3) connects to task SECD2
C             PORT(4) connects to task CORR2.
C
      INCHAN = F77_CHAN_IN_PORT (2)
      OUTCHAN = F77_CHAN_OUT_PORT (2)
      IN = F77_CHAN_IN_PORT (3)
      IN2 = F77_CHAN_IN_PORT (4)
      OUT = F77_CHAN_OUT_PORT (3)
      OUT2 = F77_CHAN_OUT_PORT (4)
      WRITE(6,*) 'CHANNELS DEFINED.'
C
C       Histogram files opened.
C
      OPEN (40,FILE='NUM',STATUS='UNKNOWN')
      OPEN (42,FILE='NUMT',STATUS='UNKNOWN')
      OPEN (43,FILE='NUMC',STATUS='UNKNOWN')
      OPEN (45,FILE='NUMCT',STATUS='UNKNOWN')
C
C       Contact with the CTS drives established.
C       SETUPOK- checks that the SCSI bus is OK.
C
      CALL SETUPOK (GOOD)
      IF(.NOT.GOOD) THEN
        WRITE(6,*) 'STOPPING THE MAIN PROGRAM AFTER SETUP FAILURE'
        WRITE(6,*) 'SCSI BUS FAILURE'
        STOP
      ENDIF
C
C       Confirms that tape is in drive 0. Loads the tape, spaces
C       forward over the initial filemark and set the data transfer
C       size to 32K.
C
      CALL TAPEOK(0,GOOD)
      IF(.NOT.GOOD) THEN
        WRITE(6,*) 'TAPE NOT OK DRIVE 0'
        WRITE(6,*) 'EITHER ....NO TAPE IN DRIVE 0'
        WRITE(6,*) 'OR     ....NO INITIAL FILEMARK'
        WRITE(6,*) 'OR     ....FAILED TO LOAD TAPE'
        STOP
      ENDIF
      I1=0
      GAU2=0.
C
C       If 'write to' tape in drive 1 is new, it will need a file
C       mark at its start. Asked twice to avoid trying to write
C       over an existing filemark and possibly corrupting a tape
C       with data on it.
C
      WRITE(6,*) 'DOES TAPE IN DRIVE 1 REQUIRE A FILE MARK? (Y=1,N=0)'
      READ(5,*) I1
      IF (I1.EQ.1) THEN
        WRITE(6,*) 'ARE YOU SURE DRIVE 1 NEEDS A FILE MARK? (Y=1,N=0)'
        READ(5,*) I1
        IF (I1.EQ.1) THEN
          WRITE(6,*) 'OK'
          WRITE(6,*) 'REWINDING TAPE IN 1.'
          CALL RWOUND (1,GOOD)
          IF(.NOT.GOOD) THEN
            WRITE(6,*) 'REWIND NOT GOOD.'
            STOP
          ENDIF
          WRITE(6,*) 'WRITING FILEMARK.'
          CALL WTFMARK (1,GOOD)
          IF(.NOT.GOOD) THEN
            WRITE(6,*) 'FAILED TO WRITE FILEMARK'
            STOP
          ENDIF
          WRITE(6,*) 'FILEMARK WRITTEN ON 1'
        ENDIF
      ENDIF
C
C       Confirms that the tape in drive 2 is ready to receive
C       results.
C
      CALL TAPEOK(1,GOOD)
      IF(.NOT.GOOD) THEN
        WRITE(6,*) 'TAPE NOT OK DRIVE 1'
        STOP
      ENDIF
      SVHEAD=.FALSE.
      TTTINC=1/13104.
      GOODP=0.075
      IERR=0
C
C       5 HEADER blocks are read from start of tape.
C
      CALL HEADER(0,INDATA,GOOD,EOTAPE,SVHEAD,'NULL',ITIME)
      IF((.NOT.GOOD).AND.(.NOT.EOTAPE)) THEN
        WRITE(6,*) 'PROBLEM READING HEADER BLOCKS'
        STOP
      ELSEIF(EOTAPE) THEN
        WRITE(6,*) 'END OF TAPE FOUND IN MAIN PROGRAM'
        CALL EJECT(0,GOOD)
        IF(GOOD) THEN
          WRITE(6,*) 'TAPE HAS BEEN EJECTED'
          ELSE
          WRITE(6,*) 'EJECT FAILED'
        ENDIF
        STOP
      ENDIF
C
C       If above routine successful, the array ITIME contains
C       date and time information from the start of the tape.
C
      WRITE(6,100) ITIME(3),ITIME(2),ITIME(1)
      WRITE(6,200) ITIME(4),ITIME(5),ITIME(6)
  100 FORMAT(' DATE WAS ',I2,'/',I2,'/',I4)
  200 FORMAT(' TIME WAS ',I2,':',I2,':',I2)
  201 FORMAT(' TIME WAS ',I2,':',I2,':',F13.6)
      DATA TAPRES/8192*0/
      IFILE=1
      ITIM=1
C
C       Requirements for the analysis are entered.
C
      WRITE(6,*) 'TAPE NO?'
      READ(5,*) NTAPE
      WRITE(6,*) 'NO OF LOOPS?'
      READ(5,*) NLOOP
      WRITE(6,*) 'STARTING BLOCK DRIVE 0 ?'
      READ(5,*) NB
      WRITE(6,*) 'DO YOU WANT TO INPUT STARTING BLK FOR D1(Y=1,N=0)?'
      READ(5,*) NYN
C
C       First empty block on results tape is held in file EMPTYD1.
C
      OPEN (544,FILE='EMPTYD1',STATUS='UNKNOWN')
      IF (NYN.EQ.1) THEN
        WRITE(6,*) 'STARTING BLOCK DRIVE 1 ?'
        READ(5,*) NB2
      ELSE
        READ(544,*) NEMPTY1
        NB2=NEMPTY1(1)
        NEMPTY1(2)=NEMPTY1(1)
        WRITE(6,*) 'FIRST EMPTY BLOCK  ',NB2
      ENDIF
      NB2=NB2-1
C
C       Option to save histogram arrays periodically throughout
C       the run.
C
      WRITE(6,*) 'SAVE HISTOGRAM DATA DURING RUN?(Y=1,N=0)'
      READ(5,*) SAVE
      IF (SAVE.EQ.0) THEN
        SAVE=20000
        LS=1
        GO TO 881
      ENDIF
      WRITE(6,*) 'SAVE EVERY?'
      READ(5,*) SAVE
      WRITE(6,*) 'START SAVE ?'
      READ(5,*) LS
  881 NB=NB-1
C
C     The time of the first true data block on the tape is found.
C     It is calculated backwards from the first minute mark found
C     in the data using the fact that one block represents
C     0.3276 seconds and also that the sampling frequency for the
C     digital byte from which the minute marker is extracted is
C     10000Hz hence each element represents 0.0001 seconds.
C
C     Assuming that the first data occurs within 10 seconds of the
C     header block, NB is the minimum number of blocks that can be
C     spaced over before there any possibility of finding the first
C     minute mark.
C
      TTINC=1/3276.
      WRITE(6,*) 'DO YOU WANT TO USE THE SPACE ROUTINE (Y=1,N=0)'
      READ(5,*) IYN
      IF (IYN.EQ.0) THEN
        GO TO 373
      ENDIF
      IF (ITIME(6).GT.52) THEN
        NB1=0
      ELSE
        NB1=INT((52-ITIME(6))*10000*TTINC)
      ENDIF
      CALL SPACE(0,NB1,NUMOUT,RES,FILMRK)
      IF(FILMRK) THEN
        WRITE(6,*) 'FILEMARK FOUND IN SPACE ROUTINE.'
      ENDIF
      IF(RES) THEN
        WRITE(6,*) 'SPACE ROUTINE SUCCESSFUL'
      ELSE
        WRITE(6,*) 'SPACE ROUTINE FAILED'
        STOP
      ENDIF
C
C       Blocks of actual data are read into the array INDATA in
C       an attempt to find the first minute mark.
C
  373 CALL RDTAPE(0,INDATA,NBLK,GOOD,FILMRK)
      IF(FILMRK) THEN
        WRITE(6,*) 'FILEMARK FOUND.'
      ENDIF
      IF(GOOD) THEN
        WRITE(6,*) NBLK
      ELSE
        WRITE(6,*) 'NOT GOOD (DRIVE 0)'
        STOP
      ENDIF
C
C       GTDATA extracts the digital byte from INDATA and reads it into
C       the array DIGITS.
C
      CALL GTDIGT(INDATA,DIGITS)
C
C       Each element of DIGITS is checked for the minute mark.
C
      DO 371 J8=1,3276
      IF (MINMRK(DIGITS(J8))) THEN
        WRITE(6,*) NBLK,J8
C
C       When found the time of the first block is calculated and
C       the loop governed by GO TO 373 is left.
C
        F1TIME=60-NBLK*.3276-J8*.0001
        ITIME(6)=INT(F1TIME)
        F1BLK=0
        F2BLK=NB
        WRITE(6,*) 'TIME AT FIRST BLOCK.'
        WRITE(6,201) ITIME(4),ITIME(5),F1TIME
        GO TO 372
      ENDIF
  371 CONTINUE
      GO TO 373
  372 RFLAG1=0
      KINC=0
C
C       If the required starting place(NB) in the data tape is not at
C       its start then the number of minute markers passed in getting
C       to NB is taken into account in up dating the time.
C
      IF (NB.GT.NBLK) THEN
        KINC=INT((NB-NBLK)*0.3276/60.+1)
        F2BLK=F2BLK+KINC*183.
        WRITE(6,*) 'KINC=',KINC,'     F2BLK=',F2BLK
      ENDIF
      NBC=NB-NBLK
      WRITE(6,*) NBC
      IF (NBC.GT.1000) THEN
        INBC=INT(NBC/1000.)
        NBC=NBC-INBC*1000
        DO IJ=1,INBC
        CALL SPACE(0,1000,NUMOUT,RES,FILMRK)
        IF (FILMRK) THEN
          WRITE(6,*) 'FILEMARK IN SPACE 0'
          STOP
        ENDIF
        IF (.NOT.RES) THEN
          WRITE(6,*) 'SPACE IN 0 FAILED'
          STOP
        ENDIF
        WRITE(6,*) IJ*1000
        ENDDO
      ENDIF
      CALL SPACE(0,NBC,NUMOUT,RES,FILMRK)
      IF (FILMRK) THEN
        WRITE(6,*) 'FILEMARK IN SPACE 0'
        STOP
      ENDIF
      IF (.NOT.RES) THEN
        WRITE(6,*) 'SPACE IN 0 FAILED'
        STOP
      ENDIF
C
C       Results tape is moved to appropriate place.
C
      CALL SPACE(1,NB2,NUMOUT,RES,FILMRK)
      IF (FILMRK) THEN
        WRITE(6,*) 'FILEMARK IN SPACE 1'
        STOP
      ENDIF
      IF (.NOT.RES) THEN
        WRITE(6,*) 'SPACE IN 1 FAILED'
        STOP
      ENDIF
      NUM=0.
      SECER1(6553)=0.
C
C       Main analysis begins. First block read from tape.
C
      CALL RDTAPE(0,INDATA,NBLK,GOOD,FILMRK)
      IF(FILMRK) THEN
        WRITE(6,*) 'FILEMARK FOUND.'
      ENDIF
      IF(GOOD) THEN
        WRITE(6,*) NBLK
      ELSE
        WRITE(6,*) 'TAPE NOT GOOD DRIVE 0.'
        STOP
      ENDIF
      HFLAG=1
C
C       Secondary error point data extracted along with the various
C       streams of housekeeping data.
C
      CALL GTSERR(INDATA,SECERR)
      CALL GTMICR(INDATA,MICRO)
      CALL GTPERR(INDATA,PRIM)
      CALL GTSEIS(INDATA,SEIS)
      CALL GTSVIS(INDATA,SECV)
      CALL GTSFED(INDATA,SFED)
      CALL GTDIGT(INDATA,DIGITS)
```

```
C
C       The presence of a calibration comb in the data is tested.
C
      DO 161 I9=1,3276
      CONVET(I9)=DIGITS(I9)
      IF (CALIB(DIGITS(I9))) THEN
        SECER1(6553)=1.
      ENDIF
      SECER1(I9)=SECERR(I9)
      I99=I9+3276
  161 SECER1(I99)=SECERR(I99)
C
C       First block of secondary error point data sent to SECD1.
C       The 6553'rd point is set to 1 if a calibration comb is
C       applied in this stretch of data.
C
      CALL F77_CHAN_OUT_MESSAGE (26212,SECER1,OUTCHAN)
      L=-1
C
C       Computer time at start of main looping structure is found.
C
      CALL ICLOCK(SEC1)
C
C       Main loop begins.
C       If histogram statistics are required SECER1(6553) is set to 5.
C
  350 IF (LS.EQ.SAVE) THEN
        SECER1(6553)=5
        GO TO 351
      ENDIF
      SECER1(6553)=0
  351 SECER2(6553)=0
      SECER3(6553)=0
      SECER4(6553)=0
C
C       First of the four data block read in each loop.
C       Provides second block in each group.
C
      CALL RDTAPE(0,INDATA,NBLK,GOOD,FILMRK)
      IF (FILMRK) THEN
        WRITE(6,*) 'FILEMARK FOUND.'
      ENDIF
      IF (GOOD) THEN
        WRITE(6,*) NBLK
      ELSE
        WRITE(6,*) 'READ TAPE FAILED, DRIVE 0.'
        STOP
      ENDIF
      CALL GTSERR(INDATA,SECERR)
C
C       The value of HFLAG decides on to which of the 3 tier's the
C       housekeeping data is read. *1 is the first, *5 the second
C       and *9 the third. At this stage the second blocks worth of
C       each of the tiers is occupied.
C
      IF (HFLAG.EQ.1) THEN
        CALL GTMICR(INDATA,MICRO1)
        CALL GTPERR(INDATA,PRIM1)
        CALL GTSVIS(INDATA,SECV1)
        CALL GTSEIS(INDATA,SEIS1)
        CALL GTSFED(INDATA,SFED1)
        GO TO 1101
      ENDIF
      IF (HFLAG.EQ.2) THEN
        CALL GTMICR(INDATA,MICRO5)
        CALL GTPERR(INDATA,PRIM5)
        CALL GTSEIS(INDATA,SEIS5)
        CALL GTSVIS(INDATA,SECV5)
        CALL GTSFED(INDATA,SFED5)
        GO TO 1101
      ENDIF
      CALL GTMICR(INDATA,MICRO9)
      CALL GTPERR(INDATA,PRIM9)
      CALL GTSEIS(INDATA,SEIS9)
      CALL GTSVIS(INDATA,SECV9)
      CALL GTSFED(INDATA,SFED9)
 1101 CALL GTDIGT(INDATA,DIGITS)
C
C       Minute marks are only searched for in specific regions.
C       F2BLK is the block in which the search for the next
C       minute mark starts. When found the time is updated.
C
      IF (NBLK.GE.F2BLK) THEN
        DO 771 J8=1,3276
        IF (MINMRK(DIGITS(J8))) THEN
          IF (KINC.GT.0) THEN
            ITIME(5)=ITIME(5)+KINC
            KINC=0
          ENDIF
          F1TIME=0
          ITIME(5)=ITIME(5)+1
          IF (ITIME(5).GE.60) THEN
            IDIFF=INT(ITIME(5)/60.)
            ITIME(5)=ITIME(5)-60*IDIFF
            ITIME(4)=ITIME(4)+1*IDIFF
          ENDIF
          IF (ITIME(4).GE.24) THEN
            ITIME(3)=ITIME(3)+1
            ITIME(4)=ITIME(4)-24
          ENDIF
          F1BLK=NBLK+J8*TTINC
          F2BLK=NBLK+183
          WRITE(6,*) 'TIME AT BLOCK.',NBLK
          WRITE(6,201) ITIME(4),ITIME(5),F1TIME
          GO TO 772
          ENDIF
  771     CONTINUE
      ENDIF
      IHF=(HFLAG-1)*13104+3276
  772 DO 171 I9=1,3276
C
C       DIGITS from all 3 tiers are held in one continuously updated
C       array, CONVET.
C
      CONVET(I9+IHF)=DIGITS(I9)
      II9=I9+3276
      SECER1(II9)=SECERR(II9)
  171 SECER1(I9)=SECERR(I9)
      IF (SECER1(6553).NE.0) THEN
        GO TO 1722
      ENDIF
C
C       Comb search.
C
      DO 172 I9=1,3276
      IF (CALIB(DIGITS(I9))) THEN
        SECER1(6553)=1.
        GO TO 1722
      ENDIF
  172 CONTINUE
C
C       Second of the four data block read in each loop.
C       Provides third block in each group.
C
 1722 CALL RDTAPE(0,INDATA,NBLK,GOOD,FILMRK)
      IF(FILMRK) THEN
        WRITE(6,*) 'FILEMARK FOUND.'
      ENDIF
      IF (GOOD) THEN
        WRITE(6,*) NBLK
      ELSE
        WRITE(6,*) 'READ TAPE FAILED, DRIVE 0.'
        STOP
      ENDIF
      CALL GTSERR(INDATA,SECERR)
C
C       At this stage the third blocks worth of each of the tiers
C       is occupied.
C
      IF (HFLAG.EQ.1) THEN
        CALL GTMICR(INDATA,MICRO2)
        CALL GTPERR(INDATA,PRIM2)
        CALL GTSVIS(INDATA,SECV2)
        CALL GTSEIS(INDATA,SEIS2)
        CALL GTSFED(INDATA,SFED2)
        GO TO 1102
      ENDIF
      IF (HFLAG.EQ.2) THEN
        CALL GTMICR(INDATA,MICRO6)
        CALL GTPERR(INDATA,PRIM6)
        CALL GTSVIS(INDATA,SECV6)
        CALL GTSEIS(INDATA,SEIS6)
        CALL GTSFED(INDATA,SFED6)
        GO TO 1102
      ENDIF
      CALL GTMICR(INDATA,MICRO10)
      CALL GTPERR(INDATA,PRIM10)
      CALL GTSEIS(INDATA,SEIS10)
      CALL GTSVIS(INDATA,SECV10)
      CALL GTSFED(INDATA,SFED10)
 1102 CALL GTDIGT(INDATA,DIGITS)
      IF (NBLK.GE.F2BLK) THEN
        DO 773 J8=1,3276
        IF (MINMRK(DIGITS(J8))) THEN
          IF (KINC.GT.0) THEN
            ITIME(5)=ITIME(5)+KINC
            KINC=0
          ENDIF
          F1TIME=0
          ITIME(5)=ITIME(5)+1
          IF (ITIME(5).GE.60) THEN
            IDIFF=INT(ITIME(5)/60.)
            ITIME(5)=ITIME(5)-60*IDIFF
            ITIME(4)=ITIME(4)+1*IDIFF
          ENDIF
          IF (ITIME(4).GE.24) THEN
            ITIME(3)=ITIME(3)+1
            ITIME(4)=ITIME(4)-24
            ENDIF
            F1BLK=NBLK+J8*TTINC
            F2BLK=NBLK+183
            WRITE(6,*) 'TIME AT BLOCK.',NBLK
            WRITE(6,201) ITIME(4),ITIME(5),F1TIME
            GO TO 774
          ENDIF
  773     CONTINUE
        ENDIF
        IHF=(HFLAG-1)*13104+2*3276
  774   DO 181 I9=1,3276
        CONVET(I9+IHF)=DIGITS(I9)
        II9=I9+3276
        SECER2(II9)=SECERR(II9)
  181   SECER2(I9)=SECERR(I9)
        IF (SECER2(6553).NE.0) THEN
          GO TO 1822
        ENDIF
        DO 182 I9=1,3276
        IF (CALIB(DIGITS(I9))) THEN
          SECER2(6553)=1.
          GO TO 1822
        ENDIF
  182   CONTINUE
C
C         Third of the four data block read in each loop.
C         Provides forth block in each group.
C
 1822   CALL RDTAPE(0,INDATA,NBLK,GOOD,FILMRK)
        IF(FILMRK) THEN
          WRITE(6,*) 'FILEMARK FOUND.'
        ENDIF
        IF (GOOD) THEN
          WRITE(6,*) NBLK
        ELSE
          WRITE(6,*) 'READ TAPE FAILED, DRIVE 0.'
          STOP
        ENDIF
        CALL GTSERR(INDATA,SECERR)
C
C         At this stage the forth blocks worth of each of the tiers
C         is occupied.
C
        IF (HFLAG.EQ.1) THEN
          CALL GTMICR(INDATA,MICRO3)
          CALL GTPERR(INDATA,PRIM3)
          CALL GTSVIS(INDATA,SECV3)
          CALL GTSEIS(INDATA,SEIS3)
          CALL GTSFED(INDATA,SFED3)
          GO TO 1103
        ENDIF
        IF (HFLAG.EQ.2) THEN
          CALL GTMICR(INDATA,MICRO7)
          CALL GTPERR(INDATA,PRIM7)
          CALL GTSVIS(INDATA,SECV7)
          CALL GTSEIS(INDATA,SEIS7)
          CALL GTSFED(INDATA,SFED7)
          GO TO 1103
        ENDIF
        CALL GTMICR(INDATA,MICRO11)
        CALL GTPERR(INDATA,PRIM11)
        CALL GTSVIS(INDATA,SECV11)
        CALL GTSEIS(INDATA,SEIS11)
        CALL GTSFED(INDATA,SFED11)
 1103   CALL GTDIGT(INDATA,DIGITS)
        IF (NBLK.GE.F2BLK) THEN
          DO 775 J8=1,3276
          IF (MINMRK(DIGITS(J8))) THEN
            IF (KINC.GT.0) THEN
              ITIME(5)=ITIME(5)+KINC
              KINC=0
            ENDIF
            F1TIME=0
            ITIME(5)=ITIME(5)+1
            IF (ITIME(5).GE.60) THEN
              IDIFF=INT(ITIME(5)/60.)
              ITIME(5)=ITIME(5)-60*IDIFF
              ITIME(4)=ITIME(4)+1*IDIFF
            ENDIF
            IF (ITIME(4).GE.24) THEN
              ITIME(3)=ITIME(3)+1
              ITIME(4)=ITIME(4)-24
            ENDIF
            F1BLK=NBLK+J8*TTINC
            F2BLK=NBLK+183
            WRITE(6,*) 'TIME AT BLOCK.',NBLK
            WRITE(6,201) ITIME(4),ITIME(5),F1TIME
            GO TO 776
          ENDIF
  775     CONTINUE
        ENDIF
        IHF=(HFLAG-1)*13104+3*3276
  776   DO 191 I9=1,3276
        CONVET(I9+IHF)=DIGITS(I9)
```

```
      II9=I9+3276
      SECER3(II9)=SECERR(II9)
191   SECER3(I9)=SECERR(I9)
      IF (SECER3(6553).NE.0) THEN
        GO TO 1922
      ENDIF
      DO 192 I9=1,3276
      IF (CALIB(DIGITS(I9))) THEN
        SECER3(6553)=1.
        GO TO 1922
      ENDIF
192   CONTINUE
1922  CALL RDTAPE(0,INDATA,NBLK,GOOD,FILMRK)
      IF(FILMRK) THEN
        WRITE(6,*) 'FILEMARK FOUND.'
      ENDIF
      IF (GOOD) THEN
        WRITE(6,*) NBLK
      ELSE
        WRITE(6,*) 'READ TAPE FAILED, DRIVE 0.'
        STOP
      ENDIF
      CALL GTSERR(INDATA,SECERR)
C
C       At this stage the fifth blocks worth of each of the tiers
C       is occupied and the statistics (standard deviations and means)
C       for the housekeeping in the current tier are worked out.
C
      IF (HFLAG.EQ.1) THEN
        CALL GTMICR(INDATA,MICRO4)
        CALL GTPERR(INDATA,PRIM4)
        CALL GTSVIS(INDATA,SECV4)
        CALL GTSEIS(INDATA,SEIS4)
        CALL GTSFED(INDATA,SFED4)
        HFLAG=2
        BSECV1=SECVIS(1)
        SDMC1=0.
        MEANMC1=0
        SDPM1=0.
        MEANPM1=0
        SDSE1=0.
        MEANSE1=0
        SDSF1=0.
        MEANSF1=0
        DO 1105 I=1,13104
        IF (I.LT.2185) THEN
          MEANSE1=MEANSE1+SEISMO(I)
          SDSE1=SDSE1+SEISMO(I)*SEISMO(I)
          IF (SECVIS(I).GT.BSECV1) THEN
            BSECV1=SECVIS(I)
          ENDIF
        ENDIF
        MEANSF1=MEANSF1+SECFED(I)
        SDSF1=SDSF1+SECFED(I)*SECFED(I)
        MEANPM1=MEANPM1+PRIERR(I)
        SDPM1=SDPM1+PRIERR(I)*PRIERR(I)
        MEANMC1=MEANMC1+MICROP(I)
1105    SDMC1=SDMC1+MICROP(I)*MICROP(I)
C
C         TTTINC=13104 which the number of housekeeping elements in
C         each 4 block tier, except for the seismic stream which has
C         only 2184 elements.
C
        MEANMC1=MEANMC1*TTTINC
        SDMC1=SQRT(SDMC1*TTTINC-MEANMC1**2)
        MEANPM1=MEANPM1*TTTINC
        SDPM1=SQRT(SDPM1*TTTINC-MEANPM1**2)
        MEANSE1=MEANSE1/2184.
        SDSE1=SQRT(SDSE1/2184.-MEANSE1**2)
        MEANSF1=MEANSF1*TTTINC
        SDSF1=SQRT(SDSF1*TTTINC-MEANSF1**2)
        IHF=4*3276
        GO TO 1104
      ENDIF
      IF (HFLAG.EQ.2) THEN
        CALL GTMICR(INDATA,MICRO8)
        CALL GTPERR(INDATA,PRIM8)
        CALL GTSVIS(INDATA,SECV8)
        CALL GTSEIS(INDATA,SEIS8)
        CALL GTSFED(INDATA,SFED8)
        HFLAG=3
        BSECV2=SECVIS(2185)
        SDMC2=0.
        MEANMC2=0
        SDPM2=0.
        MEANPM2=0
        SDSE2=0.
        MEANSE2=0
        SDSF2=0.
        MEANSF2=0
        DO 1106 I=13105,26208
        IF (I.LT.15289) THEN
          I1=I-10920
          MEANSE2=MEANSE2+SEISMO(I1)
```

```
                SDSE2=SDSE2+SEISMO(I1)*SEISMO(I1)
                IF (SECVIS(I1).GT.BSECV2) THEN
                  BSECV2=SECVIS(I1)
                ENDIF
              ENDIF
              MEANPM2=MEANPM2+PRIERR(I)
              SDPM2=SDPM2+PRIERR(I)*PRIERR(I)
              MEANSF2=MEANSF2+SECFED(I)
              SDSF2=SDSF2+SECFED(I)*SECFED(I)
              MEANMC2=MEANMC2+MICROP(I)
 1106         SDMC2=SDMC2+MICROP(I)*MICROP(I)
              MEANMC2=MEANMC2*TTTINC
              SDMC2=(SDMC2*TTTINC-MEANMC2**2)**.5
              MEANPM2=MEANPM2*TTTINC
              SDPM2=(SDPM2*TTTINC-MEANPM2**2)**.5
              MEANSE2=MEANSE2/2184.
              SDSE2=(SDSE2/2184.-MEANSE2**2)**.5
              MEANSF2=MEANSF2*TTTINC
              SDSF2=(SDSF2*TTTINC-MEANSF2**2)**.5
              IHF=13104+4*3276
              GO TO 1104
            ENDIF
            CALL GTMICR(INDATA,MICRO12)
            CALL GTPERR(INDATA,PRIM12)
            CALL GTSVIS(INDATA,SECV12)
            CALL GTSEIS(INDATA,SEIS12)
            CALL GTSFED(INDATA,SFED12)
            HFLAG=1
            BSECV3=SECVIS(4369)
            SDMC3=0.
            MEANMC3=0
            SDPM3=0.
            MEANPM3=0
            SDSE3=0.
            MEANSE3=0
            SDSF3=0.
            MEANSF3=0
            DO 1107 I=26209,39312
            IF (I.LT.28393) THEN
              I1=I-21840
              MEANSE3=MEANSE3+SEISMO(I1)
              SDSE3=SDSE3+SEISMO(I1)*SEISMO(I1)
              IF (SECVIS(I1).GT.BSECV3) THEN
                BSECV3=SECVIS(I1)
              ENDIF
            ENDIF
            MEANPM3=MEANPM3+PRIERR(I)
```

```
            SDPM3=SDPM3+PRIERR(I)*PRIERR(I)
            MEANSF3=MEANSF3+SECFED(I)
            SDSF3=SDSF3+SECFED(I)*SECFED(I)
            MEANMC3=MEANMC3+MICROP(I)
 1107       SDMC3=SDMC3+MICROP(I)*MICROP(I)
            MEANMC3=MEANMC3*TTTINC
            SDMC3=(SDMC3*TTTINC-MEANMC3**2)**.5
            MEANPM3=MEANPM3*TTTINC
            SDPM3=(SDPM3*TTTINC-MEANPM3**2)**.5
            MEANSE3=MEANSE3/2184.
            SDSE3=(SDSE3/2184.-MEANSE3**2)**.5
            MEANSF3=MEANSF3*TTTINC
            SDSF3=(SDSF3*TTTINC-MEANSF3**2)**.5
            IHF=0
 C
 C            GTDIGT is tier independent.
 C
 1104       CALL GTDIGT(INDATA,DIGITS)
            IF (NBLK.GE.F2BLK) THEN
              DO 777 J8=1,3276
              IF (MINMRK(DIGITS(J8))) THEN
                IF (KINC.GT.0) THEN
                  ITIME(5)=ITIME(5)+KINC
                  KINC=0
                ENDIF
                F1TIME=0
                ITIME(5)=ITIME(5)+1
                IF (ITIME(5).GE.60) THEN
                  IDIFF=INT(ITIME(5)/60.)
                  ITIME(5)=ITIME(5)-60*IDIFF
                  ITIME(4)=ITIME(4)+1*IDIFF
                ENDIF
                IF (ITIME(4).GE.24) THEN
                  ITIME(3)=ITIME(3)+1
                  ITIME(4)=ITIME(4)-24
                ENDIF
                F1BLK=NBLK+J8*TTINC
                F2BLK=NBLK+183
                WRITE(6,*) 'TIME AT BLOCK.',NBLK
                WRITE(6,201) ITIME(4),ITIME(5),F1TIME
                GO TO 778
              ENDIF
 777          CONTINUE
            ENDIF
 778        DO 231 I9=1,3276
            CONVET(I9+IHF)=DIGITS(I9)
            II9=I9+3276
```

```
      SECER4(II9)=SECERR(II9)
231   SECER4(I9)=SECERR(I9)
      IF (SECER4(6553).NE.0) THEN
        GO TO 2322
      ENDIF
      DO 232 I9=1,3276
      IF (CALIB(DIGITS(I9))) THEN
        SECER4(6553)=1.
        GO TO 2322
      ENDIF
232   CONTINUE
2322  IF (L.LT.1) THEN
        L=L+1
        GO TO 251
      ENDIF
C
C       Results are returned from task SECD2.
C
      CALL F77_CHAN_IN_MESSAGE (800,RESULT,IN)
C
C       NUM is the accumulative total number of events found in SECD2.
C
      NUM=NUM+RESULT(1)
      WRITE(6,*) 'RESULTS BACK.',L,NLOOP,LS,NUM
C     WRITE(50,*) RESULT(6)
C     WRITE(51,*) RESULT(4)
      IF (ABS(RESULT(5)-1).LT.GOODP) THEN
        GAU2=GAU2+1
      ENDIF
C     WRITE(52,*) RESULT(5)
C     WRITE(53,*) RESULT(2)
      LS=LS+1
      L=L+1
C
C       Results are returned from task CORR2.
C
      CALL F77_CHAN_IN_MESSAGE (1600,CORRES,IN2)
      WRITE(6,*) 'CORR RESULTS BACK.'
C
C       Histogram statistics returned at appropriate time from
C       tasks SECD2 and CORR2.
C
      IF (LS.EQ.SAVE+3) THEN
        CALL F77_CHAN_IN_MESSAGE (2000,Z5,IN)
        WRITE(42,*) Z5
        CALL F77_CHAN_IN_MESSAGE (400,Z6,IN2)
        WRITE(45,*) Z6
        WRITE(6,*) 'STATS BACK.'
        LS=0
      ENDIF
C
C       The 612 points associated with each group are built
C       up from the results returned from the other tasks.
C       The particular housekeeping data associated with each event is
C       added along with a 12 element header which includes the tape
C       number, time and date information and the means of the
C       housekeeping data taken during this group.
C
      IF (RFLAG1.LE.12) THEN
C
C       Header is built up.
C
        IFL=612*RFLAG1
        TAPRES(IFL+1)=NTAPE
        TAPRES(IFL+2)=ITIME(1)
        TAPRES(IFL+3)=ITIME(2)
        TAPRES(IFL+4)=ITIME(3)
        TAPRES(IFL+5)=ITIME(4)
        F6BLK=(NBLK-13-F1BLK)
        IF (F6BLK.LT.0) THEN
          TAPRES(IFL+6)=ITIME(5)-1
          TAPRES(IFL+7)=60+F6BLK*.3276+F1TIME
        ELSE
          TAPRES(IFL+6)=ITIME(5)
          TAPRES(IFL+7)=F6BLK*.3276+F1TIME
        ENDIF
        IF (TAPRES(IFL+7).GE.60) THEN
          TAPRES(IFL+6)=TAPRES(IFL+6)+INT(TAPRES(IFL+7)/60.)
          TAPRES(IFL+7)=TAPRES(IFL+7)-INT(TAPRES(IFL+7)/60.)*60.
        ENDIF
        IF (TAPRES(IFL+6).GE.60) THEN
          TAPRES(IFL+6)=TAPRES(IFL+6)-60.
          TAPRES(IFL+5)=TAPRES(IFL+5)+1.
        ENDIF
        IF (TAPRES(IFL+5).GE.24) THEN
          TAPRES(IFL+5)=TAPRES(IFL+5)-24
          TAPRES(IFL+4)=TAPRES(IFL+4)+1.
        ENDIF
        RFLAG1=RFLAG1+1
      ENDIF
      WRITE(6,*) 'HFLAG= ',HFLAG
C
C       Depending on the position in the tier, the appropriate
C       housekeeping means are added to the header.
```

```
C
      IF (HFLAG.EQ.1) THEN
        TAPRES(IFL+8)=SDMC1
        TAPRES(IFL+9)=SDPM1
        TAPRES(IFL+10)=SDSE1
        TAPRES(IFL+11)=SDSF1
        TAPRES(IFL+12)=BSECV1
        IFILE=IFILE+1
        IHOS=14
C
C         The results from SECD2 are dealt with. RESULT(1) is the
C         number of events.
C
        DO J20=1,INT(RESULT(1))
        IHOS1=INT((RESULT(IHOS-4)+1)/2)
        RESULT(IHOS)=ABS(MICROP(IHOS1)/SDMC1)
        RESULT(IHOS+1)=ABS(PRIERR(IHOS1)/SDPM1)
        RESULT(IHOS+2)=ABS(SECFED(IHOS1)/SDSF1)
        IHOS2=INT(IHOS1/6.)
        RESULT(IHOS+3)=ABS(SEISMO(IHOS2)/SDSE1)
        IHOS=IHOS+10
        ENDDO
        IHOS=10
C
C         The results from CORR1 and CORR2 are dealt with.
C
        DO J21=1,INT(CORRES(1))
        IHOS1=INT((CORRES(IHOS-5)+1)/2)
        CORRES(IHOS)=ABS(MICROP(IHOS1)/SDMC1)
        CORRES(IHOS+1)=ABS(PRIERR(IHOS1)/SDPM1)
        CORRES(IHOS+2)=ABS(SECFED(IHOS1)/SDSF1)
        IHOS2=INT(IHOS1/6.)
        CORRES(IHOS+3)=ABS(SEISMO(IHOS2)/SDSE1)
        IHOS=IHOS+12
        ENDDO
        IHOS=210
        DO J22=1,INT(CORRES(201))
        IHOS1=INT((CORRES(IHOS-5)+1)/2)
        CORRES(IHOS)=ABS(MICROP(IHOS1)/SDMC1)
        CORRES(IHOS+1)=ABS(PRIERR(IHOS1)/SDPM1)
        CORRES(IHOS+2)=ABS(SECFED(IHOS1)/SDSF1)
        IHOS2=INT(IHOS1/6.)
        CORRES(IHOS+3)=ABS(SEISMO(IHOS2)/SDSE1)
        IHOS=IHOS+12
        ENDDO
C
C         The housekeeping in the last block in tier 3 becomes the
C         first block of tier 1.
C
        DO 1109 I=1,3276
        IF (I.LE.546) THEN
          SECVIS(I)=SECV12(I)
          SEISMO(I)=SEIS12(I)
        ENDIF
        PRIERR(I)=PRIM12(I)
        SECFED(I)=SFED12(I)
 1109   MICROP(I)=MICRO12(I)
      ENDIF
      IF (HFLAG.EQ.2) THEN
        TAPRES(IFL+8)=SDMC2
        TAPRES(IFL+9)=SDPM2
        TAPRES(IFL+10)=SDSE2
        TAPRES(IFL+11)=SDSF2
        TAPRES(IFL+12)=BSECV2
        IFILE=IFILE+1
        IHOS=14
        DO J21=1,INT(RESULT(1))
        IHOS1=INT((RESULT(IHOS-4)+1)/2)+13104
        RESULT(IHOS)=ABS(MICROP(IHOS1)/SDMC2)
        RESULT(IHOS+1)=ABS(PRIERR(IHOS1)/SDPM2)
        RESULT(IHOS+2)=ABS(SECFED(IHOS1)/SDSF2)
        IHOS2=INT(IHOS1/6.)
        RESULT(IHOS+3)=ABS(SEISMO(IHOS2)/SDSE2)
        IHOS=IHOS+10
        ENDDO
        IHOS=10
        DO J21=1,INT(CORRES(1))
        IHOS1=INT((CORRES(IHOS-5)+1)/2)
        CORRES(IHOS)=ABS(MICROP(IHOS1)/SDMC2)
        CORRES(IHOS+1)=ABS(PRIERR(IHOS1)/SDPM2)
        CORRES(IHOS+2)=ABS(SECFED(IHOS1)/SDSF2)
        IHOS2=INT(IHOS1/6.)
        CORRES(IHOS+3)=ABS(SEISMO(IHOS2)/SDSE2)
        IHOS=IHOS+12
        ENDDO
        IHOS=210
        DO J22=1,INT(CORRES(201))
        IHOS1=INT((CORRES(IHOS-5)+1)/2)
        CORRES(IHOS)=ABS(MICROP(IHOS1)/SDMC2)
        CORRES(IHOS+1)=ABS(PRIERR(IHOS1)/SDPM2)
        CORRES(IHOS+2)=ABS(SECFED(IHOS1)/SDSF2)
        IHOS2=INT(IHOS1/6.)
        CORRES(IHOS+3)=ABS(SEISMO(IHOS2)/SDSE2)
        IHOS=IHOS+12
```

```
            ENDDO
          ENDIF
          IF (HFLAG.EQ.3) THEN
            TAPRES(IFL+8)=SDMC3
            TAPRES(IFL+9)=SDPM3
            TAPRES(IFL+10)=SDSE3
            TAPRES(IFL+11)=SDSF3
            TAPRES(IFL+12)=BSECV1
            IFILE=IFILE+1
            IHOS=14
            DO J22=1,INT(RESULT(1))
            IHOS1=INT((RESULT(IHOS-4)+1)/2)+26208
            RESULT(IHOS)=ABS(MICROP(IHOS1)/SDMC3)
            RESULT(IHOS+1)=ABS(PRIERR(IHOS1)/SDPM3)
            RESULT(IHOS+2)=ABS(SECFED(IHOS1)/SDSF3)
            IHOS2=INT(IHOS1/6.)
            RESULT(IHOS+3)=ABS(SEISMO(IHOS2)/SDSE3)
            IHOS=IHOS+10
            ENDDO
            IHOS=10
            DO J21=1,INT(CORRES(1))
            IHOS1=INT((CORRES(IHOS-5)+1)/2)
            CORRES(IHOS)=ABS(MICROP(IHOS1)/SDMC3)
            CORRES(IHOS+1)=ABS(PRIERR(IHOS1)/SDPM3)
            CORRES(IHOS+2)=ABS(SECFED(IHOS1)/SDSF3)
            IHOS2=INT(IHOS1/6.)
            CORRES(IHOS+3)=ABS(SEISMO(IHOS2)/SDSE3)
            IHOS=IHOS+12
            ENDDO
            IHOS=210
            DO J22=1,INT(CORRES(201))
            IHOS1=INT((CORRES(IHOS-5)+1)/2)
            CORRES(IHOS)=ABS(MICROP(IHOS1)/SDMC3)
            CORRES(IHOS+1)=ABS(PRIERR(IHOS1)/SDPM3)
            CORRES(IHOS+2)=ABS(SECFED(IHOS1)/SDSF3)
            IHOS2=INT(IHOS1/6.)
            CORRES(IHOS+3)=ABS(SEISMO(IHOS2)/SDSE3)
            IHOS=IHOS+12
            ENDDO
          ENDIF
          IFL=IFL+12
          IFL2=IFL+200
          IFL3=IFL+400
C
            Results are combined as a part of the array TAPRES.
C
          DO 348 IR=1,200
          TAPRES(IR+IFL2)=CORRES(IR)
          TAPRES(IR+IFL3)=CORRES(IR+200)
  348     TAPRES(IR+IFL)=RESULT(IR)
C
C           If TAPRES is full, ie contains the results from 13 groups
C           shown when RFLAG1=13 then it is written to tape using WTTAPE,
C           in the form of the character array CHRES. The equivalence
C           between TAPRES and CHRES is defined at the start.
C
          IF (RFLAG1.EQ.13) THEN
            RFLAG1=0
C
C           Should WTTAPE fail then it is called again and again up to 10
C           times after which if CHRES is still not written the program
C           is terminated.
C
  916       CALL WTTAPE(1,CHRES,GOOD)
            IF (.NOT.GOOD) THEN
              WRITE(6,*) 'ERROR WRITING RESULTS AT LOOP   ',L
              IERR=IERR+1
              IF (IERR.EQ.10) THEN
                GO TO 848
              ENDIF
              GO TO 916
            ENDIF
            WRITE(6,*) 'RESULTS WRITTEN TO TAPE.'
          ENDIF
          IERR=0
C
C           If L=NLOOP the main loop is over.
C
  251     IF (L.EQ.NLOOP) THEN
            GO TO 550
          ENDIF
C
C           Next group of secondary error point data sent into the
C           transputer system.
C
          CALL F77_CHAN_OUT_MESSAGE (26212,SECER1,OUTCHAN)
          CALL F77_CHAN_OUT_MESSAGE (26212,SECER2,OUTCHAN)
          CALL F77_CHAN_OUT_MESSAGE (26212,SECER3,OUTCHAN)
          CALL F77_CHAN_OUT_MESSAGE (26212,SECER4,OUTCHAN)
          GO TO 350
C
C           End of the loop.
C
  550     SECER1(6553)=-1
```

```
C
C       Call computer clock again so that total time for the
C       main looping procedure is known.
C
      CALL ICLOCK(SEC2)
C
C       Final results back from SECD2.
C
      CALL F77_CHAN_IN_MESSAGE (800,RESULT,IN)
      NUM=NUM+RESULT(1)
      WRITE(6,*) 'RESULTS BACK.',L,NLOOP,LS,NUM
C
C       Process termination order sent out into the system.
C
      CALL F77_CHAN_OUT_MESSAGE (26212,SECER1,OUTCHAN)
C
C       Overall histogram statistics received from SECD2.
C
      CALL F77_CHAN_IN_MESSAGE (2000,Z,IN)
      WRITE(6,*) 'NO MORE SENT'
      WRITE(40,*) Z
      WRITE(6,*) 'STATS GOT BACK.'
C
C       Final results back from CORR1 and CORR2.
C
      CALL F77_CHAN_IN_MESSAGE (1600,CORRES,IN2)
      WRITE(6,*) 'CORR RESULTS BACK.'
      IFL=612*RFLAG1
      TAPRES(IFL+1)=NTAPE
      TAPRES(IFL+2)=ITIME(1)
      TAPRES(IFL+3)=ITIME(2)
      TAPRES(IFL+4)=ITIME(3)
      TAPRES(IFL+5)=ITIME(4)
      F6BLK=(NBLK-13-F1BLK)
      IF (F6BLK.LT.0) THEN
        TAPRES(IFL+6)=ITIME(5)-1
        TAPRES(IFL+7)=60-F6BLK*.3276
      ELSE
        TAPRES(IFL+6)=ITIME(5)
        TAPRES(IFL+7)=F6BLK*.3276+F1TIME
      ENDIF
      TAPRES(IFL+8)=SDMC1+SDMC2+SDMC3
      TAPRES(IFL+9)=SDPM1+SDPM2+SDPM3
      TAPRES(IFL+10)=SDSE1+SDSE2+SDSE3
      TAPRES(IFL+11)=SDSF1+SDSF2+SDSF3
      IFL=IFL+12
      IFL2=IFL+200
      IFL3=IFL+400
      DO 349 IR=1,200
      TAPRES(IR+IFL2)=CORRES(IR)
      TAPRES(IR+IFL3)=CORRES(IR+200)
349   TAPRES(IR+IFL)=RESULT(IR)
      IFL4=612*(RFLAG1+1)+1
      DO 864 I5=IFL4,8192
864   TAPRES(I5)=0
C
C       Last array written to tape regardless of wether its full
C       or not.
C
      CALL WTTAPE(1,CHRES,GOOD)
      IF (.NOT.GOOD) THEN
        WRITE(6,*) 'ERROR WRITING RESULTS AT LOOP  ',L
        STOP
      ENDIF
      WRITE(6,*) 'RESULTS WRITTEN TO TAPE.'
C
C       Correlation histogram returned.
C
      CALL F77_CHAN_IN_MESSAGE (400,SRES,IN2)
      WRITE(43,*) SRES
      WRITE(6,*) 'CORR STATS GOT BACK.'
848   CLOSE (40)
      CLOSE (42)
      CLOSE (43)
      CLOSE (45)
C
C       If task halted due to a write fault then IERR=10.
C
      IF (IERR.EQ.10) THEN
        WRITE(6,*) 'FILES CLOSED,PROGRAM TERMINATED.'
        NULOOP=INT(NLOOP/13.)
        WRITE(6,*) 'NUMBER OF BLKS WRITTEN',NULOOP
        NEMPTY1(1)=NEMPTY1(1)+NULOOP
        WRITE(544,*) NEMPTY1
        CLOSE (544)
        STOP
      ENDIF
      NULOOP=13*INT(NLOOP/13.)
      IF (NLOOP.GT.NULOOP) THEN
        NULOOP=INT(NULOOP/13)+1
      ELSE
        NULOOP=INT(NULOOP/13)
      ENDIF
      WRITE(6,*) 'NUMBER OF BLKS WRITTEN',NULOOP
      NEMPTY1(1)=NEMPTY1(1)+NULOOP
      WRITE(544,*) NEMPTY1
      CLOSE (544)
      SEC2=SEC2-SEC1
      TMS=REAL(SEC2)/REAL(NLOOP)
      WRITE(6,*) 'TIME FOR ',NLOOP,'  LOOPS IS,',SEC2
      WRITE(6,*) 'TIME PER LOOP',TMS
      GAU2=100.*GAU2/REAL(NLOOP)
      WRITE(6,*) 'PERCENTAGE "GOOD" DATA,    ',GAU2
      STOP
      END
      PROGRAM SECD1
      INCLUDE 'CHAN.INC'
      DIMENSION SECERR(32768),SEC1(6553),SEC2(6553),SEC3(6553),
     +SEC4(6553),SEC5(6553),SLOPE(20),SECC(1024),
     +EVENT(200),Z(160),SEC15(6553),SDT(4)
      INTEGER INCHAN,OUTCHAN,IN,OUT
      REAL ICOMB(21),MEAN,COMB
      EQUIVALENCE (SECERR(1),SEC1(1)),(SECERR(6553),SEC2(1)),
     +(SECERR(13105),SEC3(1)),(SECERR(19657),SEC4(1)),
     +(SECERR(26209),SEC5(1))
C
C        Software connectionms are defined.
C        port(0) is the root
C        port(1) is the 3rd.
C
      INCHAN = F77_CHAN_IN_PORT (0)
      OUTCHAN = F77_CHAN_OUT_PORT (0)
      IN = F77_CHAN_IN_PORT (1)
      OUT = F77_CHAN_OUT_PORT (1)
      DATA ICOMB/21*1/,SLOPE/20*0/,SDT/4*0/
      SDT(1)=-1
      FLAG1=0
      COMB=0
      SSD=0
      SSD1=0
      A=1
      DATA SECERR/32768*0./
C
C       Sec1 is called once prior to the start of the main loop, because
C       initially 5 blks are sent to form the first group.
C
      CALL F77_CHAN_IN_MESSAGE (26212,SEC1,INCHAN)
100   IF (SEC1(6553).EQ.1) THEN
        SDT(1)=1.
        IF (COMB.EQ.1) THEN
          DO 811 I11=1,1024
811       SECC(I11)=SEC1(I11)
          CALL CALCOM(SECC,ICOMB,SLOPE)
          COMB=2
        ELSEIF (COMB.EQ.0) THEN
          COMB=1
        ENDIF
      ELSEIF (COMB.EQ.2) THEN
        COMB=0
        SDT(1)=0
      ENDIF
      IF (SEC1(6553).EQ.-1) THEN
        SECERR(32761)=-1
        CALL F77_CHAN_OUT_MESSAGE (131072,SECERR,OUT)
        CALL F77_CHAN_OUT_MESSAGE (16,SDT,OUT)
        STOP
      ENDIF
C
C       If sec*(6553)=1,a calibration comb is set at this point in
C       the time series.
C       If sec*(6553)=-1 program is terminated,and a message is sent
C       to stop the other transputers.
C
C       Sec2 is the first block to be taken from the root in every
C       group of 4 and is always the second block of every group.
C       If sec2(6553)=5 then mid run stats are asked for.
C
      CALL F77_CHAN_IN_MESSAGE (26212,SEC2,INCHAN)
      IF (SEC2(6553).EQ.1) THEN
        SDT(1)=1.
C
C       When SEC2(6553)=1 and COMB=0 then a comb was found to present
C       at some point in the current block.
C       Only when COMB=1 is it certain that the comb was applied
C       throughout the whole of the current block and hence any
C       part of that block can be used to update the comb used in
C       the program to calibrate the data. The comb is updated by
C        the routine CALCOM.
C
       IF (COMB.EQ.1) THEN
          DO 812 I22=1,1024
812       SECC(I22)=SEC2(I22)
          CALL CALCOM(SECC,ICOMB,SLOPE)
          COMB=2
        ELSEIF (COMB.EQ.0) THEN
          COMB=1
        ENDIF
C
C         If COMB=2 then the prevous blk contained a comb and the
C         recalibration has been carried out.
C
      ELSEIF (COMB.EQ.2) THEN
        COMB=0
        SDT(1)=0
      ENDIF
      IF (SEC2(6553).EQ.-1) THEN
        SECERR(32761)=-1
        CALL F77_CHAN_OUT_MESSAGE (131072,SECERR,OUT)
```

```
          CALL F77_CHAN_OUT_MESSAGE (16,SDT,OUT)
          STOP
        ENDIF
        IF (SEC2(6553).EQ.5) THEN
          FLAG1=1
        ENDIF
C
C        Sec3 is the second block to be taken from the root in every
C        group of 4 and is always the third block of every group.
C
        CALL F77_CHAN_IN_MESSAGE (26212,SEC3,INCHAN)
        IF (SEC3(6553).EQ.1) THEN
          SDT(1)=1.
          IF (COMB.EQ.1) THEN
          DO 813 I33=1,1024
813         SECC(I33)=SEC3(I33)
            CALL CALCOM(SECC,ICOMB,SLOPE)
            COMB=2
          ELSEIF (COMB.EQ.0) THEN
            COMB=1
          ENDIF
        ELSEIF (COMB.EQ.2) THEN
          COMB=0
          SDT(1)=0
        ENDIF
        IF (SEC3(6553).EQ.-1) THEN
          SECERR(32761)=-1
          CALL F77_CHAN_OUT_MESSAGE (131072,SECERR,OUT)
          CALL F77_CHAN_OUT_MESSAGE (16,SDT,OUT)
          STOP
        ENDIF
C
C         Sec4 is the third block to be taken from the root in every
C         group of 4 and is always the fourth block of every group.
C
        CALL F77_CHAN_IN_MESSAGE (26212,SEC4,INCHAN)
        IF (SEC4(6553).EQ.1) THEN
          SDT(1)=1.
          IF (COMB.EQ.1) THEN
            DO 814 I44=1,1024
814         SECC(I44)=SEC1(I44)
            CALL CALCOM(SECC,ICOMB,SLOPE)
            COMB=2
          ELSEIF (COMB.EQ.0) THEN
            COMB=1
          ENDIF
        ELSEIF (COMB.EQ.2) THEN
          COMB=0
          SDT(1)=0
        ENDIF
        IF (SEC4(6553).EQ.-1) THEN
          SECERR(32761)=-1
          CALL F77_CHAN_OUT_MESSAGE (131072,SECERR,OUT)
          STOP
        ENDIF
C
C        Sec5 is the fourth block to be taken from the root in every
C        group of 4 and is always the fifth block of every group.
C
        CALL F77_CHAN_IN_MESSAGE (26212,SEC5,INCHAN)
        IF (SEC5(6553).EQ.-1) THEN
          SECERR(32761)=-1
          CALL F77_CHAN_OUT_MESSAGE (131072,SECERR,OUT)
          CALL F77_CHAN_OUT_MESSAGE (16,SDT,OUT)
          STOP
        ENDIF
C
C         The 5th data block in a group is saved as it is not analysied
C         with the other blocks in that group. It becomes the 1st block
C         of the next group. The array SEC5 is copied to SEC15.
C
        DO 610 I2=1,6553
610     SEC15(I2)=SEC5(I2)
C
C         The equivalence command at the start of the program automatically
C         occupies the first 32760 points of the array SECERR with the 5
C         approprieate blocks. The last 8 points of SECERR are filled with
C         zeros.
C
        DO 611 I3=32761,32768
611     SECERR(I3)=0.
C
C         The mean of the raw data is written to SDT(2).
C
        DO 612 I4=1,26208
612     SDT(2)=SDT(2)+SECERR(I4)
        SDT(2)=SDT(2)/26208.
C
C         The FFT of SECERR is taken, followed by calibration and
C         weighting routines.
C
        CALL REALFT(SECERR,16384,1)
C
C         The necessary multiplicative factor 1/16384 is incorperated
```

```
C          in UPDCOM.
C
        CALL UPDCOM(SECERR,SLOPE,ICOMB)
        CALL WEIGHT(SECERR,SSD,SSD1)
        SDT(3)=SSD
        SDT(4)=SSD1
        IF (FLAG1.EQ.1) THEN
          SECERR(32761)=5555
          FLAG1=0
        ENDIF
C
C          Data sent to third transputer
C
        CALL F77_CHAN_OUT_MESSAGE (131072,SECERR,OUT)
        CALL F77_CHAN_OUT_MESSAGE (16,SDT,OUT)
        SDT(2)=0.
        SDT(3)=0.
        SDT(4)=0.
C
C          The last block of the group becomes the first block of
C          the next group.
C
        DO 101 L=1,6553
  101   SEC1(L)=SEC15(L)
        GO TO 100
        END


C
C          CALCOM  updates the calibration comb array ICOMB when a
C          new comb is applied in the data stream.
C          The first 4096 points of the block containg the comb are
C          used.
C

        SUBROUTINE CALCOM(SEC,ICOMB,SLOPE)
        DIMENSION SEC(4096),SLOPE(20)
        REAL ICOMB(21)
        INTEGER LOC(21)
C
C          The FFT is found.
C          The combs in the Fourier plane are separated by 198
C          points, with the first (complex point) at 97 and 98.
C          ICOMB is the inverse comb.
C
        CALL REALFT (SEC,2048,1)
        LOC(1)=96+2
C
C          Multiplicative factor 8=2048/16384.
C          2048 corrects this routines REALFT.
C          16384 corrects the main REALFT in the main body of the task.
C
        ICOMB(1)=(1.19E-13)/(8*(SEC(LOC(1))**2+SEC(LOC(1)-1)**2)**.5)
        DO 10 I=2,21
        LOC(I)=(2*I-1)*96+2
C
C          Comb at 234.375Hz is 1.19e-13m,all others 1.19e-16m.
C
        ICOMB(I)=(1.19E-16)/(8*(SEC(LOC(I))**2+SEC(LOC(I)-1)**2)**.5)
   10   CONTINUE
C
C          SLOPE(K) is the slope of the line joining inverse
C          combs ICOMB(K) and ICOMB(K+1).
C
        DO 40 K=1,20
   40   SLOPE(K)=(ICOMB(K+1)-ICOMB(K))/768.
        RETURN
        END

C
C          UPDCOM calibrates the data SEC(32768) on the basis of the
C          combs calculated in CALCOM.
C

        SUBROUTINE UPDCOM(SEC,SLOPE,ICOMB)
        DIMENSION SEC(32768),SLOPE(20)
        REAL ICOMB(21),L
        INTEGER LOC(21)
        LOC(1)=770
C
C          1/16384, correcting for REALFT is built into ICOMB in CALCOM.
C
C          Data prior to the first comb is calibrated according to the
C          height of the first comb only.
C
  100   DO 50 I=1,LOC(1)
   50   SEC(I)=SEC(I)*ICOMB(1)
C
C          Data between combs is calibrated using the lower frequency
C          and the calculated slope of the line joining it with the
C          higher frequency comb.
C
        DO 60 J=2,21
        LOC(J)=(2*J-1)*768+2
```

```
      L=O
      DO 60 I=LOC(J-1)+2,LOC(J),2
      SEC(I)=SEC(I)*(ICOMB(J-1)+L)
      SEC(I-1)=SEC(I-1)*(ICOMB(J-1)+L)
      L=L+SLOPE(J-1)
60    CONTINUE
C
C       Data beyond the last comb is calibrated according to the
C       height of the last comb only.
C
      DO 70 K2=LOC(21)+1,32768
70    SEC(K2)=SEC(K2)*ICOMB(21)
      RETURN
      END

C
C       WEIGHT routine reduces the influence of bad, high amplitude,
C       noise relative to the better, low amplitude noise.
C
      SUBROUTINE WEIGHT(SECERR,SSD,SSD1)
      DIMENSION SECERR(32768),VAR(128)
      REAL PS(128),SSD,SSD1
C
C       Data divided up into 128 groups of 128 complex points.
C
      DO 20 J=1,128
      PS(J)=0
      K1=(J-1)*256+2
C
C       Varience for each of the 128 groups is found (VAR(j).)
C       Zero mean is assumed for simplicity.
C       Multiplicative factor, 1.E+5, used to prevent the chance
C       of floating point overflow in the program.
C
      DO 30 K=K1,K1+254,2
      SECERR(K)=SECERR(K)*1.E+5
      SECERR(K-1)=SECERR(K-1)*1.E+5
30    PS(J)=PS(J)+SECERR(K)*SECERR(K)+SECERR(K-1)*SECERR(K-1)
20    VAR(J)=PS(J)/128.
      SSD=0.
      SSD1=0.
      DO 40 J1=1,128
      K2=(J1-1)*256+2
C
C       Each of the groups is then devided by its varience.
C
      DO 40 K=K2,K2+254,2
      SECERR(K)=((1.E+5)*SECERR(K))/VAR(J1)
40    SECERR(K-1)=((1.E+5)*SECERR(K-1))/VAR(J1)
C
C       The mean standerd deviation for the frequency range that may
C       contain Chirp data is found.
C
      DO 50 J2=3,13
50    SSD=SSD+VAR(J2)
      SSD=(SSD/11.)**(.5)*1.E-5
C
C       Also the mean standerd deviation for the whole frequency range
C       is found.
C
      DO 60 J2=1,128
60    SSD1=SSD1+VAR(J2)
      SSD1=(SSD1/128.)**(.5)*1.E-5
      RETURN
      END
      PROGRAM SECD2
      INCLUDE 'CHAN.INC'
      DIMENSION SECERR(32768),Y(500),EVENT(200),Z(160),SDT(4)
      INTEGER INCHAN,OUTCHAN,IN,OUT,IN2,OUT2
      REAL MEAN,Y1
      INCHAN = F77_CHAN_IN_PORT (0)
      OUTCHAN = F77_CHAN_OUT_PORT (0)
      IN = F77_CHAN_IN_PORT (1)
      OUT = F77_CHAN_OUT_PORT (1)
      IN2 = F77_CHAN_IN_PORT (2)
      OUT2 = F77_CHAN_OUT_PORT (2)
      DATA Y/500*0/
      FLAG=0
C
C       Data read from task SECD1. The 32761th pt is checked for
C       termination (set to -1) or if stats wanted (set to 5555).
C
 100  CALL F77_CHAN_IN_MESSAGE (131072,SECERR,INCHAN)
      CALL F77_CHAN_IN_MESSAGE (16,SDT,INCHAN)
      IF (SECERR(32761).EQ.5555) THEN
        FLAG=1
        TFLAG=SECERR(4097)
        SECERR(4097)=5555
        SECERR(32761)=0
      ELSEIF (SECERR(32761).EQ.-1) THEN
        CALL F77_CHAN_OUT_MESSAGE (2000,Y,OUT)
        SECERR(4097)=-1
        CALL F77_CHAN_OUT_MESSAGE (16392,SECERR,OUT2)
        STOP
      ENDIF
C
C       Data is cut of at 1250 Hz,and sent on to the 4th transputer.
C       Of the 4098 points sent, the first 4096 of them contain data.
C       Point 4097 is set to -1 if the process is to be terminated.
C       Point 4098 is temporarily set to SDT(3).
C
      TFLAG1=SECERR(4098)
      SECERR(4098)=SDT(3)
      CALL F77_CHAN_OUT_MESSAGE (16392,SECERR,OUT2)
      SECERR(4098)=TFLAG1
      IF (FLAG.EQ.1) THEN
        SECERR(4097)=TFLAG
      ENDIF
C
C       Find the IFFT.
C
      CALL REALFT (SECERR,16384,-1)
      SD=0
      MEAN=0
C
C       Calling STATS routine which returns standard deviation of data
C       along with its mean and also a data histogram (Y) and a
C       calculated gaussian assessment parameter (Y1).
C
      CALL STATS (SECERR,SD,MEAN,Y,Y1,SDT)
C
C       SEARCH routine which carries out an event search in the data
C       using the sd and mean calculated in the previous routine.
C       It returns its results in the array EVENT.
C
      CALL SEARCH (SECERR,SD,MEAN,EVENT,SDT)
      EVENT(4)=SDT(1)
      EVENT(5)=Y1
      EVENT(6)=SDT(2)
      EVENT(7)=SDT(4)
C
C       Results sent back to the root.
C
      CALL F77_CHAN_OUT_MESSAGE (800,EVENT,OUT)
      IF (FLAG.EQ.1) THEN
        CALL F77_CHAN_OUT_MESSAGE (2000,Y,OUT)
        FLAG=0
      ENDIF
      GO TO 100
      END

C
C       STATS routine in which the mean and standard deviations of
C       the first 4 blocks worth of data are found, along with a
C       histogram binning the data according to each points displacement
C       from the group mean. Also calculated is the gaussian assessment
C       parameter (Z).
C

      SUBROUTINE STATS(SEGMNT,SD,MEAN,Y,Z,SDT)
      DIMENSION SEGMNT(32768),Y(500),SDT(4),Y1(500)
      REAL MEAN,SD,INC,Z
      SD=0.
      MEAN=0.
C
C       Mean and sd of the first 4blk's of each group are found.
C
      DO 10 I=1,26208
      MEAN=MEAN+SEGMNT(I)
      SEG=SEGMNT(I)*1.E-10
10    SD=SD+SEG*SEG
      MEAN=MEAN/26208.*1.E-20
      SD=(1.E+10)*(SD/26208.-MEAN**2)**0.5
      MEAN=MEAN*1.E+20
C
C       Histogram of 500 hundred bins is formed.
C       Going from 0 to 10*sd with the width of each bin, sd/50.
C
      INC=50./SD
c     INC=1./SD
      DO 36 J2=1,500
36    Y1(J2)=0
      DO 35 L=1,26208
      J=INT(ABS(SEGMNT(L)-MEAN)*INC)
      IF (J.GT.499) THEN
        Y1(500)=Y1(500)+1
        GO TO 35
      ENDIF
      Y1(J+1)=Y1(J+1)+1
35    CONTINUE
C
C       Gaussian parameter calculated by suming the first 5 bins
C       then dividing the total by the number of elements that
C       should occupy them if the data is perfectly gaussian.
C
      Z=(Y1(1)+Y1(2)+Y1(3)+Y1(4)+Y1(5))/2088.
C
C       The histogram data is added to the group accumulative
C       histogram data.
C
      DO 37 L2=1,500
37    Y(L2)=Y(L2)+Y1(L2)
38    RETURN
      END

C
C       Routine that searches the first four blocks of a data group
C       for events defined as elements of the data stream whose
C       deviation from the group mean exceed a given multiple
C        (threshold) of the standard deviation of the element.
C

      SUBROUTINE SEARCH(SEGMNT,SD,MEAN,EVENT,SDT)
      DIMENSION SEGMNT(32768),EVENT(200),SDT(4)
      REAL MEAN,SD,TRH
      INTEGER NEVENT,FLAG
      NEVENT=0
      THRESHOLD=4.
      IBIGB=0
      IBIGE=0
      TRH=THRESHOLD*SD
      FLAG=0
C
C       Each group has a 200 element array, EVENT, into which results
C       are written. Each S/N threshold crosser found has 10 elements
C       associated with it written to the array hence only the first 19
C       events are recorded.
C
      DO 11 J=1,200
11    EVENT(J)=0.
C
C       An event search over the first 26208 elements of the block
C       is carried out. Each event is treated as a potential group,
C       hence nothing is written to EVENT until the end of the group,
C       ie the last threshold crosser is found.
C
      DO 10 I=1,26208
      IF (ABS(SEGMNT(I)-MEAN).GE.TRH) THEN
C
C       If flag=0 then first event in group is found.
C
        IF (FLAG.EQ.0) THEN
          FLAG=1
          BIG=ABS(SEGMNT(I)-MEAN)
          IBIGB=I
          IBIG=I
C
C         If confirmed event is larger than previous biggest event
C         in group then it becomes the new peak.
C
        ELSEIF (ABS(SEGMNT(I)-MEAN).GE.BIG) THEN
          BIG=ABS(SEGMNT(I)-MEAN)
          IBIG=I
        ENDIF
        IBIGE=I
C
C       If current point is not an event and flag=1 then event group
C       is over and results are written to the array EVENT.
C
      ELSEIF (FLAG.EQ.1) THEN
        FLAG=0
        NEVENT=NEVENT+1
        EVENT(NEVENT*10-2)=NEVENT       ! Event group number (1-19).
        EVENT(NEVENT*10-1)=IBIGB        ! Start of event group.
        EVENT(NEVENT*10)=IBIG           ! Position of peak in group.
        EVENT(NEVENT*10+1)=IBIGE        ! End of event group.
        BIG=BIG/SD
        EVENT(NEVENT*10+2)=BIG          ! Peak S/N.
        EVENT(NEVENT*10+3)=BIG*SDT(4)/(2*10000*10000/32768.)**(.5)
C
C         Actual amplitude of event is calculated and written to
C         EVENT(NEVENT*10+3).
C
        IBIGB=0
        IBIGE=0
      ENDIF
C
C       Only 19 events possible.
C
      IF (NEVENT.EQ.19) THEN
        GO TO 30
      ENDIF
 10   CONTINUE
C
C       Event array started with the total number of event groups.
C       This is followed by the standard deviation and mean of the
C       relevant part(first 26208 points) of the whole group.
C
 30   EVENT(1)=NEVENT*1.
      EVENT(2)=SD
      EVENT(3)=MEAN
      RETURN
      END
      PROGRAM CORR1
      INCLUDE 'CHAN.INC'
      DIMENSION C(32768),FILTERS(49200),CORR1(4096),FDATA(4098),
     +EVENT(200),BIG(3),COF(3),CORR2(4096),FILNORM(7)
      REAL MEAN1,MEAN2,PS1,PS2,IBIG(3)
      INTEGER PAR,PHASE,INCHAN,OUTCHAN,FLAG,OUT,IN
C
C        The following 4 statement functions are used to calculate the
C        filters on the basis of equations given in Chapter 2.
C        A( ),B( ) and Chirp( ) define the chirp and come from Equation 2.7
C        and TD( ) is Equation 2.9.
C
      A(TIM,PAR)=(1-0.34*TIM*PAR)**(-.25)
      B(TIM,PAR)=(1005.31*(1-A(TIM,PAR)**(-5./2))/(.34*PAR))
      CHIRP(TIM,PAR,PHASE)=A(TIM,PAR)*COS(B(TIM,PAR)+PHASE*1.570796)
      TD(PAR)=(3.**(-8./3)-10.**(-8./3))/(.34*PAR)
C
C        This tasks software connections are defined.
C        PORT (0) is connected to SECD1,
C        PORT (0) is connected to CORR2.
C
      INCHAN = F77_CHAN_IN_PORT (0)
      OUTCHAN = F77_CHAN_OUT_PORT (0)
      IN = F77_CHAN_IN_PORT (1)
      OUT = F77_CHAN_OUT_PORT (1)
      DATA C/32768*0/,FILNORM/7*0/
      DATA CORR1/4096*0/,CORR2/4096*0/,BIG/3*0/,IBIG/3*0/
      DATA EVENT /200*0/,COF/3*0/,FILTERS/49200*0/
C
C        S/N event threshold is defined.
C
      THRESHOLD=3.5
      INC=0
CC
CC       The following loop utilises the statement functions to build
CC       up the filter template. The 10 filters are based on 5 distingt
CC       mass parameters over which the outer loop is incremented, with
CC       2 phases (0 and pi/2) for each mass parameter.
CC
C
C          Mass parameter(PAR) going from 2 to 6.
C
      DO 10 PAR=2,6
      DO 10 PHASE=0,1
C
C          T is the starting time for the chirp, ie when it reaches 300Hz.
C
      T=(1-3.**(-8/3))/(.34*PAR)
C
C        NPTS is the number of non zero points that compose the chirp
C        sampled at 20000Hz.
C
      NPTS=INT(TD(PAR)*20000.)+1
C
C        The chirp is built up with the non zero elements at the start
C        filling the remaing elements of the array up with zeros.
C
      DO 30 I=1,NPTS
      C(I)=CHIRP(T,PAR,PHASE)
   30 T=T+.00005
      DO 31 I4=NPTS+1,32768
   31 C(I4)=0.
C
C        The FFT of the chirp is found which replaces the array C with
C        a complex array that contains the positive frequency half of the
C        FFT only, form 0-10000Hz.
C        C(2) does not represent the imaginary part of the first point hence
C        it is set to zero.
C
      CALL REALFT(C,16384,1)
      C(2)=0.
C
C        FILNORM is a quantity calculated for each filter. It is the
C        Standerd Deviation of the filter from 300-1000Hz and is used later
C        in the program to infer an actual amplitude from any events found.
C
      IF (PHASE.EQ.0) THEN
        DO 18 I18=983,3277
   18   FILNORM(PAR)=C(I18)**2+FILNORM(PAR)
        FILNORM(PAR)=FILNORM(PAR)*10000./32768.
      ENDIF
C
C        The first 4096 (1250Hz) of the FFTs  of each of the filters is
C        appended in turn  to the array filters.
C
      DO 40 J=1,4096
   40 FILTERS(J+INC)=C(J)
   10 INC=INC+4096
      y1=1
      SDT=1
CC
CC       This point represents the start of the main loop within the
CC       program.
CC
```

```
 2000   NEVENT=0
        INC=0
  C
  C          A 4098 point array is recieved from the previous task. The first
  C          4096 points are the first 1250Hz of the data stream to be correlated
  C          with the filters. The 4097th point convays instructions to the task.
  C          If it is set to -1 the task is terminated.
  C          If it is set to 5555, the task is requested to transmit histogram
  C          data back to the root task.
  C          The 4098th point contains the standard deviation of the frequency
  C          region in which chirp filters are defined.
  C          This is allocated to the variable SDT.
  C
        CALL F77_CHAN_IN_MESSAGE (16392,FDATA,INCHAN)
        CALL F77_CHAN_OUT_MESSAGE (16392,FDATA,OUT)
        IF (FDATA(4097).EQ.-1) STOP
        SDT=FDATA(4098)
        ICH=4
  C
  C          The 200 elements of the results array associated with each event
  C          are put to zero, as is the event counter NEVENT.
  C
        DO 101 L7=1,200
  101   EVENT(L7)=0.
        NEVENT=0
  C
  C          The loop (do 60) contains the correlation of the data with the
  C          filters in turn, counting through each of the 5 the mass parameters.
  C          Within each loop, the correlation of the data with both filters of
  C          that mass parameter is carried out and then combined onto one.
  C
        DO 60 K=1,5
  C
  C          The real and imaginary parts of the FFT's of both correlations are
  C          found.
  C          INC takes the template through each pair of filters in turn.
  C
        DO 70 K1=1,4095,2
        K11=K1+INC
        CORR1(K1)=FDATA(K1)*FILTERS(K11)+FDATA(K1+1)*FILTERS(K11+1)
        CORR1(K1+1)=FDATA(K1+1)*FILTERS(K11)-FDATA(K1)*FILTERS(K11+1)
        K22=K11+4096
        CORR2(K1)=FDATA(K1)*FILTERS(K22)+FDATA(K1+1)*FILTERS(K22+1)
        CORR2(K1+1)=FDATA(K1+1)*FILTERS(K22)-FDATA(K1)*FILTERS(K22+1)
  70    CONTINUE
        INC=INC+8192
  C
  C          Time series correlations are found by taking the IFFT.
  C
        CALL REALFT(CORR1,2048,-1)
        CALL REALFT(CORR2,2048,-1)
  C
  C          Means and standerd deviations for both streams (1 and 2) are
  C          found.
  C          TT is a necessary multiplicative factor inorder to complete
  C          the IFFT.
  C
        MEAN1=0.
        PS1=0.
        MEAN2=0.
        PS2=0.
        TT=1/2048.
  C
  C          3276=4096*26208/32768
  C          The 4096 correlation elements represent 1.6384 seconds of original
  C          data which is 32768 elements or just over 5 blocks. Hence as only 4
  C          blocks are analysed each time only the first 3276 points of the
  C          4096 need be statistically analysed.
  C
        DO 100 L=1,3276
        CORR1(L)=CORR1(L)*TT
        MEAN1=MEAN1+CORR1(L)
        PS1=PS1+(CORR1(L))**2
        CORR2(L)=CORR2(L)*TT
        MEAN2=MEAN2+CORR2(L)
  100   PS2=PS2+(CORR2(L))**2
        MEAN1=MEAN1/3276.
        MEAN2=MEAN2/3276.
        SD1=(PS1/3276.-MEAN1**2)**.5
        SD2=(PS2/3276.-MEAN2**2)**.5
  C
  C          The correlation streams are combined for the event search,
  C          hence the S/N threshold (TRH) that must be exceded for an event to
  C          be recorded must be worked out on the basis of the standerd
  C          deviations of both streams.
  C
        TRH=THRESHOLD*(SD1**2+SD2**2)**.5
  C
  C          FLAG is always 0 in the analysis loop unless the previous is a
  C          threshold crosser in which case it is set to 1. Initally it is
  C          set to 0.
  C
        FLAG=0
```

```
      DO 110 L1=1,3276
C
C          The deviation of each point, in both correlation streams, from
C          the means for that stream are found and then combined to produce
C          a single deviation.
C
      CO1=CORR1(L1)-MEAN1
      CO2=CORR2(L1)-MEAN2
      CO=(CO1**2+CO2**2)**.5
C
C          Points of interest are only those that excede the threshold.
C
      IF (CO.GE.TRH) THEN
C
C          If FLAG is set to 0 then the previous point was not an threshold
C          crosser, hence the current point is the start of a threshold
C          crossing group.
C
        IF (FLAG.EQ.0) THEN
          FLAG=1

C
C          BIG and IBIG are 3 element arrays that hold infomation about the
C          peak S/N point in an event group as well as the points either side.
C          BIG records the S/N's of the 3 points and IBIG their positions in
C          the data stream. The second point in both is the peak point.
C          The first point of an event group found is always initially assumed
C          to be the peak.
C          ISTART is the position of the first point in the group.
C
          BIG(2)=CO
          IBIG(2)=L1
          ISTART=L1
C
C          The phase of the event is calculated, taking into account into
C          which quadrant it falls.
C
          RAD=ATAN(CO2/(CO1+1.E-30))
          IF (CO1.LT.0) THEN
            RAD=RAD+3.14159
          ELSEIF (CO2.LT.0) THEN
            RAD=RAD+6.28319
          ENDIF

C
C          BIG(1) is initially calculated. If the first event is the
C          first point then BIG(1) is given the same value as BIG(2).
C
            IF (L1.EQ.0) THEN
              BIG(1)=0
              GO TO 110
            ENDIF
            BIG(1)=ABS(CORR1(L1-1)-MEAN)*COS(RAD)+
     +      ABS(CORR2(L1-1)-MEAN)*COS(RAD+1.5708)
            GO TO 110
          ENDIF
C
C           If the current point is a threshold crosser with larger S/N than
C           the previous point (also a threshold crosser) then it becomes the
C           new peak point.
C
          IF (CO.GE.BIG(2)) THEN
            BIG(2)=CO
            IBIG(2)=L1
c           RAD=ATAN(-CO2/(CO1+1.E-30))
            RAD=ATAN(CO2/(CO1+1.E-30))
            IF (CO1.LT.0) THEN
              RAD=RAD+3.14159
            ELSEIF (CO2.LT.0) THEN
              RAD=RAD+6.28319
            ENDIF
            BIG(1)=ABS(CORR1(L1-1)-MEAN)*COS(RAD)+
     +      ABS(CORR2(L1-1)-MEAN)*COS(RAD+1.5708)
          ENDIF
          GO TO 110
        ENDIF
C
C           If the current point is not an event but FLAG is set then an
C           event group is over and the results are appended to the array
C           EVENT.
C
        IF (FLAG.EQ.1) THEN
          FLAG=0
C
C           NEVENT is the number of the event and IEND the postion of the
C           last threshold crosser in the group.
C
          NEVENT=NEVENT+1
          IEND=L1
          BIG(3)=ABS(CORR1(L1)-MEAN)*COS(RAD)+
     +    ABS(CORR2(L1)-MEAN)*COS(RAD+1.5708)
C
C           The arrays IBIG and BIG are sent to the subroutine POLCOE
C           which returns the coefficents of a polynomial from which
```

```
C          a better estimate of the position of the peak is made in terms of
C          the 26208 points comprising the 4 blocks analysed.
C          IBIG(2) is set to 0 to ensure POLCOE does not excede the floating
C          point limit.
C
         IBIG(1)=IBIG(2)-1.
         IBIG(3)=IBIG(2)+1.
         CALL POLCOE(IBIG,BIG,3,COF)
         TBIG=-COF(2)/(2*COF(3)+1.E-30)
         TBIG=INT(8*TBIG+.49)
C
C          BIG(2) is converted into S/N.
C
         BIG(2)=BIG(2)*THRESHOLD/TRH
         EVENT(ICH-2)=NEVENT                     ! Event number.
         EVENT(ICH-1)=BIG(2)                     ! Peak signal to noise.
         EVENT(ICH)=BIG(2)*SDT/(2*FILNORM(K))**.5 ! Amplitude of wave.
         EVENT(ICH+1)=TBIG                       ! Position of max peak.
         EVENT(ICH+2)=K+1                        ! Filter mass parameter.
         EVENT(ICH+3)=RAD                        ! Phase of event.
         EVENT(ICH+4)=ISTART*8                   ! Start of event group.
         EVENT(ICH+5)=IEND*8                     ! End of event group.
         ICH=ICH+12
       ENDIF
C
C        Total number of events cannot excede 16.
C
       IF (NEVENT.GE.16) THEN
         GO TO 61
       ENDIF
110    CONTINUE
60     CONTINUE
C
C        First number in the EVENT array set to be the total number of
C        events recorded in this group.
C
61     EVENT(1)=NEVENT
C
C        EVENT(195) set to mean of chirp frequency range.
C        This element is empty even if 16 events found.
C
       EVENT(195)=SDT
C
C         EVENT array sent on to CORR2.
C
       CALL F77_CHAN_OUT_MESSAGE (800,EVENT,OUT)
       GO TO 2000
       END
```

```
      PROGRAM CORR2
      INCLUDE 'CHAN.INC'
      DIMENSION C(32768),FILTERS(49200),CORR1(4096),CORR2(4096),BIG(3),
     +FDATA(4098),EVENT(400),COF(3),Y(100),EVENT2(200),FILNORM(5)
      REAL MEAN1,PS1,MEAN2,PS2,IBIG(3),RPAR
      INTEGER PAR,PHASE,INCHAN,OUTCHAN,FLAG,OUT,IN,IN2,OUT2
      EQUIVALENCE (EVENT(201),EVENT2(1))
C
C        The following 4 statement functions are used to calculate the
C        filters on the basis of equations given in Chapter 2.
C        A( ),B( ) and Chirp( ) define the chirp and come from Equation 2.7
C        and TD( ) is Equation 2.9.
C
      A(TIM,RPAR)=(1-0.34*TIM*RPAR)**(-.25)
      B(TIM,RPAR)=(1005.31*(1-A(TIM,RPAR)**(-5./2))/(.34*RPAR))
      CHIRP(TIM,RPAR,PHASE)=A(TIM,RPAR)*COS(B(TIM,RPAR)+PHASE*1.570796)
      TD(RPAR)=(3.**(-8./3)-10.**(-8./3))/(.34*RPAR)
C
C        This tasks software connections are defined.
C        PORT (0) is connected to CORR1,
C        PORT (1) is connected to ROOT.
C
      INCHAN = F77_CHAN_IN_PORT (0)
      OUTCHAN = F77_CHAN_OUT_PORT (0)
      IN = F77_CHAN_IN_PORT (1)
      OUT = F77_CHAN_OUT_PORT (1)



      DATA C/32768*0/,Y/100*0/,FILNORM/5*0/
      DATA CORR1/4096*0/,CORR2/4096*0/,BIG/3*0/,IBIG/3*0/
      DATA EVENT /400*0/,COF/3*0/,FILTERS/49200*0/

C
C        S/N event threshold is defined.
C
      THRESHOLD=3.5


      INC=0
      FLAG1=0

CC
CC       The following loop utilises the statement functions to build
CC       up the filter template. The 10 filters are based on 5 distingt
CC       mass parameters over which the outer loop is incremented, with
CC       2 phases (0 and pi/2) for each mass parameter.
CC
C
C         Mass parameter,(RPAR) going from 1.38305 to 1.41085.
C
      DO 50 PAR=1,5
      RPAR=1.3761+.00695*PAR
      DO 50 PHASE=0,1
C
C         T is the starting time for the chirp, ie when it reaches 300Hz.
C
      T=(1-3.**(-8/3))/(.34*RPAR)
C
C         NPTS is the number of non zero points that compose the chirp
C         sampled at 20000Hz.
C
      NPTS=INT(TD(RPAR)*20000.)+1
C
C         The chirp is built up with the non zero elements at the start
C         filling the remaing elements of the array up with zeros.
C
      DO 52 I=1,NPTS
      C(I)=CHIRP(T,RPAR,PHASE)
52    T=T+.00005
      DO 51 I2=NPTS+1,32768
51    C(I2)=0.
C
C         The FFT of the chirp is found which replaces the array C with
C         a complex array that contains the positive frequency half of the
C         FFT only, form 0-10000Hz.
C         C(2) does not represent the imaginary part of the first point hence
C         it is set to zero.
C
      CALL REALFT(C,16384,1)
      C(2)=0.
C
C         FILNORM is a quantity calculated for each filter. It is the
C         Standerd Deviation of the filter from 300-1000Hz and is used later
C         in the program to infer an actual amplitude from any events found.
C
      IF (PHASE.EQ.0) THEN
        DO 18 I18=983,3277
18      FILNORM(PAR)=FILNORM(PAR)+C(I18)**2
        FILNORM(PAR)=FILNORM(PAR)*10000./32768.
      ENDIF
C
```

```
C          The first 4096 (1250Hz) of the FFTs  of each of the filters is
C          appended in turn  to the array filters.
C
      DO 53 J=1,4096
53    FILTERS(J+INC)=C(J)
50    INC=INC+4096
C
C
      Y1=1.
      SDT=1.
CC
CC         This point represents the start of the main loop within the
CC         program.
CC
2000  NEVENT=0
      INC=0
C
C          A 4098 point array is recieved from the previous task. The first
C          4096 points are the first 1250Hz of the data stream to be correlated
C          with the filters. The 4097th point convays instructions to the task.
C          If it is set to -1 the task is terminated.
C          If it is set to 5555, the task is requested to transmit histogram
C          data back to the root task.
C          The 4098th point contains the standard deviation of the frequency
C          region in which chirp filters are defined.
C          This is allocated to the variable SDT.
C
      CALL F77_CHAN_IN_MESSAGE (16392,FDATA,INCHAN)
      IF (FDATA(4097).EQ.-1) THEN
        CALL F77_CHAN_OUT_MESSAGE (400,Y,OUT)
        STOP
      ELSEIF (FDATA(4097).EQ.5555) THEN
        FLAG1=1
      ENDIF
      SDT=FDATA(4098)
      ICH=4
C
C          The 200 elements of the results array associated with each event
C          are put to zero, as is the event counter NEVENT.
C
      DO 101 L7=1,200
101   EVENT(L7)=0.
      NEVENT=0
C
C          The loop (do 60) contains the correlation of the data with the
C          filters in turn, counting through each of the 5 the mass parameters.
C          Within each loop, the correlation of the data with both filters of
C          that mass parameter is carried out and then combined onto one.
C
      DO 60 K=1,5
C
C          The real and imaginary parts of the FFT's of both correlations are
C          found.
C          INC takes the template through each pair of filters in turn.
C
      DO 70 K1=1,4095,2
      K11=K1+INC
      CORR1(K1)=FDATA(K1)*FILTERS(K11)+FDATA(K1+1)*FILTERS(K11+1)
      CORR1(K1+1)=FDATA(K1+1)*FILTERS(K11)-FDATA(K1)*FILTERS(K11+1)
      K22=K11+4096
      CORR2(K1)=FDATA(K1)*FILTERS(K22)+FDATA(K1+1)*FILTERS(K22+1)
      CORR2(K1+1)=FDATA(K1+1)*FILTERS(K22)-FDATA(K1)*FILTERS(K22+1)
70    CONTINUE
      INC=INC+8192
C
C          Time series correlations are found by taking the IFFT.
C
      CALL REALFT(CORR1,2048,-1)
      CALL REALFT(CORR2,2048,-1)
C
C          Means and standerd deviations for both streams (1 and 2) are
C          found.
C          TT is a necessary multiplicative factor inorder to complete
C          the IFFT.
C
      MEAN1=0.
      PS1=0.
      MEAN2=0.
      PS2=0.
      TT=1/2048.
C
C          3276=4096*26208/32768
C          The 4096 correlation elements represent 1.6384 seconds of original
C          data which is 32768 elements or just over 5 blocks. Hence as only 4
C          blocks are analysed each time only the first 3276 points of the
C          4096 need be statistically analysed.
C
      DO 100 L=1,3276
      CORR1(L)=CORR1(L)*TT
      MEAN1=MEAN1+CORR1(L)
      PS1=PS1+(CORR1(L))**2
      CORR2(L)=CORR2(L)*TT
      MEAN2=MEAN2+CORR2(L)
100   PS2=PS2+(CORR2(L))**2
```

```
      MEAN1=MEAN1/3276.
      MEAN2=MEAN2/3276.
      SD1=(PS1/3276.-MEAN1**2)**.5
      SD2=(PS2/3276.-MEAN2**2)**.5
C
C         When the mass parameter reaches 5 on one of the correlation
C         streams is binned in a similar way to the time series in task
C         SECD2. In this case there only 100 bins (array Y) of size SD/10.
C         Again all displacements above 10*SD are counted in the final bin.
C
      IF (K.EQ.5) THEN
        SDINC=SD1/10.
        DO 35 L=1,3276
        J=INT(ABS(CORR1(L)-MEAN1)/SDINC)
        IF (J.GE.100) THEN
          Y(100)=Y(100)+1
          GO TO 35
        ENDIF
        Y(J+1)=Y(J+1)+1
   35   CONTINUE
      ENDIF
C
C         The correlation streams are combined for the event search,
C         hence the S/N threshold (TRH) that must be exceded for an event to
C         be recorded must be worked out on the basis of the standerd
C         deviations of both streams.
C
      TRH=THRESHOLD*(SD1**2+SD2**2)**.5
C
C         FLAG is always 0 in the analysis loop unless the previous is a
C         threshold crosser in which case it is set to 1. Initally it is
C         set to 0.
C
      FLAG=0
C
      DO 110 L1=1,3276
C
C         The deviation of each point, in both correlation streams, from
C         the means for that stream are found and then combined to produce
C         a single deviation.
C
      CO1=CORR1(L1)-MEAN1
      CO2=CORR2(L1)-MEAN2
      CO=(CO1**2+CO2**2)**.5
C
C         Points of interest are only those that excede the threshold.
C
      IF (CO.GE.TRH) THEN
C
C         If FLAG is set to 0 then the previous point was not an threshold
C         crosser, hence the current point is the start of a threshold
C         crossing group.
C
        IF (FLAG.EQ.0) THEN
          FLAG=1
C
C          BIG and IBIG are 3 element arrays that hold infomation about the
C          peak S/N point in an event group as well as the points either side.
C          BIG records the S/N's of the 3 points and IBIG their positions in
C          the data stream. The second point in both is the peak point.
C          The first point of an event group found is always initially assumed
C          to be the peak.
C          ISTART is the position of the first point in the group.
C
          BIG(2)=CO
          IBIG(2)=L1
          ISTART=L1
C
C          The phase of the event is calculated, taking into account into
C          which quadrant it falls.
C
          RAD=ATAN(CO2/(CO1+1.E-30))
          IF (CO1.LT.0) THEN
            RAD=RAD+3.14159
          ELSEIF (CO2.LT.0) THEN
            RAD=RAD+6.28319
          ENDIF
C
C          BIG(1) is initially calculated. If the first event is the
C          first point then BIG(1) is given the same value as BIG(2).
C
          IF (L1.NE.1) THEN
            BIG(1)=(ABS(CORR1(L1-1)-MEAN)**2+
     +      ABS(CORR2(L1-1)-MEAN)**2)**0.5
          ELSE
            BIG(1)=BIG(2)
          ENDIF
          GO TO 110
          ENDIF
C
C         If the current point is a threshold crosser with larger S/N than
C         the previous point (also a threshold crosser) then it becomes the
C         new peak point.
C
```

```
        IF (CO.GE.BIG(2)) THEN
          IF (L1.EQ.IBIG(2)+1) THEN
            BIG(1)=BIG(2)
          ELSE
            BIG(1)=(ABS(CORR1(L1-1)-MEAN)**2+
     +      ABS(CORR2(L1-1)-MEAN)**2)**0.5
          ENDIF
          BIG(2)=CO
          IBIG(2)=L1
          RAD=ATAN(CO2/(CO1+1.E-30))
          IF (CO1.LT.0) THEN
            RAD=RAD+3.14159
          ELSEIF (CO2.LT.0) THEN
            RAD=RAD+6.28319
          ENDIF
        ENDIF
        GO TO 110
      ENDIF
C
C         If the current point is not an event but FLAG is set then an
C         event group is over and the results are appended to the array
C         EVENT.
C
      IF (FLAG.EQ.1) THEN
        FLAG=0
C
C         NEVENT is the number of the event and IEND the postion of the
C         last threshold crosser in the group.
C
        NEVENT=NEVENT+1
        IEND=L1-1
        BIG(3)=(ABS(CORR1(IBIG(2)+1)-MEAN)**2+
     +  ABS(CORR2(IBIG(2)+1)-MEAN)**2)**0.5
C
C         The arrays IBIG and BIG are sent to the subroutine POLCOE
C         which returns the coefficents of a polynomial from which
C         a better estimate of the position of the peak is made in terms of
C         the 26208 points comprising the 4 blocks analysed.
C         IBIG(2) is set to 0 to ensure POLCOE does not excede the floating
C         point limit.
C
        IBIGT=IBIG(2)
        IBIG(2)=0.
        IBIG(1)= -1.
        IBIG(3)= +1.
        CALL POLCOE(IBIG,BIG,3,COF)
        TBIG=-COF(2)/(2*COF(3)+1.E-30)+IBIGT
        TBIG=INT(8*TBIG+.49)
C
C         BIG(2) is converted into S/N.
C
        BIG(2)=BIG(2)*THRESHOLD/TRH
        EVENT(ICH-2)=NEVENT                         ! Event number.
        EVENT(ICH-1)=BIG(2)                         ! Peak signal to noise.
        EVENT(ICH)=BIG(2)*SDT/(2*FILNORM(K))**.5    ! Amplitude of wave.
        EVENT(ICH+2)=1.3761+.00695*K                ! Filter mass parameter.
        EVENT(ICH+3)=RAD                            ! Phase of event.
        EVENT(ICH+4)=ISTART*8                       ! Start of event group.
        EVENT(ICH+5)=IEND*8                         ! End of event group.
        ICH=ICH+12
      ENDIF
C
C       Total number of events cannot excede 16.
C
      IF (NEVENT.GE.16) THEN
        GO TO 61
      ENDIF
 110  CONTINUE
 60   CONTINUE
C
C       First number in the EVENT array set to be the total number of
C       events recorded in this group.
C
 61   EVENT(1)=NEVENT
C
C       EVENT(195) set to mean of chirp frequency range.
C       This element is empty even if 16 events found.
C
      EVENT(195)=SDT
C
C       EVENT2 array recieved from task CORR1 which is automatically
C       appended to EVENT by virtue of the equivalence command at the
C       start of the task.
C
      CALL F77_CHAN_IN_MESSAGE (800,EVENT2,INCHAN)
      CALL F77_CHAN_OUT_MESSAGE (1600,EVENT,OUT)
C
C       If FLAG1 is set to 1 the histogram is outputed to the ROOT task.
C
      IF (FLAG1.EQ.1) THEN
        CALL F77_CHAN_OUT_MESSAGE (400,Y,OUT)
        FLAG1=0
      ENDIF
      GO TO 2000
```

368 END

```
      SUBROUTINE FOUR1(DATA,NN,ISIGN)
      DOUBLE PRECISION WR,WI,WPR,WPI,WTEMP,THETA
      DIMENSION DATA(*)
      N=2*NN
      J=1
      DO 11 I=1,N,2
        IF(J.GT.I)THEN
          TEMPR=DATA(J)
          TEMPI=DATA(J+1)
          DATA(J)=DATA(I)
          DATA(J+1)=DATA(I+1)
          DATA(I)=TEMPR
          DATA(I+1)=TEMPI
        ENDIF
        M=N/2
1       IF ((M.GE.2).AND.(J.GT.M)) THEN
          J=J-M
          M=M/2
        GO TO 1
        ENDIF
        J=J+M
11    CONTINUE
      MMAX=2
2     IF (N.GT.MMAX) THEN
        ISTEP=2*MMAX
        THETA=6.28318530717959D0/(ISIGN*MMAX)
        WPR=-2.D0*DSIN(0.5D0*THETA)**2
        WPI=DSIN(THETA)
        WR=1.D0
        WI=0.D0
        DO 13 M=1,MMAX,2
          DO 12 I=M,N,ISTEP
            J=I+MMAX
            TEMPR=SNGL(WR)*DATA(J)-SNGL(WI)*DATA(J+1)
            TEMPI=SNGL(WR)*DATA(J+1)+SNGL(WI)*DATA(J)
            DATA(J)=DATA(I)-TEMPR
            DATA(J+1)=DATA(I+1)-TEMPI
            DATA(I)=DATA(I)+TEMPR
            DATA(I+1)=DATA(I+1)+TEMPI
12        CONTINUE
          WTEMP=WR
          WR=WR*WPR-WI*WPI+WR
          WI=WI*WPR+WTEMP*WPI+WI
13      CONTINUE
        MMAX=ISTEP
      GO TO 2
      ENDIF
      RETURN
      END

      SUBROUTINE REALFT(DATA,N,ISIGN)
      DOUBLE PRECISION WR,WI,WPR,WPI,WTEMP,THETA
      DIMENSION DATA(32768)
      THETA=6.28318530717959D0/2.0D0/DBLE(N)
      C1=0.5
      IF (ISIGN.EQ.1) THEN
        C2=-0.5
        CALL FOUR1(DATA,N,+1)
      ELSE
        C2=0.5
        THETA=-THETA
      ENDIF
      WPR=-2.0D0*DSIN(0.5D0*THETA)**2
      WPI=DSIN(THETA)
      WR=1.0D0+WPR
      WI=WPI
      N2P3=2*N+3
      DO 11 I=2,N/2+1
        I1=2*I-1
        I2=I1+1
        I3=N2P3-I2
        I4=I3+1
        WRS=SNGL(WR)
        WIS=SNGL(WI)
        H1R=C1*(DATA(I1)+DATA(I3))
        H1I=C1*(DATA(I2)-DATA(I4))
        H2R=-C2*(DATA(I2)+DATA(I4))
        H2I=C2*(DATA(I1)-DATA(I3))
        DATA(I1)=H1R+WRS*H2R-WIS*H2I
        DATA(I2)=H1I+WRS*H2I+WIS*H2R
        DATA(I3)=H1R-WRS*H2R+WIS*H2I
        DATA(I4)=-H1I+WRS*H2I+WIS*H2R
        WTEMP=WR
        WR=WR*WPR-WI*WPI+WR
        WI=WI*WPR+WTEMP*WPI+WI
11    CONTINUE
      IF (ISIGN.EQ.1) THEN
        H1R=DATA(1)
        DATA(1)=H1R+DATA(2)
        DATA(2)=H1R-DATA(2)
      ELSE
        H1R=DATA(1)
        DATA(1)=C1*(H1R+DATA(2))
        DATA(2)=C1*(H1R-DATA(2))
        CALL FOUR1(DATA,N,-1)
      ENDIF
      RETURN
      END

      SUBROUTINE POLCOE(X,Y,N,COF)
      PARAMETER (NMAX=5)
      DIMENSION X(N),Y(N),COF(N),S(NMAX)
      DO 11 I=1,N
        S(I)=0.
        COF(I)=0.
11    CONTINUE
      S(N)=-X(1)
      DO 13 I=2,N
        DO 12 J=N+1-I,N-1
          S(J)=S(J)-X(I)*S(J+1)
12      CONTINUE
        S(N)=S(N)-X(I)
13    CONTINUE
      DO 16 J=1,N
        PHI=N
        DO 14 K=N-1,1,-1
          PHI=K*S(K+1)+X(J)*PHI
14      CONTINUE
        FF=Y(J)/PHI
        B=1.
        DO 15 K=N,1,-1
          COF(K)=COF(K)+B*FF
          B=S(K)+X(J)*B
15      CONTINUE
16    CONTINUE
      RETURN
      END
```