THE UNIVERSITY OF HULL

The Design and Development of a System for Automatic Sound Analysis

being a Thesis submitted for the Degree of

Doctor of Philosophy

in the University of Hull

by

Samantha Jane Barry, MChem

(December 2006)

ACKNOWLEDGEMENTS

This work has been funded by the Engineering and Physical Sciences Research Council (EPSRC) and by a CASE award from the Department of Academic Medicine, Castle Hill Hospital.

I would like to thank my supervisor, Dr Tony Walmsley, for his help, advice and guidance during my PhD. I would also like to thank my second supervisor, Prof. Alyn Morice, whose determination to win the cough-monitor race opened up this opportunity for me.

I would like to express my appreciation to all the people who have helped with this work. Thanks to Dr Ian Flynn for his help and advice on the DSP work. Thanks to Graham Naylor, Nick Mitchell and the rest of the Medical Physics Department, Hull Royal Infirmary for their help with developing the hardware.

Thanks to patients at the Cough Clinic, Castle Hill Hospital who have taken part in the cough studies and also to the staff who I have worked with along the way.

Thanks to the many people from the Department of Chemistry who have helped me during my time in Hull, both as an undergraduate and as a postgraduate.

Thanks also to my friends over the years who have made me laugh and kept me sane and took the time to listen; you know who you are.

Special thanks to Steve for all his help and encouragement during my PhD.

My greatest "thank you" of all goes to my parents. The list of reasons would be as long as the thesis, but ultimately for their unfaltering support and love throughout my many years in education, without which, this and many other achievements would not have been possible. To them I dedicate this thesis.

ABSTRACT

Cough monitoring has been undertaken for many years but the subsequent analysis of cough frequency and the temporal relation to trigger factors have proven problematic. Since cough is episodic, data collection over many hours is required; real-time aural analysis of the resultant data is therefore highly time-consuming. As cough as a symptom becomes a more important area of study, the requirement for a method for cough monitoring and assessment increases.

An objective, portable device has been developed for the 24 hour acoustic monitoring of cough together with a system for the automatic recognition and counting of coughs in audio recordings.

Digital signal processing was applied to extract the spectral features of sound events, followed by a variety of investigated data pre-treatment techniques.

Due to their success in other areas, artificial neural networks were studied in depth for their potential application as a pattern classification step; they were, however, found to be unsuitable for the discriminations between cough and non-cough events due to the high degree of variability present in both classes.

Digital filtering was then applied to remove a range of low frequencies common to many sounds in order to study the characteristics of the higher frequencies.

The system has been demonstrated to correctly classify 82% of all coughs present in audio recordings; the 18% of coughs that were missed were mostly very low in amplitude and would not have been classified as coughs by an experienced cough listener. This figure of 82% is therefore largely underestimated. Only 11% of the events classified coughs were actually non-coughs. Results for sensitivity are calculated to be 82%, specificity is 97% and accuracy is 94%. These are considered to be highly acceptable for clinical applications.

Ambulatory high-quality cough recordings are now possible over a 24 hour period. Automatic analysis affords rapid and reproducible association of cough with environmental triggers.

ABBREVIATIONS

AI	Artificial Intelligence
ANN	Artificial neural network
ATRAC	Adaptive transform acoustic coding
CDF	Cumulative density function
CF	Cystic fibrosis
DAT	Digital audio tape
DCT	Discrete cosine transform
DFT	Discrete Fourier transform
DoE	Design of experiment
DSP	Digital signal processing
ECG	Electrocardiograph
EMG	Electromyogram
FDA	Food and Drug Administration
FF	Feed-forward (neural network)
FFT	Fast Fourier transform
GORD	Gastro-oesophageal reflux disease
GUI	Graphical user interface
HDD	Hard disk drive
Hi-MD	High density MiniDisc
HMM	Hidden Markov model
HOSA	Higher order spectral analysis
LPC	Linear predictive coding
LVQ	Learning vector quantisation (neural network)
MD	MiniDisc
MFCC	Mel-frequency cepstral coefficients
PC	Principal component
PCA	Principal component analysis
PCM	Pulse code modulation
PNN	Probabilistic neural network
PSD	Power spectral density
RAM	Random access memory
RMS	Root mean square
SBE	Sector boundary error
SNR	Signal to noise ratio
SOM	Self organising map
SVD	Singular value decomposition
URTI	Upper respiratory tract infection
USB	Universal serial bus
WAV	Waveform

CONTENTS

CHAPTER 1

INTRODUCTION

1.1 AIMS

The aims of this work were to develop a system for the acoustic monitoring of cough in patients. Intended applications of the system included the study of the patterns of patients' cough episodes and their correlation to various trigger factors and also the comparison of cough frequency before and after medication in cough suppressant efficacy studies. The work comprised of two elements; the design and production of both a monitoring device and a method for the automatic analysis of the results.

The monitoring device had to be objective and capable of the ambulatory monitoring of cough for 24 hour patient studies. It also had to be capable of recording various patient activities to allow the study of trigger factors. Due to the time requirements of manually processing recordings of 24 hour duration, automatic analysis was necessary to make this a feasible application. A computerised method was therefore required to process the resultant audio recordings and thus achieve automatic cough recognition. The results of this process then need to be presented in a clear way for medical interpretation.

A method for the automatic recognition of cough had previously been designed; the aim was to develop this method for application to 24 hour ambulatory studies. A further aim was to investigate additional methods appropriate for the automatic recognition of cough.

1.2 COUGH

1.2.1 Introduction

Sound has been used for many years to monitor medical conditions. Devices for acoustic monitoring include the stethoscope invented in 1816 by Rene T.H. Laennec to monitor heart and lung sounds, monitors for snoring and sleep apnoea[1], infant monitoring systems to guard against sudden infant death syndrome (SIDS)[2] and cough monitoring which has implications in drug development for efficacy studies as well as for the monitoring of disease in patients.

Cough is an essential protective reflex which clears the respiratory tract and prevents the inhalation of foreign materials and excessive secretions into the respiratory system. Cough is one of the most common complaints for which patients seek medical advice[3] and population studies reported prevalence of cough to vary between 3% and 40%[4]. In contrast to the trivial complaints of cough caused by acute upper respiratory tract infections (URTIs), such as the common cold, chronic cough sufferers can experience severe debilitating effects. Fortunately, the perception of cough as a symptom is now changing. Progress has been made in defining the range of conditions responsible for persistent cough and scientific advances in understanding the biology of the cough reflex are expected to lead to improved therapeutic strategies.

Based on duration there are two categories of cough that are not mutually exclusive: acute and chronic.

1.2.1.1 Acute cough

Acute cough is defined as lasting for less than three weeks, although prolonged post-viral cough could cause symptoms as long as a few months later[5]. The vast majority of acute cough is caused by URTIs, in which cough is usually non-productive and self-limiting. The management of acute cough has massive health economic consequences with the use of over-the-counter cough remedies in the UK being estimated at 75 million unit sales per annum[4]. The main active ingredients are codeine, which has more recently been replaced with pholcodine due to its less toxic metabolites, and dextromethorphan. However, there is a disparity regarding the efficacy of many of these antitussives, the most common reason for which appears to be the different models employed to study the antitussive effects. Many studies have shown these agents to be effective in animal models[6-9] and human induced-cough models[10-12] although there are few data to show the effectiveness of treating acute cough due to URTI in humans. Some studies have even excluded the use of placebos during antitussive trials, thus failing to show any significant improvements over the placebo effect. It is also questionable that any improvement seen in patients with URTI is due to the antitussive effect rather than the natural progression of the

infection. Furthermore, a recent study of over-the-counter cough medicines has suggested they possess little clinically relevant efficacy[13].

1.2.1.2 Chronic cough

Persistent cough lasting eight weeks or more is classed as chronic. Chronic cough is significantly associated with adverse physical and psychosocial effects on quality of life[14]. Debilitating physical effects can include exhaustion, incontinence, loss of sleep, cough syncope, vomiting, hoarseness and even broken ribs; whilst psychosocial effects can include embarrassment, self-consciousness, absence from work, inability to socially interact and changes in lifestyle and activities.

Irwin *et al.* reported that a cause for chronic cough could be found in 98% of cases and was due to either cough variant asthma (CVA), rhinisinusitis associated with postnasal drip syndrome (PNDS) or gastro-oesophageal reflux disease (GORD)[5]. In addition, a common cause of cough in sufferers of high blood pressure is ACE-inhibitor cough, directly caused by the medication, which significantly lowers the cough threshold[3, 15].

1.2.1.2.1 Cough variant asthma (CVA)

Cough variant asthma is a type of asthma in which the sole or most predominant symptom is cough[16]. Patients typically exhibit asthmatic responses to hyperresponsiveness but not bronchoconstriction and respond well to anti-asthma treatment. Other symptoms of asthma such as wheezing and dyspnoea are not present. Treatment of CVA is achieved with anti-inflammatory therapy such as inhaled corticosteroids and short-acting bronchodilators as needed.

1.2.1.2.2 Postnasal drip syndrome (PNDS)

Postnasal drip syndrome is a symptom, not a disease, the causes of which can include allergic rhinitis, vasomotor rhinitis, viral or bacterial infections and nasal polyps[17]. The cough reflex is stimulated by the drainage of fluid from the nose or sinuses down the back of the throat. Treatment depends on the cause of the excessive secretions and can include antihistamines, decongestants or antibiotics.

1.2.1.2.3 Gastro-oesophageal reflux disease (GORD)

Gastro-oesophageal reflux disease (GORD) is defined as occurring when typical reflux of substances from the stomach into the oesophagus leads to symptoms or physical complications. The cough reflex is stimulated by the reflux into the oesophagus and subsequent aspiration. In up to 75% of cases, cough is the sole presenting symptom of GORD which can pose problems with diagnosis[3]. Treatment is usually by stomach acid suppression using drugs such as proton pump inhibitors or prokinetic agents or by anti-reflux surgery[3].

1.2.2 Cough studies

The objective assessment of cough frequency is an important measure for both clinical and research purposes. Cough is a common symptom of many respiratory disorders and when monitored, can provide important information for diagnosis, therapeutic efficacy and progression of disease. In addition, the assessment of cough is important in pharmacological studies for determining the efficacy of antitussive drugs.

Interestingly, the actual definition of a cough remains an unresolved issue in the medical world. Some argue that every cough event must have its own preceding inspiration to be counted as one cough, while others argue that the number of cough events should be defined by the number of characteristic cough sounds made. Equally, some suggest that cough severity can be measured based on the time spent coughing, others believe that there needs to be a measure of cough intensity included. It is therefore widely accepted that the cough counter can make a personal definition of cough as long as it is consistent and reasoned.

The evaluation of cough severity has so far relied mainly on subjective measures such as cough reflex sensitivity and on the patient's perception of the symptom assessed by cough visual analogue scores, quality of life questionnaires, cough symptom scores and patient's diaries[18-21]. However, subjective recording or scoring of cough is unreliable as individual perception of cough differs from mild irritation to marked impairment of quality of life[14, 19]. In addition, subjective assessment of cough frequency during the night-time has been shown to be unreliable[22, 23]. The measure of

cough reflex sensitivity by cough-inducing agents such as citric acid[24], capsaicin[25] and water aerosols[26] has long been carried out. However, while induced-cough models serve as effective screening tools for antitussive agents and for studying the cough reflex, there are few data to relate treatment responses using this model with those in disease. Induced cough is therefore not yet considered a definitive clinical model to demonstrate the efficacy of antitussive drugs. Further to this, Sevelius noted that patients could become tolerant of the inhaled irritants used to induce cough[27]. Woolf showed that a drug which had previously been proven effective using induced cough tests to be ineffective when a cough count was performed[28]; thus indicating that induced cough responds differently to suppressants and that cough monitoring is a more representative method.

Successful cough frequency monitoring therefore depends on an objective cough counting method. The basic requirements for such a monitor are the possibility to record over a representative amount of time, usually 24 hours to study the diurnal changes of cough, and the use of a portable recording system to study patients in their own environment carrying out their usual daily activities. In addition, since the manual examination of cough monitoring recordings is a slow and tedious process, the capacity to automatically detect the occurrences of cough sounds from the recordings is required.

1.2.2.1 History of objective cough counting

Several attempts at objective cough counting have been made over the years. In 1954, Gravenstein *et al.* recognised the problem of finding quantitative data to show that antitussive agents reduce the frequency of pathological cough[29]. Previous studies by Keats *et al.* had shown placebos to be an effective therapy for ailments where subjective factors play a major role[30]. To solve this, Gravenstein used the Grass sound recorder developed by Albert Grass, founder of the Grass Instrument Company, to test the efficacy of antitussives on pathological cough. Cough was recorded by use of an ink-writer on a moving paper tape in the form of spikes of varying frequency and amplitude, depending on the frequency of the cough. In 1964, Woolf and Rosenberg developed a method to record patients using a free-field microphone and tape

recorder[28]. The tape recorder was activated by an electronic on/off unit which was triggered by a sound signal; this meant that up to 24 hours of observations could be made on one tape and manual counting could be achieved in approximately two hours. A time signal was generated every half hour to allow more accurate and temporal cough frequency counting. In 1966, Sevelius and Colmore developed a very similar set up, this time using a throat microphone to exclude background noise[27]. A limited amount of movement by the patient was enabled by use of a long cord. In 1967, a more complex system for cough frequency monitoring was developed by Loudon[31, 32]. Again, the system used a free-field microphone which detected all sounds in a patient's room and fed the signal through to a sound-activated tape recorder. However, the signal was then amplified and passed through a frequency-specific attenuator to an amplitude discriminator with an adjustable threshold. The frequency attenuator and amplitude discriminator were tuned to accept coughs and reject all other sounds, thus reducing the required counting time even further.

In 1974, Thomas *et al.* developed a semi-ambulatory system in which patients were "free-roaming" within a controlled environment[33]. A microphone was attached to the patient's throat and sound was sent, via a transmitter, to a tape recorder. Cough was differentiated from other noise and subsequently counted by satisfying two criteria: firstly possessing a rapid rate of rise that is characteristic of coughs; and secondly possessing an energy content above a determined threshold.

In 1983, Matthys *et al.* measured cough frequency and intensity during sleeping hours by use of a pressure transducer attached over the trachea and connected to a recorder which measured the amplitude of the coughing[34].

In 1988, Salmi *et al.* developed a method for long term recording and automatic analysis based on the simultaneous recording of filtered acoustic signals and cough-induced fast movements of the body[35]. A dynamic microphone mounted in the focus of a paraboloid acoustic fibreglass mirror was placed at the foot of the bed and directed toward the face of the patient. Signals from the microphone were high-pass filtered to eliminate low-frequency noise. Body movements were recorded with a static charge-sensitive bed connected to an amplifier and a filter unit. High-pass filtering was used to record only fast body movements typical for cough. Events were

thus identified as coughs if the signals in both cases simultaneously exceeded the detection threshold. While the method gave excellent levels of sensitivity (99%) and specificity (98%), it was limited by the fact that the patient had to remain in bed, could only engage in quiet conversation and had to refrain from making any quick movements whilst speaking.

In 1997, Chang *et al.* adapted a disused Holter monitor, a device for ambulatory electrocardiography, to create an inexpensive cough monitor[19, 36]. The device used the electromyogram (EMG) signal alongside an audio signal and both were filtered before recording to minimise unwanted signals. A cough was defined as a positive response occurring simultaneously in both channels. The device gave good results when validated against an ordinary tape recorder, although the tests were carried out overnight whilst the subjects were asleep, thus encountering very limited background noise. In addition, it was necessary for the cough counting to be carried out manually by a trained investigator, leading to a time-consuming process.

While all these systems were addressing the need for objective cough monitoring and even reducing the processing time of the data by sound activated tape recordings, there remained the problem that patients could not be fully mobile under study. Eccles showed that simple resting causes a reduction in cough frequency[37], thus further illustrating the need for ambulatory cough counters.

In 1994, Hsu *et al.* developed the first 24 hour ambulatory cough monitor[38]. The device made simultaneous recordings of cough sounds and of EMGs of the lower respiratory muscles, including diaphragmatic activity. Cough counting was performed manually by visualising the audio and EMG signals. Later the same year, Munyard *et al.* reported a similar device with the addition of electrocardiograph (ECG) monitoring and an accelerometer to determine the level of the subject's activity[39].

In 1996, Pavesi *et al.* developed a computerised cough acquisition system to test the efficacy of dextromethorphan on acute cough caused by URTI[40, 41]. Developments were carried out to allow a degree of portability, such that patients were allowed to move around within a confined area, such as their home. The device used a contact microphone attached to the patient's suprasternal notch and detected cough audio vibration signals. These signals were then transmitted by frequency modulation to a

receiver which was positioned within the area of study. Analysis of the data gave information about cough frequency, intensity and latency. However, the analysis procedure still required manual listening and input from a trained investigator.

In a similar way in 1998, Rietveld *et al.* developed a system to record tracheal sounds of children in their homes[42]. The ambulatory device consisted of an electret microphone and a transmitter which fed to a receiver and recorder placed in the patient's home. The study covered two recording durations, 24 hours and 72 hours and analysis was carried out manually. A conclusion of the report was that the time consumption made the method unattractive for clinical application.

1.2.2.1.1 System assessment

Cough recognition systems are generally assessed in terms of sensitivity, specificity and accuracy; these are defined in Equations 1-1, 1-2 and 1-3 respectively, where TP is true positive, TN is true negative, FP is false positive and FN is false negative.

$$\text{Sensitivity} \quad \frac{TP}{TP + FN} \qquad \textit{Equation 1-1}$$

$$\text{Specificity} \quad \frac{TN}{TN + FP} \qquad \textit{Equation 1-2}$$

$$\text{Accuracy} \quad \frac{TP + TN}{TP + TN + FP + FN} \qquad \textit{Equation 1-3}$$

1.2.2.2 State of the art

Only very recently has the technology evolved such that it is possible to incorporate all the desired features into an automatic cough monitor; portability, extended recording durations and automatic analysis. Automatic cough recognition from ambulatory multi-channel physiological recordings has been reported by Coyle *et al.* in 2005[43]. The system is a cardio-respiratory monitoring device called a Lifeshirt™ (Vivometrics Inc., Ventura, California) which, like the devices by Chang *et al.*, Salmi *et al.* and Hsu *et al.*, relies upon multiple signal monitoring for cough recognition. Respiratory inductance plethysmography (RIP) non-invasively measures ventilation, a unidirectional contact microphone monitors cough sounds, an ECG measures heart

rate and a centrally located 3-axis accelerometer monitors posture. Automatic analysis uses data from all channels to identify a cough event. Accuracy of the Lifeshirt™ system is given at 99%, sensitivity was lower at a value of 78.1% whilst specificity was 99.6%. Of a total of 3,645 coughs counted using video surveillance, 3,363 coughs (92%) were counted. However, while the results appear to show the Lifeshirt™ as unequalled, there are some limitations associated with the device and its use that are not defined. The authors note that a limitation of the video surveillance was the human error introduced to counting due to fatigue, suggesting that the percentage of coughs that the system counts could be lower. The validation method of video surveillance unfortunately limits patient mobility and fails to illustrate how well the system would cope in a fully ambulatory situation with patients undergoing their usual daily activities. The system is incorporated into a vest which must be worn underneath the clothing for the entire study duration once it has been fitted, including during sleep, which introduces impracticalities of the patient being unable to remove the device for bathing. The whole device is expensive with the purchase of one Lifeshirt™ plus software costing up to $30,000 (US). Personal communication with L.P.A. McGarvey (The Queen's University of Belfast) and J.A. Smith (Wythenshawe Hospital, Manchester) revealed that investigators that have used the Lifeshirt™ find its sensitivity and specificity to be much less than claimed in the published paper. Equally, the authors have admitted that the system identifies throat clearings as coughs; thus potentially introducing a degree of inaccuracy into the results. A further disadvantage is that data must be analysed by Vivometrics, who then compile the results; this "black-box" approach is too ambiguous for medical applications and regulations.

In 2002, Hiew *et al.* developed an automatic approach to cough monitoring of asthma patient recordings[44]. The system consists of a digital recording device and a computer algorithm. To compress the sound file, any silent periods are firstly eliminated by way of an applied threshold level. The remaining recording is then divided into one second portions for further analysis. Each one second envelope is checked for exceeding a second threshold and if it does, auto-correlation is applied to determine whether the sound fulfils the criteria of being an asthma cough. The final check is to see whether the significant frequency is within a set range. If all of these criteria are

met, the one second envelope is classed as a cough second. Values for sensitivity and specificity were quoted at 70.5% and 98.3% respectively. However, the study used overnight patient recordings which immediately limits the amount of background noise encountered and therefore does not truly test the performance of the system. Equally, it is not possible to demonstrate the performance of the system in instances of fully ambulatory patients. In 2006, part of the same group, Smith *et al.* used the compression stage of the algorithm to exclude any periods of silence and then manually counted coughs in the remainder of the recording[45], perhaps indicating the weaknesses of the original system. This semi-automatic approach, very similar to that used by Woolf *et al.* in 1964, took processing time of a 10 hour recording down to 2.5 to 3 hours. Also in 2006, Matos *et al.* developed a system which consists of a portable digital audio recorder to record cough in ambulatory patients together with an algorithm based on a Hidden Markov Model (HMM) to carry out automatic analysis[46]. The system correctly recognised cough at a detection rate of 71% with a false positive rate of 13 events/hr. As expected, false positives included speech (at an average of 4 events/hr), throat clearing sounds, impulsive noises and cough sounds from other persons. To reduce the number of false positives being counted, two energy thresholds were introduced at 90% (CDF90) and 95% (CDF95) of the recording's energy cumulative density function (CDF). However, this threshold immediately excluded a percentage of the cough sounds from analysis, with only averages of 84% (range 48% to 100%) still being present at CDF90 and dropping to 72% (range 28% to 94%) at CDF95. Of these two percentages, only averages of 77% and 82% respectively were actually recognised by the algorithm leading to an overall average result of 65% at CDF90 and 59% at CDF95 with the false positive rate at 9 events/hr and 7 events/hr respectively. These actual figures show that the system does not perform as well as suggested by the authors. Later the same year, Paul *et al.* developed an ambulatory, objective cough monitor, which lacked automated analysis[47]. The system was based on an accelerometer which measured vibrations in the chest and was thus subject to interference from other upper respiratory sounds such as speech, laughter and snoring. However, it was not reported whether or not these sounds caused the occurrence of false positives. In addition, the system had only been tested on short durations, most being only 30 minutes, due to the time

consuming nature of validating the method against video monitoring. No 24-hour studies were carried out to test the possibility of such durations or the time required to process them. Also in 2006, Barry *et al.* developed an automatic method for the analysis of cough sounds in audio recordings but without the ability to make 24 hour recordings[48]. The system used a digital audio tape (DAT) recorder to monitor cough sounds in smoking patients. The automatic system then isolated periods of sound, calculated the cepstral coefficients and used a probabilistic neural network (PNN) for the classification of cough and non-cough sounds.

Although progress is definitely being made, there still remains to be developed a fully ambulatory monitoring system capable of making objective 24 hour cough studies with high figures of sensitivity, specificity and accuracy.

1.3 Analysis of Cough

In the attempt to develop an objective and automatic cough counter, the acoustic properties of cough must be studied in order to determine specific features which could be used to distinguish cough from any other sounds.

1.3.1 Studies of cough sounds

Cough is easily recognised by the human ear and can be audibly distinguished from other upper airway sounds such as speech, laughter, throat clearing and snoring.

The cough reflex can be described as a sudden rapid air expulsion, caused by a coordinated contraction of the respiratory musculature against a closed glottis which then opens[49, 50]. Cough sounds can generally be characterised as a composite of two or three phases[51] [See Figure 1.1]. The first is an expulsive phase occurring at the moment of glottal opening and causing a high intensity sound. This is followed by an intermediate or steady state phase which is lower in amplitude and associated with a steady flow of air through the open glottis. Frequently, there is also a third voiced phase which is caused by the partial closing of the vocal chords vibrating toward the end of the cough[49], though this third phase is not always present.

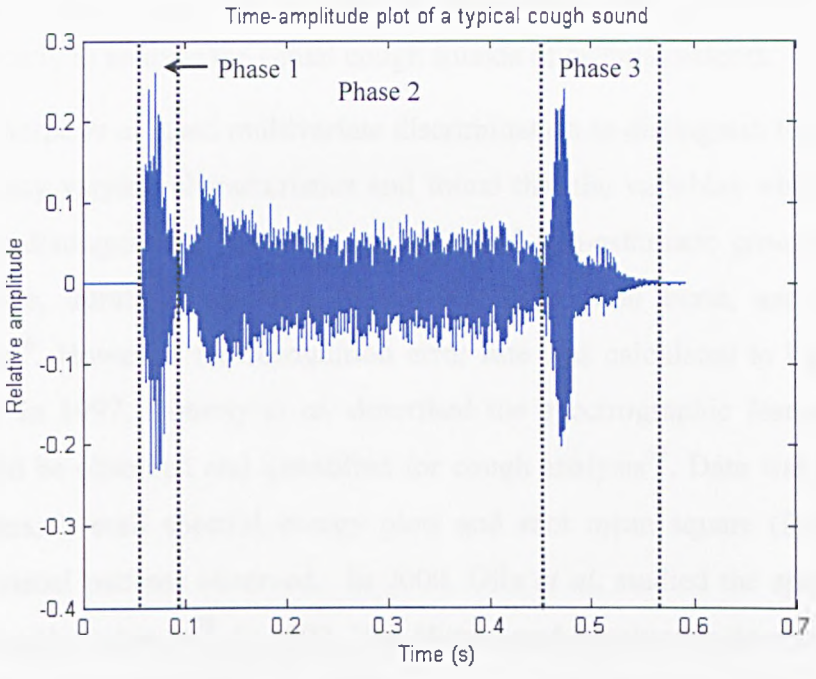


Figure 1.1 Typical three-phase cough sound

Cough can be separated into four common cough-classes, respectively labelled as voluntary, chemical-induced, acute and chronic[49], where acute and chronic are both spontaneous cough. However, it has also been demonstrated that cough sounds can vary widely and that their acoustical differences, caused by gender or the presence of mucus for example, can be detected by the human ear[52]. The large number of descriptors used for cough sounds, including brassy, rattling, bovine and barking, perhaps also indicate the potential difficulties in classifying all these sounds as one type.

Several studies have been made into the classification of cough types based on their acoustical features. In 1989, Piirilä and Sovijärvi studied the acoustics of spontaneous cough in patients with various pulmonary diseases using spectrograms[53]. They found that sound characteristics and flow dynamics can be typical of certain diseases and for some parameters there was no overlap between patient groups. In 1991, Toop *et al.* designed a portable but not ambulatory system for the spectral analysis of cough sounds in asthma[54, 55]. The hardware comprised of a flow meter, a contact microphone for chest sounds, a free-field microphone for cough sounds, an amplifier, filter and replay unit and a computer; all of which was fitted onto a trolley for

portability. The method was not one for monitoring cough frequency over long periods, merely to analyse the actual cough sounds of asthma patients.

In 1992, Thorpe *et al.* used multivariate discrimination to distinguish the cough types by their many varying characteristics and found that the variables which were most effective at distinguishing between asthmatic and non-asthmatic groups were: zero-crossing rate, duration, presence or absence of a final burst, and the cepstral coefficients[56]. However, the recognition error rate was calculated to be between 20 and 30 %. In 1997, Doherty *et al.* described the spectrographic features of cough which could be observed and quantified for cough analysis[57]. Data was presented as spectrograms, overall spectral energy plots and root mean square (RMS) pressure plots and visual patterns observed. In 2000, Olia *et al.* studied the spectrograms of cough in healthy subjects[58]. In 2002, Van Hirtum and Berckmans developed a system to discriminate between voluntary and spontaneous cough types to an accuracy of 96%[59]. Recognition was achieved using various methods of digital signal processing and a learning vector quantisation (LVQ) neural network.

In 2006, Murata *et al.* described a system for automatic cough counting based on six types of classified cough sounds[60]. In this, five parameters of cough sounds were identified which could be used to characterise the coughs. From this, coughs were classed into six types and a system was designed to recognise any sounds that matched any of these categories. The resulting system achieved a sensitivity of 90.2% and a specificity of 96.5%. The study did however necessitate recordings in a silent room during sleeping hours to avoid any interference from background noise. The group had also previously demonstrated the ability to discriminate between productive and non-productive cough sounds by spectral analysis[61]. In 2002, Korpas *et al.* studied the variation in cough sound patterns during disease using time-amplitude plots or tussinophonograms[49]. They concluded that tussinophonograms possess sufficient variation between diseases to be useful in diagnosis [See Figure 1.2]

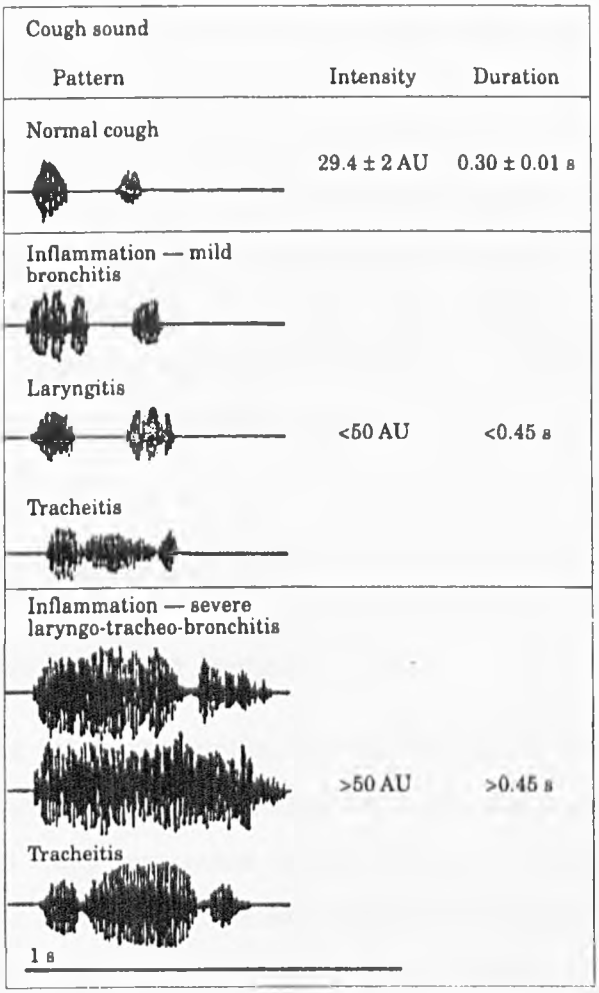


Figure 1.2 A summary of the variety in cough sound patterns (illustrated by time-amplitude plots) intensities and durations for different diseases. [Reproduced from Korpas et al.[49]]

1.3.2 Digital signal processing

Digital signal processing (DSP) is the use of mathematics, algorithms and techniques to analyse, modify or extract information from digital signals. Signals are digitised by taking uniformly spaced samples from the continuous signal, which are measured over a finite time. DSP has applications in many areas including communications, medical imaging, radar and sonar and high fidelity music reproduction.

Although there has been some attempt to understand the mechanism of cough production, so far little research has been directed toward computer modelling of the cough sound. As cough can be recognised by the human ear as being different to all

other sounds, it must possess unique acoustic properties that define it as an individual type of sound. Upper airway sounds originate from the resonance of air in the mouth, pharynx and nasal cavities. This resonance can be caused by the flow of air through the larynx giving rise to vocal sounds or by the expiratory force of air against a closed glottis which then opens giving rise to a cough sound. Since these are two similar processes, both dependent upon the resonance of air in the upper airway cavity, and since recognition of speech has already been extensively researched, techniques used to analyse speech provide an insight into possible methods available for cough sound analysis.

1.3.2.1 DSP for speech analysis

1.3.2.1.1 History of speech analysis

Research in automatic speech recognition has been ongoing for over 50 years with the earliest systems for automatic speech recognition being produced in the 1950s. Since then there have been many approaches to the technique of speech recognition in an attempt to produce an accurate and robust machine, intelligent enough to understand spoken conversation on any subject by all speakers and in all environments. This ideal is still a long way from being achieved. At present, although speech recognition systems are gaining increased accuracy and can generally cope with normal speech patterns, as opposed to the previous necessity to pronounce every word clearly and slowly, they still cannot adapt to multiple speakers and different environments.

By the time the first speech recognition systems were being developed, much other work had been carried out into speech and language technology. Perhaps one of the first and most important steps was the invention of the telephone in 1876 by Alexander Graham Bell and was later followed by the first long distance transmissions of speech over the radio in 1915. By 1898, Pouslon had invented the first tape recorder allowing speech to be recorded onto steel piano wires. Commercially viable tape recorders, using magnetic plastic tape as designed by O'Neill in 1927 became available in the 1930's.

Electronic speech synthesis began development in the 1920's with simple electrical resonators simulating sustained vowels. In 1939 Homer Dudley invented the

"vocoder", the voice operated recorder, a complete electronic speech synthesis system which produced speech from data extracted from natural speech.

Attempts at actual automatic speech recognition began in 1952 at Bell Laboratories with a system for isolated spoken digit recognition for a single speaker[62, 63]. The system relied on measuring spectral resonances during the vowel region of each word. In 1956, at RCA laboratories, Olson and Belar developed a system to recognise ten monosyllabic words as spoken by a single speaker[64]. This system used an analogue filter bank to take spectral measurements during the vowel regions of the words. These early systems required the user to use a distinct speech pattern to dictate by separating every word with a pause; known as discrete speech and stop dictation and thus ruling out the possibility of application to normal, continuous speech. The repeatability and reliability of the systems were also poor. The 1960s saw many other researchers enter the field and by 1966, Reddy had developed a continuous speech recognition system by dynamic tracking of phonemes[62]. The research program set up by Reddy shortly after, remains to this day a world-leader in continuous speech recognition. In the 1970s, speech recognition research achieved a number of significant milestones. Work by three separate groups allowed isolated word or discrete utterance recognition to become a practicable technology. Velichko and Zagoruyka in Russia advanced the use of pattern recognition ideas for speech recogntion[65], Sakoe and Chiba in Japan successfully applied dynamic programming methods[66] and Itakuro in the United States developed the use of linear predictive coding (LPC)[67]. Further to this, IBM set up a highly successful group research effort in large vocabulary speech recognition[62], whilst Bell Laboratories began research into making speaker independent speech recognition systems. By the 1980s, focus had moved onto connected word recognition in order to develop the technology for the recognition of fluent speech. The 1980s also saw the introduction of the auspicious Hidden Markov Model (HMM) and the re-introduction of the neural networks. Neural networks had originally been applied in the 1950s without success, however, new understanding of the technology brought with it a new interest. A major impetus to the development of speech recognition technology was given by the Defense Advanced Research Projects Agency (DARPA), which sponsored a large research program producing several successful results. Development of these original ideas

continued into the 1990s. Today, speech recognition is increasingly used, with commercial applications in telephone networks to automate operator services and also in dictation software.

1.3.2.1.2 State of the art

The attempt to perform speech recognition by machine has lead to three different approaches to achieve the task.

1. The acoustic-phonetic approach
2. The statistical pattern recognition approach
3. The artificial intelligence approach

1.3.2.1.2.1 Acoustic-phonetic approach

In 1967, Hemdal and Hughes put forward the acoustic-phonetic theory[68]. This suggests that there are finite, distinctive phonetic units, known as phonemes, in spoken language and that these units are characterised by a set of acoustic properties that are prominent in the speech signal over time. Once these characteristics have been spectrally analysed and represented, they must be converted to a set of features that describe the broad acoustic properties of different phonetic units. The final step in the approach is the segmentation and labelling of the regions in the speech according to how well the features within that region match those of individual phonetic units. There are several disadvantages to this method which mean the acoustic-phonetic approach is still in a development stage and has not been used successfully in actual speech recognition problems. The method requires extensive knowledge of the acoustic properties of phonetic units, which is generally only available for the simplest of situations such as steady vowels. The choice of features is made on considerations specific to the situation and for most systems is actually based on intuition rather than an optimising approach. The design of the sound classifiers is not optimal and the method of the speech labelling step is doubted by many linguistic experts.

1.3.2.1.2.2 Statistical pattern-recognition approach

The statistical pattern-recognition approach[68] consists of four steps; firstly, spectral analysis is performed on the input signal in order to carry out feature measurement and thus define a "test pattern". Next, a pattern representative of the features of test patterns corresponding to speech sounds of the same class is created to serve as a reference pattern. Unknown test patterns are then compared with class reference patterns and a measure of similarity between the two patterns is computed. Finally, the similarity scores are used to determine the most fitting reference pattern to the unknown test pattern in order to determine the speech input. In this approach, the performance of the system is sensitive to the amount of training the system receives. No speech-specific knowledge is required by the system so the method is relatively insensitive to the vocabulary used. It is also straightforward to incorporate syntactic (grammatically correct) and semantic (meaningful) constraints into the pattern recognition structure thereby improving the accuracy of the system.

Along with the development of techniques such as the statistical pattern recognition approach was the requirement for spectral analysis and feature detection. DSP became more widely used as the speech recognition methods became more sophisticated. Techniques employed by the statistical pattern-recognition approach will be discussed further in Section 1.3.3.2.

1.3.2.1.2.3 Artificial intelligence approach

The third, and potentially the most successful approach to developing automatic speech recognition by machine is the artificial intelligence (AI) approach.

This technique uses knowledge from a wide variety of sources and applies it to the problem faced. Thus, the system may use a mixture of acoustic, semantic and syntactic knowledge to apply to speech recognition. The necessity for such a range of knowledge requires a method of automatic knowledge acquisition or learning and an ability to adapt to the knowledge. An ideal approach which can be applied to this is the use of neural networks; these will be discussed further in Section 1.4.

1.3.3 DSP methods

1.3.3.1 Fourier transform

Fourier analysis describes a set of mathematical techniques based on decomposing signals into sinusoids. The Fourier transform, named after the French mathematician and physicist Joseph Fourier (1768-1830), is a mathematical technique for the reversible integral transform of one function into another. The resulting output gives the coefficients of sinusoidal basis functions against their frequencies whose linear combination produces the original function. The recombination of sinusoidal basis functions is called an inverse Fourier transform. For audio signals therefore, the Fourier transform defines a relationship between a signal in the time domain and its representation in the frequency domain. In this way, the frequency content of an audio signal can be calculated and used for spectral analysis. The actual mathematical transformation is called the discrete Fourier transform (DFT) while the efficient algorithm for computing it is known as the fast Fourier transform (FFT).

1.3.3.1.1 Discrete Fourier transform (DFT)

The DFT calculates the Fourier transform of a signal at discrete time intervals. The DFT provides a means for analyzing the frequency components of a discrete-time signal. This can be achieved by fitting the signal with all allowed sine and cosine functions for each considered frequency. This constitutes a laborious process; a discrete signal of 1024 data points requires the calculation of 1024 parameters by linear regression and the calculation of the inverse of a 1024 by 1024 matrix. The DFT is defined in Equation 1-4 where $x_0,\ldots,x_{N-1}$ are complex numbers.

$$X_k = \sum_{n=0}^{N-1} x_n e^{-\frac{2\pi}{N} nk} \qquad k = 0,\ldots,N-1$$

Equation 1-4

1.3.3.1.2 Fast Fourier transform (FFT)

German mathematician Karl Friedrich Gauss (1777-1855) developed the principles of FFT in 1805 to interpolate asteroidal trajectories. However, his work was not widely

known, only being published posthumously and in neo-Latin. Various other limited forms were rediscovered through the 19th and 20th centuries, but it wasn't until 1965 when Cooley and Tukey reinvented the algorithm and described how to perform it conveniently on a computer[69] that FFT became popular. The number of operations in FFT is proportional to $(2N+1)\log_2(2N+1)$ instead of $(2N+1)$ as used in DFT, which reduces the number of operations required for 1024 data points from 10^6 to 10^4 and increases the speed by 100 times.

1.3.3.2 Spectral analysis models

The terminology surrounding the computation of the cepstrum came from the original article by Bogert et al[70], in which various terms from signal processing (spectrum, frequency, phase, analysis) were rearranged into anagrams (cepstrum, quefrency, saphe, alanysis). This was done to highlight the unusual treatment of frequency domain data, as if it were time domain data, in generating a new data set which had the quefrencies in units of seconds across its x-axis values, but which indicated variations in the frequency spectrum. Today, only the term "cepstrum" remains in common use.

1.3.3.2.1 Bank-of-filters model

The bank-of-filters model takes the magnitude frequency spectrum as calculated by a Fourier transformation. The spectrum is then smoothed with a collection of band-pass filters that span the relevant frequency region. Several designs for the arrangement of the band-pass filters exist including a uniform filter bank in which the filters are uniformly spaced over the frequency range under analysis. An alternative to this method is the non-uniform filter banks which are designed according to some criterion for the frequency spacing of the individual filters. This has the effect of concentrating data values in the more significant parts of the spectrum. Since the human perception of sound is not uniform or linear over the full range of frequencies, a common filter bank to use for speech recognition is the mel-scale based filter bank[71]. The mel-scale is based on the sound perception of the human ear and is designed to capture the phonetically important characteristics of speech. The mel-

frequency scale has linearly spaced filters at low frequencies i.e. below 1000 Hz and a logarithmic spacing at high frequencies i.e. above 1000 Hz (see Figure 1.3).

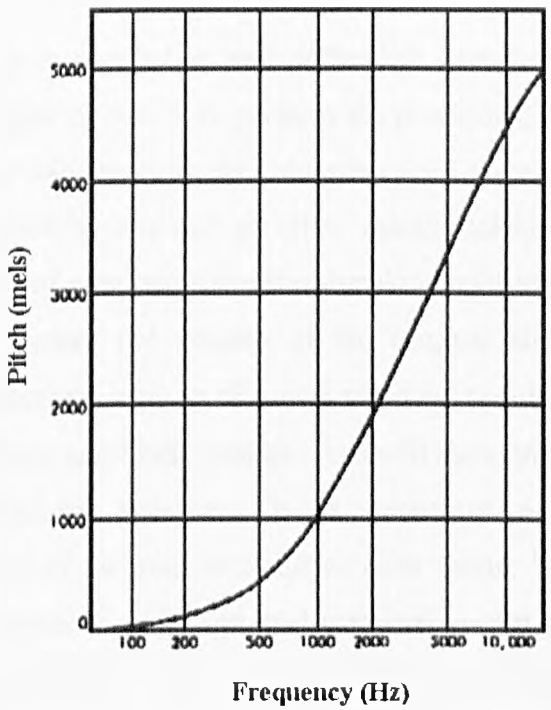


Figure 1.3 Illustration of the mel-scale [Reproduced from Fundamentals of Speech Recognition[62]]

The processing technique which uses the mel-scale filter bank is known as the Mel-Frequency Cepstral Coefficients (MFCC) method. A diagram of the structure of a MFCC processor is given in Figure 1.4.

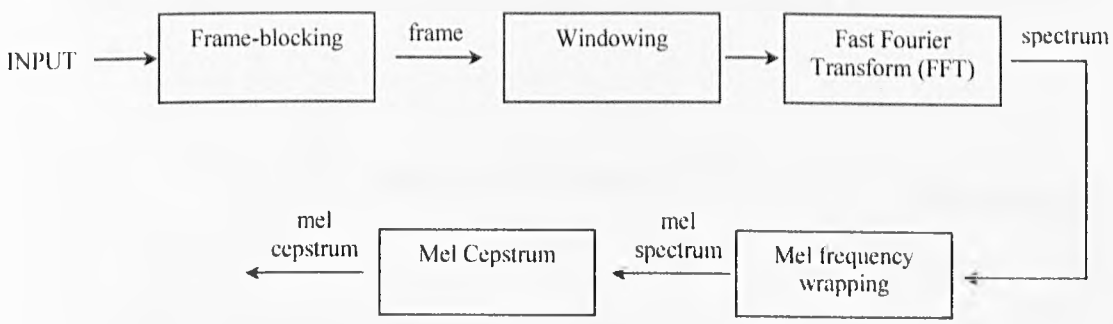


Figure 1.4 Block diagram of the MFCC processor

The MFCC processor passes the input signal through several stages:-

1. In frame-blocking, the signal is divided into overlapping frames which are then to be processed sequentially.

2. The next step is to window each individual frame so as to isolate a specific section of signal on which to perform the processing. The simplest window is a rectangular window with all values between the starting point and the end point multiplied by one and all other values multiplied by zero. A problem with this type of window is that the signal changes abruptly at each end of the window, a feature not present in the original signal. When the Fourier transform spectrum is taken of such a window, oscillations appear as artefacts due to the sharp amplitude change. To avoid such problems with the resulting Fourier transform spectrum, it is necessary to minimise the signal discontinuities at the start and end of each frame. This can be achieved by applying a window which gradually tapers toward zero at each end. Such windows include the Hanning window which is given in Equation 1-5.

$$w(n) = \frac{1}{2}\left(1 - \cos\left(\frac{2\pi n}{N}\right)\right) \qquad \textit{Equation 1-5}$$

Where N represents the width, in samples, of a discrete time window function and n is an integer with values $0 \leq n \leq N-1$.

More commonly used is the Hamming window which is given in Equation 1-6.

$$w(n) = 0.54 - 0.46\cos\left(\frac{2\pi n}{N-1}\right) \qquad \textit{Equation 1-6}$$

An example of the use of the tapered window can be seen in Figure 1.5.

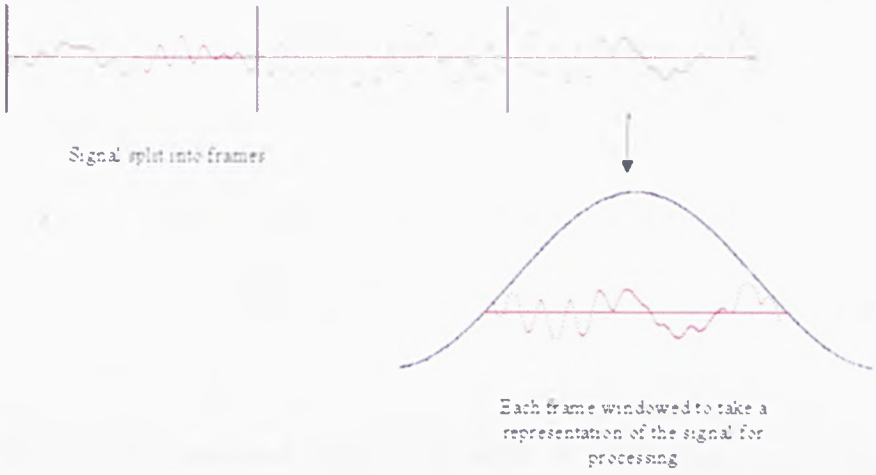


Figure 1.5 Illustration of the frame blocking and windowing shown with a tapered window.

3. The FFT converts each window of samples from the time domain into the frequency domain to produce a FFT spectrum.
4. The mel-frequency wrapping is the stage where the signal is passed through the band-pass filter bank according to the mel-frequency scale. The resulting output is therefore modified to emphasise the perceptually important lower frequencies while attenuating the higher frequencies.
5. Finally, the mel spectrum is converted back into the time domain by use of the Discrete Cosine Transformation (DCT) algorithm. The resulting output is the mel-frequency cepstral coefficients (MFCCs) and provides a representation of the spectral properties of the signal in the given frame.

1.3.3.2.2 Linear predictive coding (LPC) model

The idea of applying LPC to speech recognition problems was introduced in the 1970's and has been successfully used ever since. LPC is a method of digitally encoding analogue signals. The basic idea behind LPC is that a sample of sound $x[n]$ can be approximated as a linear combination of the past p sound samples. The linearly predicted signal is a smoothed version of the original signal.

The method uses a single or multi-level sampling system in which the value of the signal at each sample time is predicted to be a linear function of the past values of the quantised signal. Mathematically, LPC can be described as in Equation 1-7.

$$\tilde{x}[n] = a_1 x[n-1] + a_2 x[n-2] + \cdots + a_p x[n-p]$$
$$= \sum_{k=1}^{p} a_k x[n-k] \qquad \textit{Equation 1-7}$$

Where $\tilde{x}[n]$ is the predicted sample at point n and $a_1, a_2, \ldots, a_p$ are the LPC coefficients.

LPC provides a good model of the speech signal and even in the unvoiced regions of speech, which the model is less effective at coding, the results are still at an acceptable level. The model is also effective at achieving a reasonable source-vocal tract separation which is necessary in order to produce a characteristic representation of the vocal tract. LPC is a simple and straightforward method to implement and does not require as much computation as methods such as the bank-of-filters model. A diagram of the components of a LPC processor is shown in Figure 1.6.

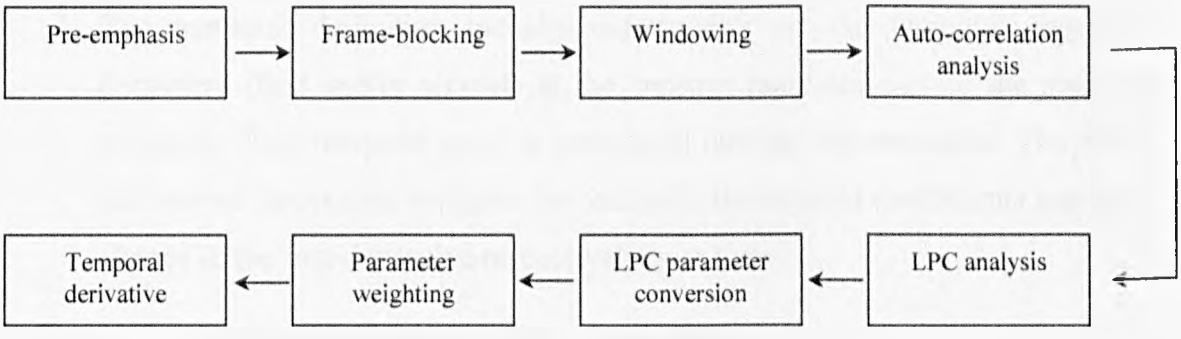


Figure 1.6 Block diagram of LPC processor

The stages involved in the LPC processor include:-

1. Pre-emphasis involves putting the digitised speech signal through a low-order digital system, typically a first order FIR filter, to spectrally flatten the signal and make it less susceptible to precision effects later in the processing.

2. The frame-blocking and windowing steps are the same as described earlier for the MFCC processor.
3. The next step is the auto-correlation analysis of each frame of windowed signal. Rather than finding the correlation between two variables, this process finds the correlation between two values of the same variable at two different times.
4. LPC analysis converts each frame of the auto-correlations into an LPC parameter set; the set may take the form of LPC coefficients, reflection coefficients, the log area ratio coefficients or the cepstral coefficients.
5. If LPC analysis was not used to derive the LPC cepstral coefficients, then the LPC parameter conversion is used to achieve this. The cepstral coefficients are the coefficients of the Fourier transform representation of the log magnitude spectrum. They are more robust and reliable than any of the other coefficients derived in the previous step.
6. Parameter weighting applies weights to the cepstral coefficients by a tapered window so as to minimise the sensitivities of the coefficients to spectral slope and noise.
7. The temporal derivative includes information on the temporal cepstral derivative (first and/or second) in the cepstral representation of the speech spectrum. Thus temporal order is introduced into the representation. The first and second derivatives estimate the change in the cepstral coefficients and the change in the first derivative respectively over time.

1.3.3.3 Hidden Markov models (HMMs)

HMMs have been increasingly used for automatic speech recognition since their introduction in the 1970's. The HMM is a finite set of states, each of which is associated with a, generally multidimensional, probability distribution. Transitions among the states are governed by a set of probabilities called transition probabilities. In a particular state an outcome or observation can be generated, according to the associated probability distribution. Only the observations are seen and the goal is to

infer the hidden state sequence. For example, the hidden states may represent words or phonemes, and the observations represent the acoustic signal. Continuous speech recognition using HMMs[72] and, furthermore, using Hidden-Articulator Markov Models[73], in which each state represents a particular articulatory configuration, have been successfully used.

1.3.3.4 Digital filtering

Digital filters enable the separation of a signal into its component parts. This allows the analysis of separate parts of the signal and removal of interference or noise. While analogue filters are cheap, fast and have large dynamic ranges for both amplitude and frequency, digital filters are far superior in their performance. Digital filters allow a high degree of accuracy and control over the signal. The four most common filters are known as Elliptic, Chebyshev Types I and II and Butterworth.

An elliptic filter, also known as a Cauer filter, is an electronic filter with equalised ripple behaviour in both the passband (band of frequencies allowed through the filter) and the stopband (band of frequencies disallowed by the filter). The amount of ripple in each band is independently adjustable, and no other filter of equal order possesses a faster transition in gain between the passband and the stopband, for the given values of ripple. Alternatively, the ability to independently adjust the passband and stopband ripple can be exchanged for the property of a filter which is maximally insensitive to component variations.

As the ripple in the stopband approaches zero, the filter becomes a type I Chebyshev filter; while as the ripple in the in the passband approaches zero, the filter becomes a type II Chebyshev filter. Chebyshev filters are analogue or digital filters named after Pafnuty Chebyshev; their mathematical characteristics having been derived from Chebyshev polynomials. Chebyshev filters have the advantage that they minimise the error between the idealised filter characteristic and the actual characteristic over the range of the filter, but still possessing ripples in either the passband or the stopband. Because of the passband ripple inherent in Chebyshev filters, filters which have a smoother response in the passband but a more irregular response in the stopband are preferred for some applications.

As both ripple values approach zero, the filter becomes a Butterworth filter. The Butterworth filter was first described by the British engineer Stephen Butterworth in 1930. It is designed to have a frequency response which is as flat as mathematically possible in the passband. They are also known as maximally flat magnitude filters. Butterworth filters can achieve a high gradient between the stopband and the passband thus allowing an accurate selection of frequencies to be made.

Figure 1.7 shows the frequency response of all the classic electronic filters. The Butterworth filter has the slowest transition but has no ripples; the types I and II Chebyshev filters have ripples in the passband and stopband respectively; whilst the elliptic filter has the sharpest transition but it shows ripples in both the passband and the stopband.

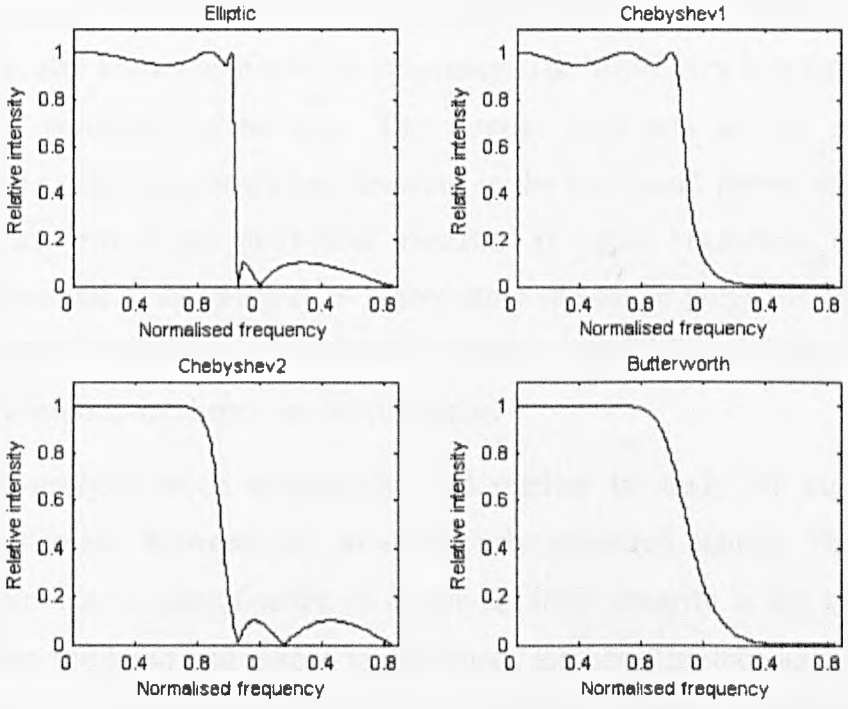


Figure 1.7 Plots illustrating the response of the classic electronic filters: The elliptic filter has the sharpest transition but it shows ripples in both the pass-band and the stop-band; the two Chebyshev filters are relatively sharp with ripples in part of the spectrum; the Butterworth filter has the slowest transition but has no ripples

1.3.3.5 Higher order spectral analysis (HOSA)

HOSA allows the analysis of signals corrupted by non-Gaussian noise or signals arising from a nonlinear process. Higher-order spectra, which are defined in terms of the cumulants (higher-order moments) of a signal, contain additional information that is not conveyed by the signal's autocorrelation or power spectrum. Higher-order spectra are useful because they suppress additive coloured Gaussian noise of an unknown power spectrum, identify non-minimum phase signals, extract information due to deviations from Gaussianity and detect and characterize nonlinear properties in signals.

In mathematics, in the area of statistical analysis, bicoherence is a squared normalised version of the bispectrum. The bicoherence takes values bounded between 0 and 1, which make it a convenient measure for quantifying the extent of phase coupling in a signal. It is also known as bispectral coherency. The bispectrum is a statistic used to search for nonlinear interactions. The Fourier transform of the second-order cumulant, i.e., the autocorrelation function, is the traditional power spectrum. The Fourier transform of the third-order cumulant is called bispectrum or bispectral density. They fall in the category of higher order spectra, or polyspectra and provide supplementary information to the power spectrum. The third order bispectrum is the easiest to compute, and hence the most popular.

Coherence analysis is an extensively used method to study the correlations in frequency domain, between two simultaneously measured signals. The coherence function provides a quantification of deviations from linearity in the system which lies between the input and output measurement sensors. The bicoherence measures the proportion of the signal energy at any bifrequency that is quadratically phase coupled.

Bispectrum and bicoherence may be applied to the case of non-linear interactions of a continuous spectrum of propagating waves in one dimension.

1.3.4 Data pre-treatment

Data pre-treatment is an essential step to prepare raw data for an ultimate classification or analysis stage in order to obtain the best possible result. Different pre-treatment methods emphasise different aspects of the data and each pre-treatment method has its own merits and drawbacks.

Mean-centring, which involves subtracting the mean of the data from each data-point, removes inherent magnitude effects and prevents skewing[74]. Auto-scaling involves dividing the mean-centred data by the standard deviation and gives all the data unit variance[74]. Savitsky-Golay smoothing[75] essentially performs a local polynomial regression on a distribution to determine the smoothed value for each point. Principal component analysis (PCA)[76] reduces the dimensionality in a dataset whilst maintaining the characteristics possessing the maximum variance.

The choice for a pre-treatment method depends on the question to be answered, the properties of the data set and the data analysis method selected. Data pre-treatment can be used to remove noise, interferents due to systematic variations rather than the variations under study or features common to more than one group. Essentially, data pre-treatment accentuates the properties of interest, i.e. those under analysis, and attenuates all others.

1.3.4.1 Principal component analysis (PCA)

PCA is the decomposition of a data matrix into eigenvectors and eigenvalues; where each eigenvector represents an abstract factor or data vector and each eigenvalue represents the relative importance of the associated eigenvector. The eigenvectors are calculated consecutively such that each successive eigenvector accounts for a maximum of the variation in the data. The linear PCA transformation then chooses an optimised orthogonal coordinate system for the dataset such that the greatest variance by any projection of the data set comes to lie on the first axis (called the first principal component), the second greatest variance on the second axis, and so on. Since the smallest set of eigenvalues account for experimental error only, PCA can be used for reducing dimensionality in a dataset while retaining those characteristics of the

dataset that contribute most to its variance by eliminating the later principal components (PCs).

Any non-singular matrix can be decomposed into a scores matrix and a loadings matrix, the combination of which will exactly reproduce the original matrix. The scores give information regarding any trends between samples and the loadings represent the variation and importance of the variables.

PCA is a useful tool for exploratory analysis; by plotting the data set on the new axes formed by the PCs, the relative distribution of variance can be discerned. Groupings of objects of similar structure will occur and outlying data points can become more clearly recognisable.

1.4 NEURAL NETWORKS

1.4.1 Artificial neural networks (ANNs)

Artificial Neural Networks (ANNs) are a paradigm of the biological nervous system and are designed to imitate the information processing and knowledge acquisition of the mammalian brain. A neural network is a parallel distributed processor consisting of simple processing elements which has a natural capacity for storing experiential knowledge and making it available for use. Connections between the neurons are weighted to store the acquired knowledge and control the functioning of the network. The learning process is performed by a learning algorithm, the function of which is to modify the weights between the neurons so that the desired output is obtained. This step can be seen as an optimisation process in which the neural network 'learns' to give accurate and correct results from the inputs it receives.

Neural networks offer many advantages over current statistical methods of data analysis. They are well suited to performing typically human tasks such as memorising objects, recognising patterns, generalising, estimating parameters and making decisions. They also have a strong ability to cope with noisy data, tolerate faulty data and adapt to circumstances.

Neural networks are highly advantageous to the applications of pattern recognition and data classification. They can derive meaning from complicated data and can be used to extract patterns and trends that are too complex to be recognised by other computer techniques.

1.4.1.1 History of neural networks

The original work on neural networks was published in 1943 by McCulloch and Pitts[77] who tried to understand the functioning of the nervous system by defining primitive information processing elements, based on mathematical logic, that represented the functional properties of biological neurons and their properties. Their model of a neuron was assumed to follow an "all-or-none" law and with a sufficient number of such units and synaptic connections, operating synchronously, they showed that a network could, in principle, perform any computable function. In 1949, Hebb described a learning rule for synaptic modification which was derived from observations of neurophysical experiments on biological neural networks[78]; now commonly known as the Hebbian learning rule. The theory suggested that neural pathways are strengthened each time they are used and that neurons adapt and form neural assemblies as learning occurs. However, up until this point, research into artificial neurons was still theoretical due to the lack of computer technology available[79]. In 1954, Gabor, the inventor of holography and one of the early pioneers of communication theory, introduced the idea of a nonlinear adaptive filter; a device that was to take six years for his team to build[80]. Learning was accomplished by feeding samples of a stochastic process into the machine together with the target function that the machine was expected to produce. In 1956, work by Rochester *et al.* was possibly the first attempt to use computer simulation to test a well-formulated neural theory based on Hebb's postulate of learning[81]. The attempt failed but showed some important requirements necessary for success. By 1958, Rosenblatt had developed the first precisely specified, computationally orientated neural network called the Perceptron[82]. Rosenblatt described a novel method for the supervised learning of a network of binary decision units (BDNs). By altering the synaptic strengths each time the network gave a wrong answer, the network gradually "learned" to give the correct answer instead. In 1960, Widrow and Hoff developed

the least mean-square algorithm and used it to develop Adaline and Madaline, two models that made use of adaptive linear elements[83]. By this time, although success had served as an impetus for the growth of artificial neural networks, and indeed encouraged more research into the area, a theoretical study by Minsky and Papert in 1969 was to severely damage this enthusiasm[84]. They showed perceptrons to have very serious limitations when applied in the form and manner which had been used up to that time. In addition, they argued that even extending the perceptron to multi-layer networks would not improve results[85]. The discovery of these problems impeded the research into neural networks. Many scientists were not encouraged to continue research and funding became less available for those that were. However, despite this, the 1970s saw the emergence of self-organising maps using competitive learning. Von der Malsburg first demonstrated self-organisation in 1973[86], and with Willshaw in 1975[87], showed the formation of self-organising maps, motivated by topologically ordered maps in the brain. The back propagation algorithm introduced by Werbos in 1974 solved a problem with training hidden neurons[88]. The algorithm allowed a neuron to propagate its errors back to former layers of the network.

1982 saw a resurgence of interest in neural networks. Hopfield introduced nonlinear transfer functions for the evaluation of the final output from neurons[89]. The Hopfield neural network enabled auto-association by means of which any information, representable by a multivariable or matrix e.g. an image, is regenerated from only partial or corrupt data. Hopfield used the idea of an energy function based on physical theory to formulate a new computation necessary for his network. This development paved the way for a surge of physical theory to enter neural modelling, thereby transforming the field of neural networks. Another important development that year was Kohonen's self-organising maps[90], which differed to the earlier work by Willshaw and von der Malsburg and which received far more attention.

In 1985, Ackley *et al.* developed the Boltzmann machine[91], a neural network very similar to that developed by Hopfield. However, in addition to the basic Hopfield elements, the Boltzmann machine used a probabilistic learning algorithm inspired by Ludwig Boltzmann's work in statistical mechanics; the Boltzmann distribution of energy.

In 1986 Rumelhart *et al.* reported development of the back-propagation algorithm[92], and despite its original mention a decade before by Werbos, much of the credit now goes to Rumelhart's group for their extensive work on the area. In 1988, Broomhead and Lowe described a procedure for the design of layered feed-forward networks using radial basis functions (RBF), an alternative to multilayer perceptrons[93].

To date, neural networks have continued to be developed, aided by the increasing power of computer processors. Their rising popularity has seen their application in many areas, both in research and commerce.

1.4.2 Applications of ANNs

Neural networks have a wide variety of applications to real problems. Although many neural networks can now achieve statistically high levels of accuracy, they are still not without a degree of error. While work continues to improve accuracy, some uses are being found for neural networks at their current levels of performance. In some decision-making processes, for example loan approval, a 90% accurate system is still an improvement over existing methods. Some banks have shown that the failure rate on loans approved by neural networks is lower than those approved by some of their best traditional methods. Credit card companies are using neural networks to aid in establishing credit risks and credit limits. Neural networks are also finding applications in stocks, bonds and international currency[94]. In addition, they are being used in the detection of credit card fraud by automatically recognising unusual spending patterns[95] and in the same way for detecting fraud in mobile phone networks[96]. Companies are also using neural networks for targeted marketing to ensure more effective and successful methods[97].

Success has also been achieved in the forecasting of sea surface temperatures[98]. Whilst the elimination of noise from phone lines using Widrow and Hoff's Madaline[83] has been used for many years.

Neural networks are being applied in food analysis for the pattern recognition of data collected using electronic noses and tongues[99]. A similar application is in quality control, where constant mechanical monitoring can achieve much better results than human inspectors who can easily become fatigued or distracted. Manufacturing has

made use of neural networks in system design, process control, robot scheduling and operational decision[100].

An important field in which neural networks have also been applied is medical diagnosis and prognosis[101]. Uses include the image processing of MRI outputs, the prediction of risk for the relapse or spread of cancer, primary screening tools e.g. for cervical cancer[102] or breast tumours[103, 104], ECG surveillance, detection of heart murmurs[105] and many more.

Neural networks are also increasingly being used for classification of chemical data such as analytical data from near infra-red (NIR)[106], mass spectrometric (MS) and nuclear magnetic resonance (NMR) techniques[85, 107]. Image processing using neural networks can also be put to many uses, such as face identification[108], document analysis[109], license plate recognition and fingerprint analysis[110].

Neural networks have been applied to automatic cough recognition, with particular focus on pig cough as an early sign of disease by Van Hirtum *et al.*[111-114].

1.4.3 Description of neural networks

The problems which neural networks can be applied to can be divided into four basic types[107]:- association (auto or hetero); classification; transformation and modelling.

Auto-association is the ability of the system to reconstruct an incomplete or corrupted pattern from a learned pattern. Hetero-association is the ability of the system to make a one-to-one association between members of two sets of patterns. Thus, on the input of a letter, for example 'A', the system can respond with an output of the number representing that letter's position in the alphabet, so 'A' would give the output '1'.

Classification assigns all inputs to appropriate classes depending on the characteristics they share. Following training, the neural network will then be able to subsequently classify any further inputs based on the clusters it has already constructed.

Transformation is the mapping of a multivariate space onto a space of lower dimensionality for example, representing a 3-dimensional object onto a 2-dimensional plane.

Modelling uses an analytical function or algorithm which will give a specified output for a given input. The training of the network depends on the 'fitting' of the model in which the operation of the algorithm is determined by the best agreement between the input data and the calculated output data.

1.4.3.1 ANN Architecture

The application of the success of the principles behind biological neurons to artificial data processors has lead to the use of a similar neural structure. The artificial neuron receives one or more signals simultaneously and sums them to give a collective input. The neuron then multiplies the total signal by a characteristic synaptic weight. An activation function or transfer function is then applied to limit the output of the neuron to a permissible value.

The operation of a neural network depends on the interconnection of the entire network rather than the workings of each individual neuron. It is also dependent on several important steps in the operation of the neural network including weighting of the input, summation of the weighted input values, the application of a transfer function and the addition of bias [See Figure 1.8].

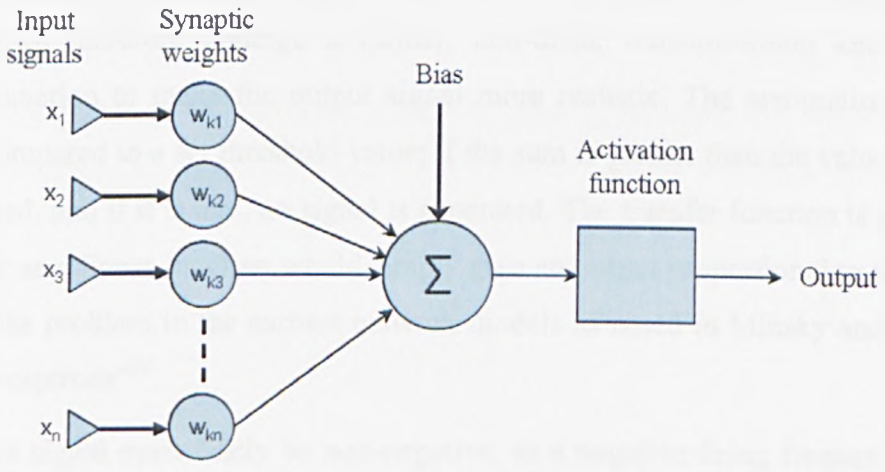


Figure 1.8 Typical model of a neuron

1.4.3.1.1 Weights

The weight of a neuron is the value that is multiplied to the input signal of that neuron to achieve the desired output. These weights can be modified in response to various training sets and according to a network's specific topology or through its learning rules. Each neuron can receive many signals simultaneously and represent these inputs by adjusting the strength of the output it passes on to the next neuron. It is the continuous adjusting of the interconnecting synaptic weights which trains the network to give the correct output to a particular input. Once the weighted inputs have been calculated, the sum of all the inputs must be computed to determine the significance of each. The simplest method is to multiply each input with the weight of the receiving neuron and add all the values together. The summation function can, however, be more complex than the input and weight sum of products and can be calculated in several ways by use of algorithms.

1.4.3.1.2 Transfer functions

The result of the summation function is then passed through a transfer function to be transformed into a working output. It would not be appropriate to simply represent the output as the weighted sum of the input signals, since the output of an artificial neuron is one of two possibilities; it either activates or it does not. The net input to the neuron must therefore undergo a further, non-linear transformation known as a transfer function to make the output signal more realistic. The summation total is usually compared to a set threshold value; if the sum is greater than the value a signal is generated, and if it is less, no signal is generated. The transfer function is generally non-linear as a linear function would simply give an output proportional to the input; this was the problem in the earliest network models as noted in Minsky and Papert's book "Perceptrons"[84].

The output signal must firstly be non-negative, as a negative firing frequency is not plausible, and secondly must be continuous and confined to a specified interval. There are several transfer functions applicable in these instances.

1.4.3.1.2.1 Hard-limiter

The hard-limiter transfer function gives only one of two values, zero or one. A threshold value determines the input required to change the output value from zero to one; once the input is above the threshold value, the neuron will 'fire'. A slightly modified form of the hard-limiter is the bipolar hard-limiter, in which the two values used are +1 and -1, as shown in Figure 1.9.

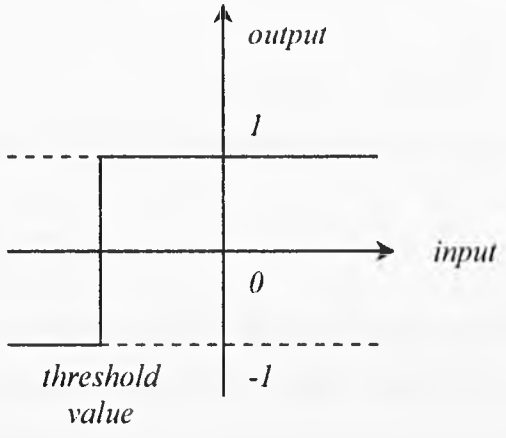


Figure 1.9 A bipolar hard-limiter transfer function

1.4.3.1.2.2 Threshold logic

The threshold logic transfer function is similar to the hard-limiter in that the output limits are between zero and one. However, there is also a swap interval included in which the output is linearly proportional to the input. It therefore represents the input within a given range and also acts as a hard limiter outside of that range, as shown in Figure 1.10.

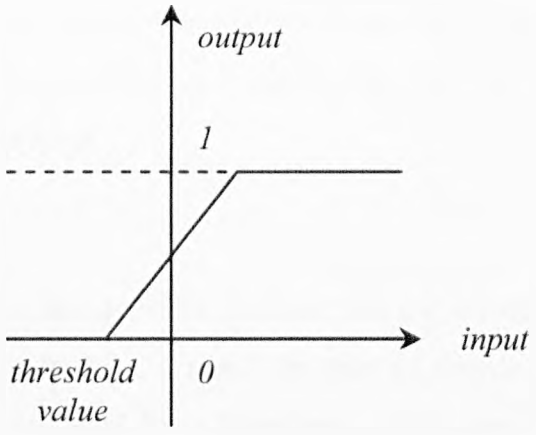


Figure 1.10 The threshold logic transfer function

1.4.3.1.2.3 Sigmoidal function

The sigmoidal function is similar to the threshold logic transfer function except that a sigmoid or S-shape curve replaces the linear region. The curve approaches the minimum and maximum values at the asymptotes, which still have values of zero and one. The curve is commonly called a sigmoid when it ranges between 0 and 1 (as shown in Figure 1.11.), and a hyperbolic tangent when it ranges between -1 and 1.

The threshold and sigmoidal functions are not simple logic functions which give a yes/no or true/false output. Instead, they hold all the values between the two extremes and this possibility forms the link between artificial neural networks and "fuzzy logic".

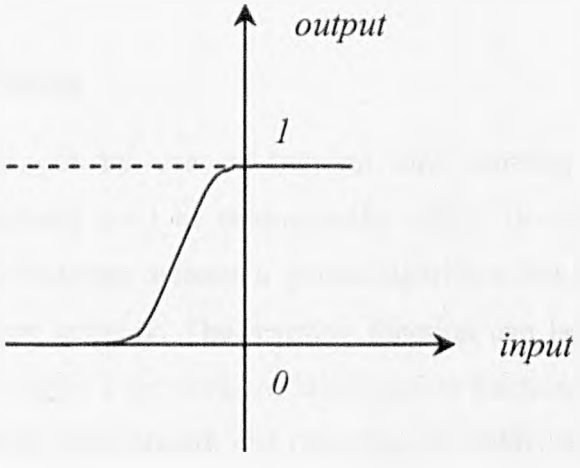


Figure 1.11 The sigmoid threshold logic transfer function

This is not a definitive list of transfer functions but instead three of the more commonly used. The chosen function is highly dependent on the use and application of the neural network at hand.

1.4.3.1.3 Bias

The addition of bias to the decision function has the effect of applying an affine transformation to the output i.e. a transformation of coordinates that is equal to a linear transformation followed by a translation. This can be illustrated with the example given Figure 1.12 in which the use of a bias function allows the separation of point D from points A, B and C.

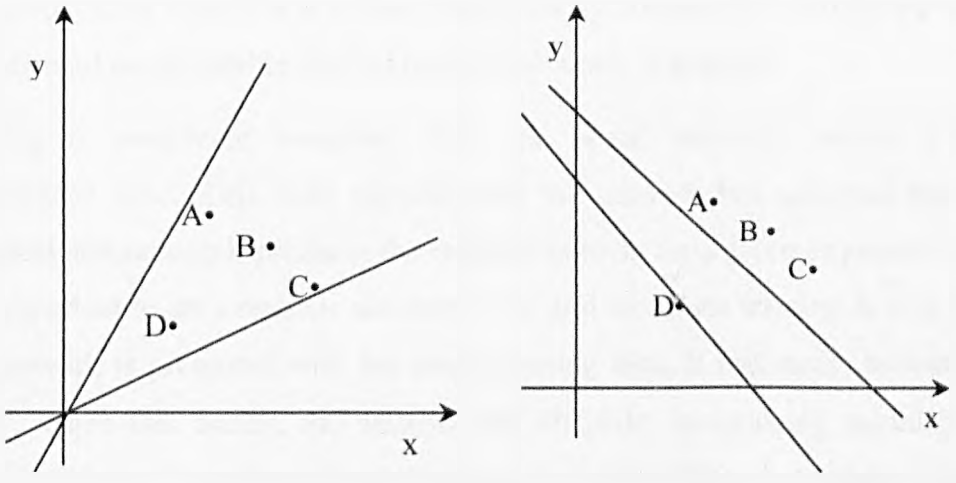


Figure 1.12 The effect of implementing a bias function to the separation of D from A,B and C

1.4.3.1.4 Training

Training is carried out by use of training and learning functions, these are mathematical procedures used to automatically adjust the network's weights and biases. The training function dictates a global algorithm that affects all the weights and biases of a given network. The learning function can be applied to individual weights and biases within a network. As discussed in Section 3.3.5.1.2.1, due to the danger of over-training the network and removing its ability to generalise, a training error of zero is not necessarily the desired aim. A performance function is therefore

applied to measure the error of the training process and to decide when training is complete.

The process of network training can be carried out in a supervised or unsupervised manner, generally depending on the application and the type of network employed.

1.4.3.1.4.1 Supervised training

Training with supervision requires an external input to the system to provide the network with information about the correct output. The network then compares the actual output to the desired output and adjusts the weights, which are usually randomly set to begin with, to produce a closer match at the next cycle or iteration. The learning method tries to minimize the current errors of all processing elements. This global error reduction is created over time by continuously modifying the input weights until an acceptable level of network accuracy is reached.

Training is considered complete when the neural network reaches a defined performance level. This level signifies that the network has achieved the desired statistical accuracy as it produces the required outputs for a given sequence of inputs. It is important to set a realistic accuracy level and terminate training as it is reached. If a network is presented with too much training data, it can easily become 'over-fitted'. When this occurs, the network has difficulty recognising anything that is slightly different from the training data and the only solution is to clear and re-train the network. When no further learning is necessary, the weights are saved by the network ready for application.

The training data must express all the properties and relationships which the network must learn in order to carry out accurate future predictions. The data must also contain as much and as varied information as possible at a time. This is to avoid the network adjusting weights very specifically for one property and then 'forgetting' that property when the network is altered for the next property.

ANNs only deal with numerical data therefore any raw data must be appropriately formatted before being presented to the network. The best format and conditioning technique will be different for each application.

The final stage is to validate the neural network by testing the outputs for known data that has not been presented to the system previously. If the outputs are correct and accurate to a satisfactory standard, then the training is complete and the network is ready for application. If there are discrepancies in the outputs, however, further training must follow to fine-tune the system. This testing is critical to ensure that the network has not simply memorized a given set of data but has learned the general patterns involved within an application.

1.4.3.1.4.2 Unsupervised training

With unsupervised learning, a system requires no external influences to adjust its weights. The network is presented with a set of data and finds any trends or patterns within the input signals by internally monitoring performance. The system must still have an optimisation criterion that is used for the evaluation of the result produced at each iteration, so that there is an inherent aim for the system to work towards. This information is built into the topology and learning rules.

1.4.3.1.5 ANN architecture

The way in which the neurons of a neural network are structured can be classified into three fundamentally different types.

1.4.3.1.5.1 Single-layer feed-forward networks

A layered network consists of neurons organised into the form of layers. The simplest form of a layered network contains an input layer of source nodes collecting the input signal that projects onto an output layer of computation nodes. The single layer refers to the one layer of computational nodes present; the source nodes simply gather the information to be processed. The network is a feed-forward type in that information only passes forward; each node in the input layer is connected to every node in the output layer so that all information is passed forward.

1.4.3.1.5.2 Multilayer feed-forward networks

Multilayer feed-forward networks contain the presence of one or more hidden layers. The hidden layers, which correspondingly contain hidden neurons, provide an

intermediate stage of computation between the input and the network output. The extra layers added to a network allow it to extract higher-order statistics; a particularly valuable property when the size of the input layer is large. Again, each node in a layer is connected to every node in the next layer so that all information is passed forward. One example of this network type is the radial basis function neural network.

The Kohonen neural network is a self-organising, unsupervised version which uses topology as a method of dealing with information. The network architecture consists of a single layer of neurons arranged in a two-dimensional plane and learns by competitive learning. In competitive learning, all neurons in the output layer compete among themselves to 'fire'. The network selects either the neuron that has the largest output in the entire network or the one that has the weight vector most similar to the input signal. Only the selected neuron will produce an output signal in a winner-takes-all method, it is then further optimised to make its weight even closer to the desired value. The weights of the neighbouring neurons are then also altered, usually scaled down to increase the contrast between those and the firing neuron. This scaling down process is largely dependent on the distance from the winning neuron and is thus known as a topology dependent function.

The Learning Vector Quantisation (LVQ) neural network is a supervised version of the Kohonen algorithm. LVQ networks are often used for pattern recognition and classification problems.

1.4.3.1.5.3 Recurrent networks

A recurrent neural network is distinguished from a feed-forward neural network in that it contains at least one feedback loop. Each neuron feeds its output signal back into the inputs of all the other neurons in the same layer. A self-feedback loop is where the output of a neuron is fed back into its own input. The presence of feedback loops increases the learning capability of the network and its performance. One example is the Hopfield neural network, developed in 1982 by J. J. Hopfield[89] which is a single-layer recurrent network capable of performing auto-association.

The probabilistic neural network (PNN) was developed by Donald Specht[115, 116]. The network provides a general solution to pattern classification problems by following an approach developed in statistics, called Bayesian classifiers. Bayes theory, developed in the 1950's, takes into account the relative likelihood of events and uses *a priori* information to improve prediction. The PNN has a feed-forward architecture and uses a supervised training algorithm similar to back-propagation. In recall mode, these distribution functions are used to estimate the likelihood of an input feature vector being part of a learned category.

1.5 HARDWARE COMPONENTS

The requirements of a device for the acoustic monitoring of cough are portability, adequate power and data storage to carry out 24-hour good quality recordings and a suitable microphone for picking up coughs from the subject whilst excluding extraneous background sounds.

1.5.1 Microphones

Microphones are transducers which detect sound signals and produce an electrical image of the sound. All microphones capture sound waves as vibrations and convert them by various methods into an electrical signal that is an analogue of the original sound. Most microphones in use today use electromagnetic generation (dynamic microphones), capacitance change (condenser microphones) or piezoelectric generation to produce the signal from mechanical vibration.

Dynamic microphones consist of a lightweight diaphragm, usually made of plastic film, attached to a very small coil of wire suspended in the field of a permanent magnet. When a sound causes the diaphragm to vibrate, the whole assembly works as a miniature electricity generator, and a minute electric current is produced.

The capacitor microphone consists of an extremely light disk and a rigid back plate, separated by an insulator, the disk being free to move in sympathy with sound waves incident upon it. The capacitor is charged via a very high resistance. The condenser

works on the principle that sound pressure changes the spacing between a thin metallic membrane and a stationary back plate which causes the capacitance to vary.

The electret microphone is a version of the capacitor microphone. Instead of applying an electrical charge to the microphone capsule via an external power source, electret microphones use a diaphragm made from an insulating material that has a permanent electrical charge. A back-electret microphone has a permanently-charged material fixed to a stationary back-plate instead of the diaphragm in order to allow much thinner diaphragms to be employed.

A piezoelectric microphone uses the phenomenon of piezoelectricity, which is the ability of some materials to produce a voltage when subjected to pressure, to convert vibrations into an electrical signal.

A laser microphone consists of a laser beam that must be reflected off a glass window or another rigid surface that vibrates in sympathy with nearby sounds. This device essentially turns any vibrating surface near the source of sound into a microphone. It does this by measuring the distance between itself and the surface extremely accurately; the tiny fluctuations in this distance become the electrical signal of the sounds picked up.

A contact microphone is designed to pick up vibrations directly from a solid surface or object, as opposed to sound vibrations carried through air. They are frequently applied to detect noise of a very low level or in situations where there are high levels of ambient noise. Contact microphones can be based on either piezoelectric microphones or a version of the dynamic microphone, in which a contact pin is used to detect vibrations and transfer them to the coil of a magnetic transducer.

1.5.2 Directionality

The directionality of a microphone indicates how sensitive it is to sounds arriving at different angles about its central axis. The sound picked up by the microphone can be controlled by the use of directional microphones which have specific pick-up patterns [See Figure 1.13]. A microphone that picks up sound equally from all angles is called omnidirectional, but this level of detection can be unfavourable when there are

unwanted sounds in the recording area. Directional microphones are more specific in their detection. Cardioid microphones pick up sound in a heart- shaped pattern such that the least sensitive spot is right behind it. Hypercardioid microphones, sometimes also known as supercardioids, are similar but with an extra small lobe of sensitivity to the rear. Bidirectional or figure-of-eight microphones, receive sound equally from the front and the back. Shotgun microphones have high sensitivity at the front, with additional lobes of sensitivity to the left, right and rear.

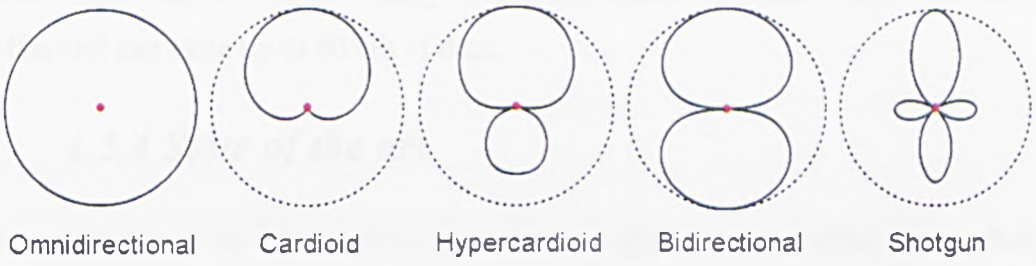


Figure 1.13 Common pick-up patterns of microphones. The microphone is positioned at the central red point, the dashed line represents the range of sound and the solid line represents the sound detected

1.5.3 Recording devices

The technology to make 24 hour, ambulatory recordings has long been awaited, yet constraints of sufficient data storage and power supply have severely delayed such advances. For many years, the most popular, and perhaps most advanced method, was the use of digital audio tape (DAT) recorders. DAT recorders were first introduced by Sony and Philips in the mid 1980s and it wasn't until December 2005 that Sony decided to discontinue production. DAT recorders make digital recordings on 6mm tape cartridges, using a modified video recording mechanism with a rotating tape head. They have been used widely in professional broadcasting because of their capability for making high quality recordings and usually have a digital output stream capability so that recordings can be transferred digitally to a computer system.

The next development was that of the MiniDisc (MD) recorder, which was introduced onto the market in 1992 by Sony, which made compressed digital recordings on a 2.5" magneto-optical disk protected in a cartridge. These were recently improved by

the development of high-density MDs (Hi-MDs) capable of storing up to one gigabyte (Gb) of audio data. This equates to approximately 34 hours when using Sony ATRAC3plus mode of recording at 64 kbps (kilobits per second). Again, these came with a USB interface which could be used to upload recordings to a computer. However, the mechanical action of the disk still meant power consumption was relatively high and could not provide for long recording durations.

By far the most recent advance in technology has been solid-state recorders, which make digital recordings onto memory cards, such as compact flash cards. The lack of mechanical activity means these devices can record for much longer durations than MDs and can store up to 40 Gb of data.

1.5.4 State of the art

Several devices have been developed for the objective monitoring of cough events. Although a variety of techniques have been employed, they all possess the common feature of acoustic monitoring. The simple recording of audio sounds using digital portable recorders has been employed by both Smith *et al.*[44, 45] and Matos et al.[46] who used various types of mp3 recorders. However, 24 hour recordings were not performed in either of these studies; Smith *et al.* quoted their inability to achieve sufficient power and data storage whilst Matos *et al.* did not specify their reasons for the limited duration, leaving the reader to infer whether they were unfeasible or simply unnecessary to the study.

A limited degree of mobility was allowed during audio recordings for cough studies by Pavesi *et al.*[41] and Rietvald *et al.*[42]. Patients were fitted with a microphone and a transmitter while a receiver and recorder were situated within the area of study. Although the systems were objective and suited to long recording durations, they were not fully ambulatory as the patients were limited to the study environment for the entire time.

The combination of audio recording with the measurement of EMG signals has been achieved by both Chang *et al.*[36] and Hsu *et al.*[38] in order to monitor cough sounds alongside the activity of the lower respiratory muscles. Cough was thus identified as a positive response occurring simultaneously in both channels.

Accelerometers have frequently been employed in cough studies to monitor the motion of the patient during cough events. Most recently, Paul *et al.*[47] used an accelerometer to measure vibrations in the patient's chest. In addition, Munyard *et al.*[39] used both the recording of audio signals and an accelerometer with ECG monitoring to determine the patient's level of activity. All channels were then considered in order to define a cough event.

Perhaps the most comprehensive cough monitoring device is the one developed by Coyle *et al.*[43] which records multiple data from four sources; a contact microphone for cough sounds, an accelerometer for posture, an ECG for heart rate and RIP to measure ventilation. A positive response must then occur simultaneously in eight channels for an event to be defined as a cough. Although the device is objective, portable and has been shown to work for 24 hour studies it has drawbacks including expense and impracticalities of wearing the device for such long durations.

1.6 EXISTING SYSTEM

Prior to beginning this work, Adrie Dane and workers at Castle Hill Hospital (Cottingham, Hull, UK) had developed a prototype for the methodology of automatic recognition and counting of cough[48] [See Appendix C]. This section introduces the existing work so as to make the unique contributions of this thesis clear.

The initial focus of the work was on the development of the automatic analysis of cough recordings. This method for the computerised processing of audio recordings operates in three steps; sound detection, feature extraction and pattern comparison

For sound detection, the signal is analysed to identify periods of sound within the recordings; these sound events are then isolated and any periods of silence are omitted from further analysis. The sounds that are isolated then undergo the feature extraction step. DSP is applied to calculate the characteristic spectral coefficients which represent each sound event. The techniques used are LPC and a bank-of-filters front-end processor. The resultant coefficients are reduced by PCA, which highlights the components of the data that contain the most variance, such that only these components are used for further analysis. Finally, the sound events are classified into

cough and non-cough events by applying a PNN to the calculated feature vectors. The PNN is trained to recognise the spectral coefficients of reference coughs and non-coughs and classify future sound events appropriately.

Parameters such as the total number of coughs and cough frequency as a function of time can then be calculated from the results of the audio processing.

1.6.1 Processing software

Original software was developed under Matlab® version 6 (Mathworks Inc., Natick, MA, USA). The following Matlab toolboxes were used: PLS_Toolbox version 2.1.1 (Eigenvector Research Inc., Manson WA), Signal processing toolbox version 5.1 (Mathworks Inc., Natick, MA, USA), Neural network toolbox version 4.0.1 (Mathworks Inc., Natick, MA, USA) and Voicebox (a free toolbox for speech recognition). The programs were executed under Windows 2000 on a 1.4 GHz Pentium 4 PC with 256 megabytes of RAM.

1.6.2 Original hardware

The original device used in the previous work, consisted of using a Sony ECM-TIS Lapel microphone connected to a Sony TCD-D8 Walkman DAT-recorder, powered by AA dry cells. Sound was recorded at a sampling frequency of 48 kHz. For each of the subjects, this recording was converted into 44.1 kHz 16 bit mono Microsoft WAV format.

1.6.2.1 Data processing

Table 1-1 defines the variables and symbols used in the analysis. The operation of the existing system can be summarised into a schematic representation as shown in Figure 1.14.

Table 1-1 Symbols used and their settings. Settings are based on established values and preliminary experiments. Symbols only used locally are explained in the text.

SYMBOL	MEANING	VALUE
f_s	Sampling Frequency	11025 Hz
t	Time in milliseconds	
σ_{signal}	Windowed standard deviation of signal	Calculated as a function of time
$\Delta t_{background}$	Background interval	11026 points (1000 ms)
$thresh_{peak}$	High (event detection) threshold	10 ($\times\sigma_{background}$)
$thresh_{limits}$	Low (event start and end) threshold	2 ($\times\sigma_{background}$)
$\sigma_{background}$	Standard deviation of background	
N_{train}	Number of reference patterns	150 (75 cough/75 non-cough)
$n_{b\text{-}o\text{-}f}$	Number of mel bank-of-filters cepstral coefficients	42 (14+14 1st derivatives +14 2nd derivatives)
n_{LPC}	Number of LPC cepstral coefficients	14 (no derivatives)
$N_{cepstral}$	Total number of cepstral coefficients ($n_{B\text{-}O\text{-}F} + n_{LPC}$)	56
N_{PCA}	Reduced number of features	45

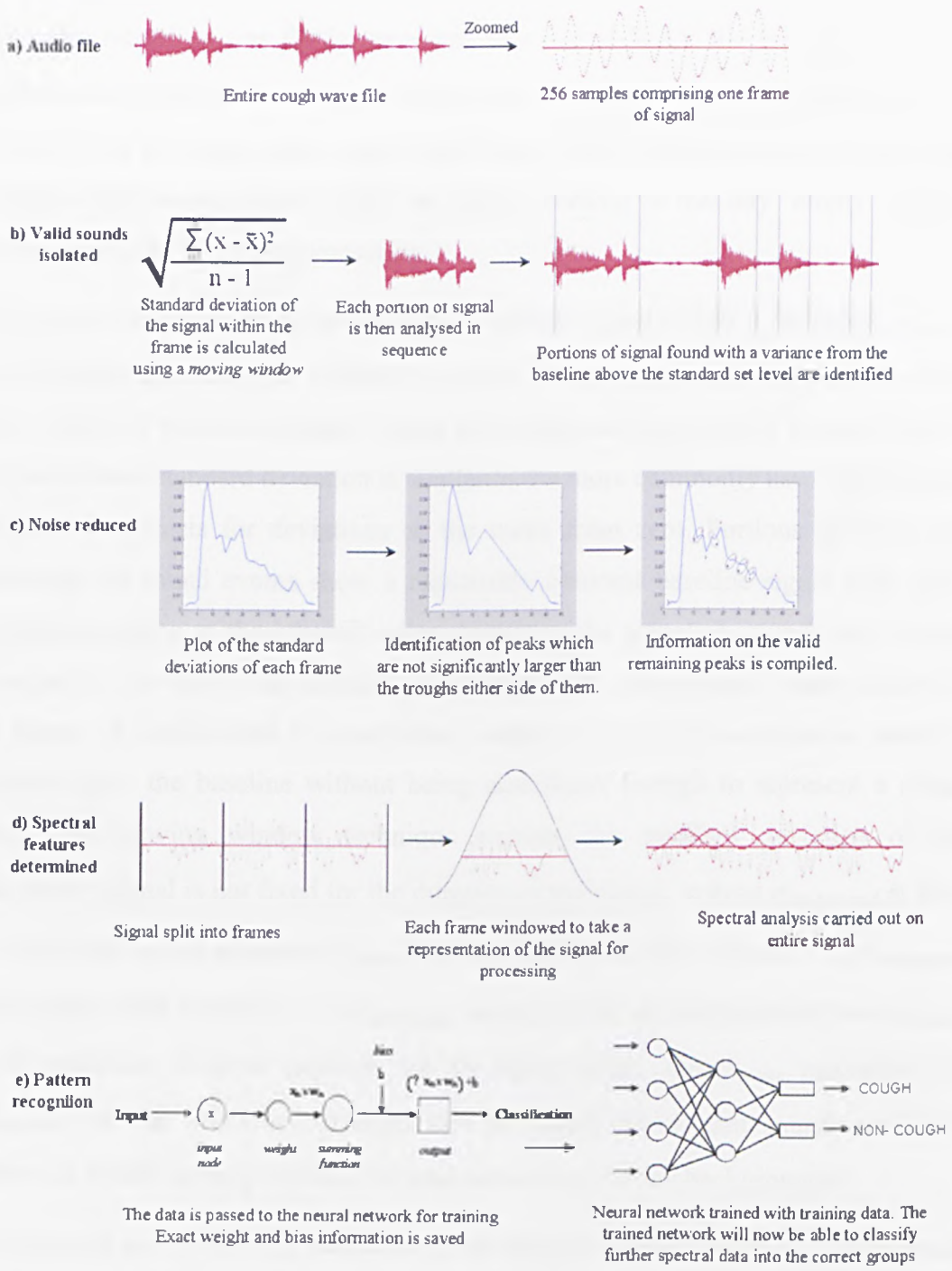


Figure 1.14 Schematic diagram illustrating the pattern recognition approach to cough, non-cough classification

1.6.2.1.1 Sound detection

The audio recording was firstly converted to a digital WAV file for processing. To minimise data storage the sound recordings were analysed at a sampling frequency, f_s of 11025 Hz by using only every fourth point. The initial processing stage then identifies valid sound events within the audio recording so that any "empty" regions can be omitted from further processing.

The signal is analysed using the moving windowed signal standard deviation, σ_{signal} , i.e. the standard deviation as a function of time. The moving window works along the entire length of the audio signal, taking each frame as the centre of a new window. This windowed standard deviation is similar to the more commonly used RMS signal, however, it corrects for deviations of the mean from zero. Portions of the signal containing no sound events show a reasonably constant baseline signal with small deviations relating to the inherent noise present in the signal. A sound event causes the signal to rise above the baseline with a magnitude proportional to the validity of the signal. A noise level is established which represents an acceptable level of variation from the baseline without being significant enough to represent a sound event. The moving window technique ensures the standard deviation of the background signal is not fixed for the duration of the signal; instead $\sigma_{background}$ at time t is calculated as the minimum σ_{signal} between the start of the window, $t - \Delta t_{background}$ and the end of the window, $t + \Delta t_{background}$. Sound events are thus detected when σ_{signal} for a particular window exceeds the threshold value, $thresh_{peak}$, multiplied by $\sigma_{background}$ for that window. Although this procedure means that sound sensitivity varies to a certain extent, it allows for peak detection in noisy backgrounds.

The start and end values of a sound event are defined as the nearest σ_{signal} before and after the peak maximum which are below the defined low level calculated by $thresh_{limits} \times \sigma_{background}$. Portions of the signal that are below this low level are removed and excluded from further analysis. Figure 1.15 illustrates this procedure. Point (a) indicates the first standard deviation larger than $thresh_{peak} \times \sigma_{background}$. Points (b) and (c) are the points nearest to point (a) where σ_{signal} is smaller than $thresh_{limits} \times$

$\sigma_{background}$. The whole region between points (b) and (c) is a sound event. In the same way, the region between points (d) and (e) will be detected as a sound event.

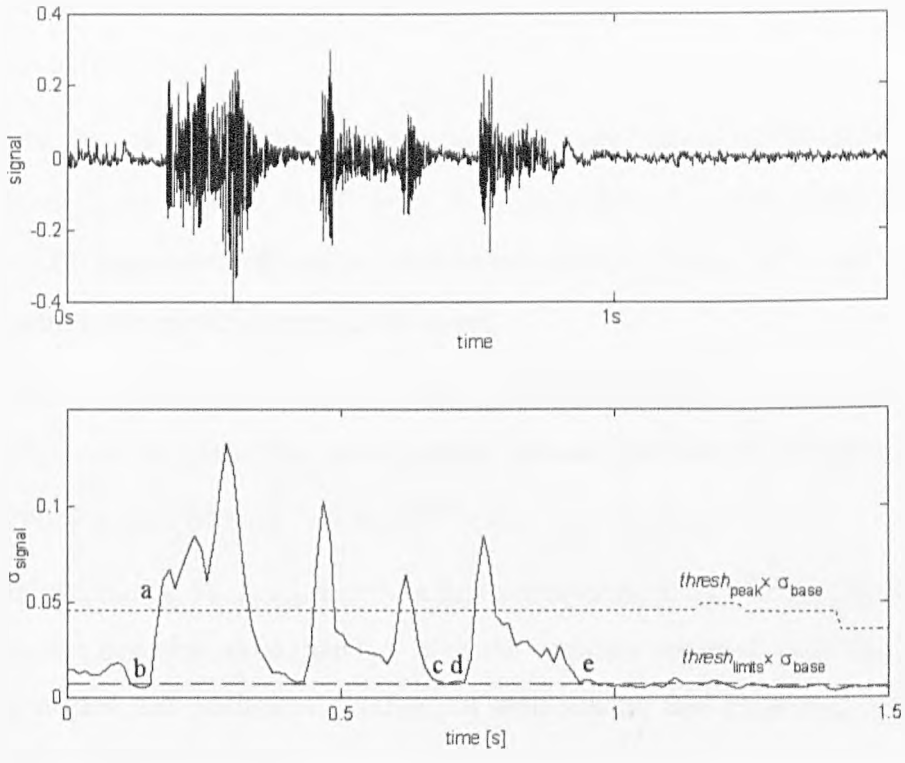


Figure 1.15 Illustration of the method used for sound detection; illustrating the original sound signal (top) and the standard deviation of the signal (bottom)

The amount of noise within the section of signal is then reduced by smoothing. The standard deviations for each frame in the section are plotted and treated as a series of peaks. Peaks with variations lower than the noise-level are removed. The remaining frames of signal are compiled for signal processing.

1.6.2.1.2 Feature extraction

The next stage was to identify characteristic features of the sound events identified to enable subsequent classification. This was achieved by calculation of spectral features of the signal by use of DSP.

Cepstral coefficients are a reliable, robust and most widely used feature set based on frequency content [See Section 1.3.3.2]. They are good at discriminating between

different phonemes, are fairly independent of each other and have approximately Gaussian distribution for a particular phoneme. They are also independent of the sound amplitude, which makes the placement of the microphone less critical and also allows for the fact that cough amplitude varies remarkably for each patient as well as between patients.

The feature vectors used in this method consist of a pre-treated combination of $n_{\text{B-O-F}}$ mel bank-of-filters cepstral coefficients with their delta and delta-delta derivatives and n_{LPC} LPC cepstral coefficients, without derivatives. These delta coefficients are used to capture the speech dynamics of sound.

Pre-treatment consists of scaling followed by projection into the PC space obtained for the reference samples. This pre-treatment reduces the number of features in each pattern from N_{cepstral} ($= n_{\text{B-O-F}} + n_{\text{LPC}}$) to N_{PCA}.

The combined X_{cough}, $X_{\text{non-cough}}$ matrix is firstly auto-scaled; then following PCA, only the scores that describe more than 0.5 % of the variance are used. New data is scaled using the means and variances obtained in auto-scaling and projected onto the PC space using a projection matrix.

1.6.2.1.3 Graphical user interface (GUI)

In order to create and test the pattern recognition stage, reference measurements are required. For this purpose a graphical user interface (GUI) was developed. The GUI allows the visual and aural inspection of an entire audio recording along with manual classification of sound events. The GUI is created by the function *coughgui* which performs the operations as described in Section 1.6.2.1.1 and displays the results. The GUI display includes the time/amplitude signal plot [Figure 1.16], the time/frequency plot [Figure 1.17] and initially, the labels identifying each located sound event. By selecting and changing labels, sound events are manually classified as either coughs or non-coughs for identification in the pattern classifying stage. Following the later processing stages, the GUI displays the classification results from the neural network.

The design also includes the ability to manually scroll through the audio data and perform rapid and accurate manual counting.

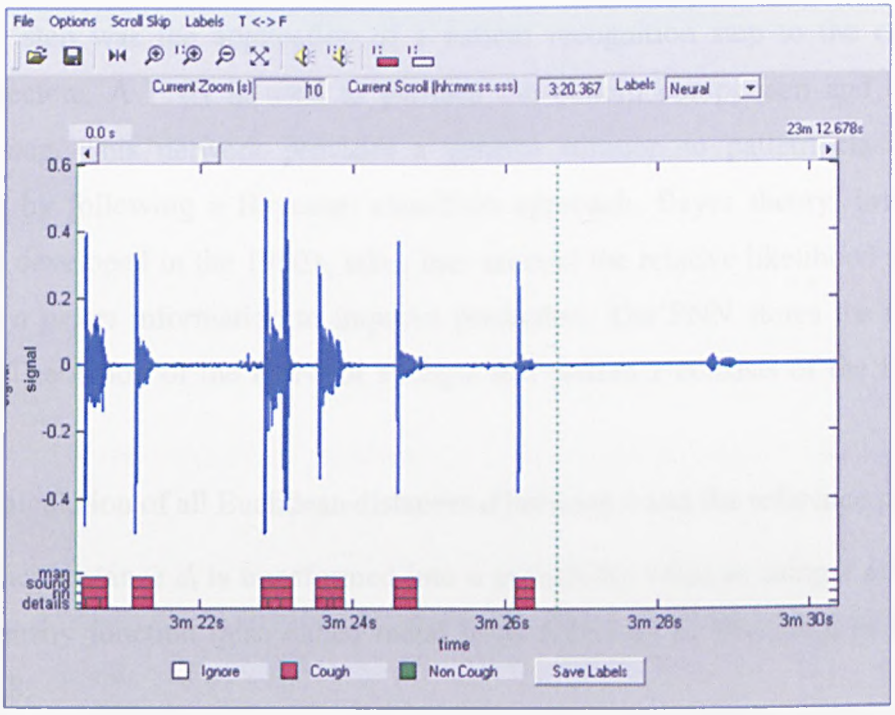


Figure 1.16 The GUI output of the software showing the time/amplitude signal plot for a 10 second window of a processed audio recording

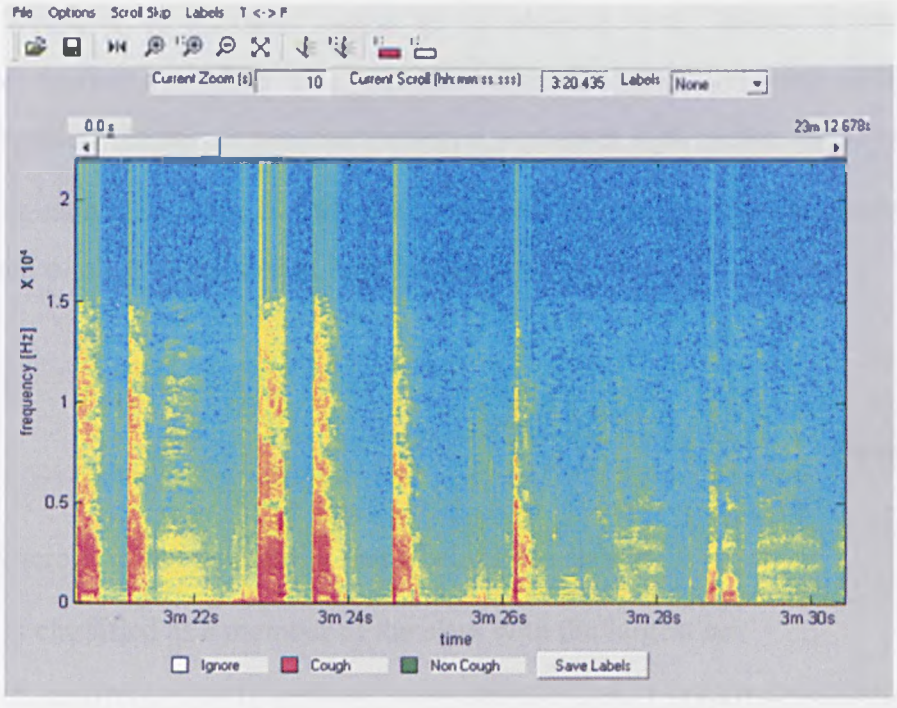


Figure 1.17 The GUI output of the software showing the time/frequency signal plot for a 10 second window of a processed audio recording

1.6.2.1.4 Pattern comparison

The next step was the application of a pattern recognition step to the calculated feature vectors. A PNN is used to perform the pattern comparison and decision-making step. This network provides a general solution to pattern classification problems by following a Bayesian classifiers approach. Bayes theory, invented in 1763 and developed in the 1950's, takes into account the relative likelihood of events and uses *a priori* information to improve prediction. The PNN stores the reference patterns. The action of the PNN for a single test pattern x consists of the following steps:

1. Calculation of all Euclidean distances d between x and the reference patterns.
2. Each distance d_i is transformed into a probability value p_i using a probability density function (also called radial basis function) as illustrated in Equation 1-8.

$$p_i = e^{-(d_i/\sigma)^2} \qquad \textit{Equation 1-8}$$

   Where p_i is maximal (equal to 1) when $d_i = 0$.. When d_i increases p_i decreases according to a Gaussian distribution. σ_{PNN} is the sensitivity or spread. The larger its value the less p_i decreases with increasing distance.

3. For each class k the p_i values are summed to form $(\sum p_i)_k$ and divided by the sum of all probabilities to form Equation 1-9.

$$p_k = \frac{\left(\sum p_i\right)_k}{\sum p_i} \qquad \textit{Equation 1-9}$$

4. Where p_k is the probability that x belongs to class k.
5. x is classified as a member of the class with the largest p_k.

Instead of classifying single patterns, (step 4) the method classifies complete sound events. The p_k values for all test patterns belonging to each sound event are summed

yielding a sum of probabilities $\sum p_k$ for each class. The sound event is recognised as a member of the class with the largest $\sum p_k$.

The PNN required initial training in order to recognise and distinguish coughs from non-coughs using prepared data. To achieve this, a dataset of coughs and non-coughs was required.

1.6.2.2 Method employment

1.6.2.2.1 Data collection

A series of audio recordings containing a relatively large number of cough events were collected in order to develop the cough recognition software. Thirty three smoking subjects, twenty male and thirteen female aged between 20 and 54 with a chronic troublesome cough were studied in the hour after rising using the recording equipment as described in Barry *et al.*[48]. All the subjects were studied in the out-patients clinic with the subjects ambulatory and television and conversation freely permitted. The resultant recordings contained a large number of coughs from the subjects along with a large number of other sounds.

1.6.2.2.2 PNN training

The first step was to identify suitable cough and non-cough events in all 23 recordings, where non-cough events are all sounds present in the audio recording which are not coughs. Sound recordings from 23 subjects were used to create a set of 75 cough patterns and 75 non-cough patterns. Suitability is determined by the clarity of the sound, and by its ability to add relevant variation to the dataset. These events are processed, as previously described, to yield feature vectors which are then combined into a cough pattern matrix X_{cough} and a non-cough pattern matrix $X_{non\text{-}cough}$. The reference patterns used for creation of the PNN are obtained by performing two k-means[117] clusterings ($k = 0.5\ N_{train}$) of approximately 2000 cough and non-cough patterns. The initial 2000 patterns are selected from X_{cough} and $X_{non\text{-}cough}$. The reference patterns are then passed through the PNN to allow future classification of cough and non-cough patterns.

Experimental data is scaled using the means and variances of the reference data and projected onto the principal component space using a projection matrix. The resultant data is then passed through the PNN for classification.

1.6.2.2.3 PNN validation

The remaining ten hour-long recordings of the subjects described in Section 1.6.2.2.1 were analysed by both the software and an independent, experienced cough listener. The GUI described in Section 1.6.2.1.3 was used to count the coughs recognised by the software. The GUI was also used to definitively identify cough and non-cough events in the recordings to establish the sensitivity and specificity of the data processing stage.

1.6.2.2.4 Results

Table 1-2 lists the total number of coughs reported by the human observers and the software. Bland-Altman plots comparing the total number of coughs calculated by both the experienced listener (A), listener (B) and the software are shown in Figures 1.18 and 1.19 respectively.

Table 1-2 Coughs counted in each recording of ten subjects as counted by two listeners (A and B) and the software

SUBJECT	LISTENER A	LISTENER B	SOFTWARE
1	8	6	8
2	21	22	25
3	5	6	9
4	26	25	31
5	14	30	28
6	9	13	9
7	8	8	15
8	20	29	27
9	28	53	50
10	98	150	140

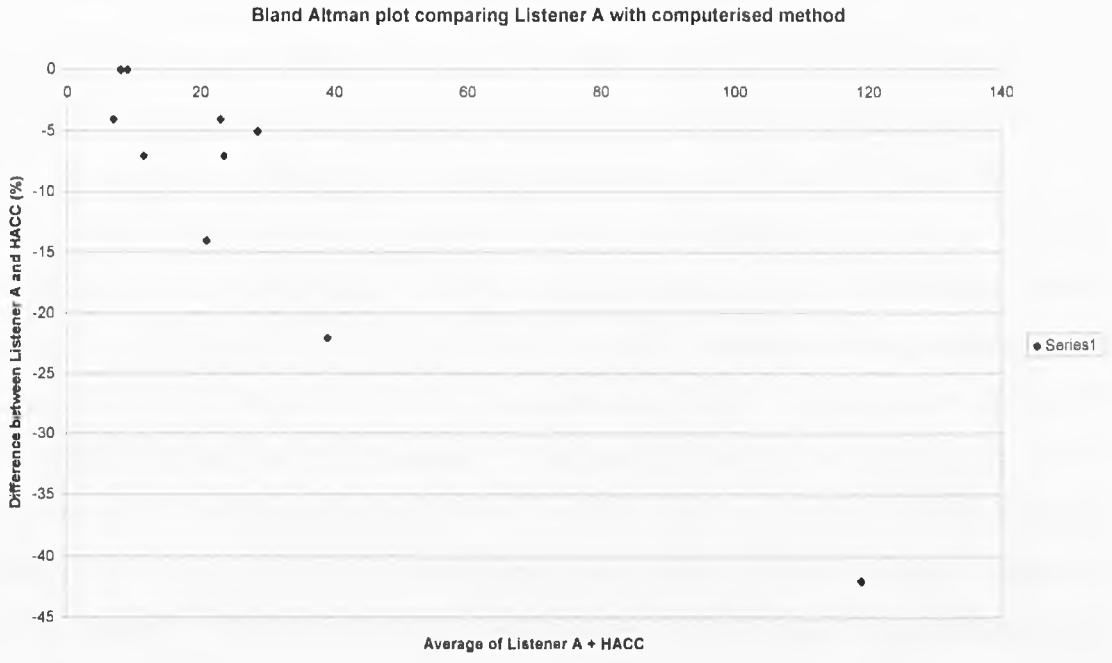


Figure 1.18 Bland Altman plot illustrating the difference between the computerised processing and an experienced cough listener

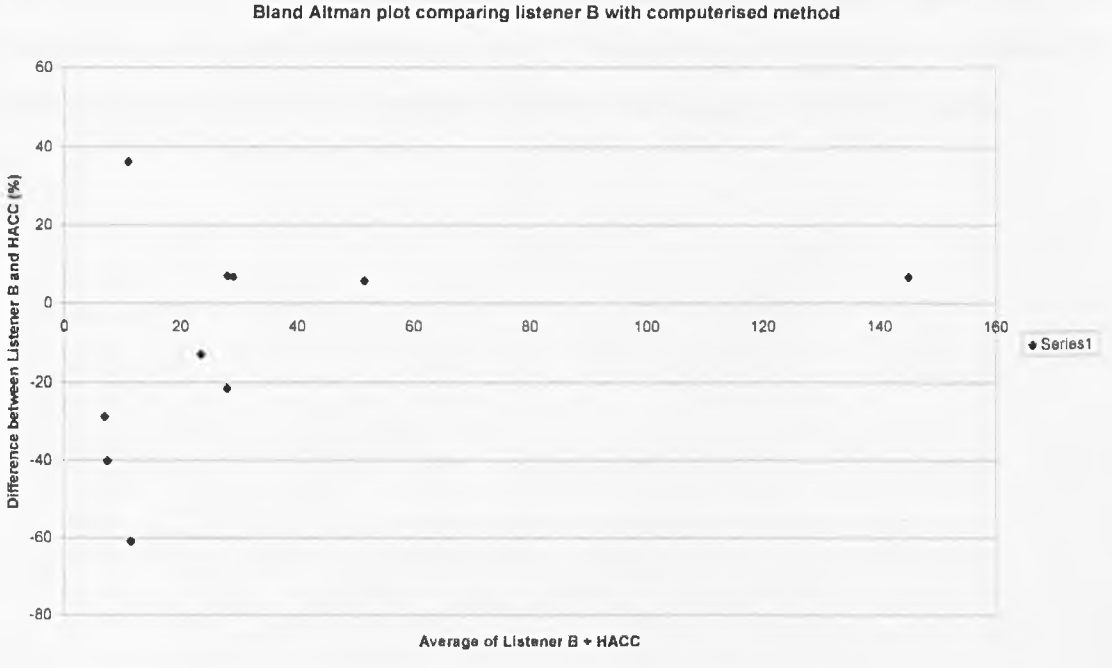


Figure 1.19 Bland Altman plot illustrating the difference between the computerised processing and an inexperienced cough listener

The results showed a significant increase in the number of coughs reported by the inexperienced listener and the computerised method compared to those reported by the experienced listener. This difference is caused by both the inexperienced listener and the software detecting and counting coughs from sources other than the subject under study. The subjects were recorded in a clinic alongside other patients and as a result, other coughs are clearly audible on the recordings. The inexperienced listener B and the software both simply counted all audible coughs which explains why the data from B and C are so similar, and exaggerated. Clearly the experience of listener A discerns between the subject closest to the microphone and the other cough events that are audible on the recordings. Thus it is clear that even with this slight disparity between the computer and the experienced listener, the computer has in fact classified all the coughs on the recordings, but without any distinction as to the source of the coughs.

One of the major advantages of the automated recording is that it is possible to re-analyse the data with minimal effort and achieve consistent results. Thus, when the same recordings were reprocessed the events classified as coughs in one run were also found to be coughs in subsequent runs. This allows development of a statistically stable analysis method with a known statistical confidence limit on the results.

CHAPTER 2

EXPERIMENTAL

2.1 AIMS AND CONTEXT

The aim of this work was to create a 24 hour ambulatory monitoring device, for application to patient cough studies, along with a method for the computerised processing of audio recordings to automatically identify coughs. The starting point for the automated analysis was the existing cough recognition system, as previously described, which required development in order to make it suitable for application to 24 hour cough monitoring and data analysis.

The work was therefore divided into two sections. The first section entailed the design and development of the cough monitoring device, to include all the necessary requirements as discussed in Sections 1.1 and 1.2.2. This was then followed by the validation of its use and the subsequent collection of audio data from 24 hour patient cough studies. The second section involved the development of a computerised method for the analysis of the resultant audio recordings in order to perform cough recognition. Following the assessment of the existing software, it was decided to use this as a starting point, with the intention of further modification in order to apply the method to the new 24 hour cough monitoring studies.

2.1.1 Collation of existing software

The initial stages of the work involved working through the existing software to ascertain the methods used to perform cough recognition on the audio files. The capabilities of the system then needed to be established before the plans for modification could be made.

2.2 DEVELOPMENT OF HACC HARDWARE

2.2.1 Developments for 24 hour recording

The aim of this work was to develop a hardware device capable of making ambulatory, 24 hour recordings for objective cough counting. The requirements for this design were:

1. Recording media capable of storing 24 hours of audio data
2. Portable power supply capable of recording for 24 hours
3. Low sensitivity microphone
4. Patient activity marker button
5. Contained within a portable, convenient and secure case

A design for the device is illustrated in Figure 2.1.

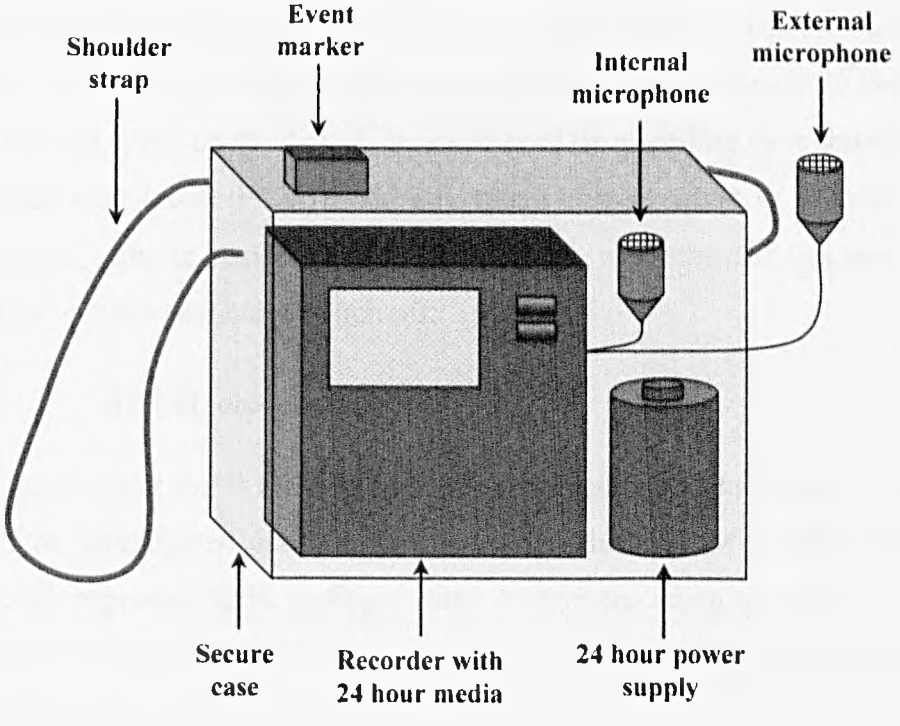


Figure 2.1 Design for a 24 hour ambulatory objective cough counter

2.2.1.1 Recorder

A recorder which uses a high quality recording media capable of storing 24 hours of audio data was required. The chosen recording device was based on a Sony® Hi-MD MZ-RH10 audio walkman, a recent advancement of the original minidisk recorder developed to use a much higher capacity recording media. The audio data is compressed using the ATRAC3plus format (Adaptive TRansform Acoustic Coding). The recorder makes compressed, single-bit analogue-to-digital conversion, digital recordings on a 2.5" magneto-optical high-density MiniDisc (Hi-MD) which is protected in a cartridge. Hi-MDs have the capacity to store up to one gigabyte (Gb) of audio data, which when using the Sony ATRAC3plus mode at 64 kilobits per second (kbps) approximates to 34 hours. The recorder uses a sampling frequency of 44.1 kHz and has a frequency response of 20 to 20,000 Hz ± 3 dB.

2.2.1.1.1 Data storage

Since only 24 hours of data collection is required, the Hi-MDs which are capable of recording for 34 hours, are filled with a non-audio buffer to leave only 24 hours worth of storage space on the disk. This is achieved by recording for a short duration in pulse code modulation (PCM) mode, equivalent to approximately 10 hours in ATRAC mode, without a microphone attached. Thus, when the disk has been filled, the recorder will automatically switch off.

2.2.1.1.2 ATRAC compression

The effects of using the ATRAC compression system to perform audio recordings needed to be investigated and compared with the linear PCM method, which is a digital, non-compressed audio coding system. A short recording of coughs and speech was made in PCM format and then converted to ATRAC. The two resultant WAV files were then compared.

2.2.1.1.3 Signal input

The optimum way to feed the signal from the microphone into the recorder out of the two possible; line-in and microphone-in, needed to be determined. Recordings were

taken using both methods and the resultant signal was then assessed for its signal-to-noise ratio (SNR).

2.2.1.1.3.1 Line input

The line-input port records the received signal with no additional amplification or control. The signal input is 49mV. Following a period of silence, a new track is created which can ultimately lead to thousands of tracks during a 24 hour period. To avoid this problem, a constant tone generator producing a signal of 14 kHz, injected into the second channel during recording, was tested.

2.2.1.1.3.2 Microphone input

The microphone input port amplifies the recorded signal and has more control than the line input over the incoming signal. The signal input is 0.13V. Amplification of the signal can cause a lower SNR and greater interference. To reduce the effects of amplification, attenuators in the form of varying levels of resistors were introduced into the system and the effects calculated.

2.2.1.2 Microphone

A directional microphone with a relatively low sensitivity in order to maximise the detection of cough sounds from the patient whilst minimising extraneous background sounds was required. The chosen microphone is a dual-channel, cardioid condenser microphone with a frequency response of 35-20,000 Hz, a signal to noise ratio of 66dB, open circuit sensitivity of −42dB and a maximum input sound level of 131dB and dynamic range of 102dB. Only the first channel of the microphone is used to record the patient and sits on a strap across the chest. The second channel has the option to be used to record background sounds in order to carry out noise cancellation if necessary, or it can simply be used to serve a purpose inside the device as described in Sections 2.2.1.1.3 and 2.2.1.5.

2.2.1.3 Power supply

An additional power source was required to ensure the recorder and microphone had sufficient power to operate for the 24 hour study period. The supplied nickel metal

hydride (NiMH) rechargeable battery has a lifetime of approximately 8.5 hours when recording in Hi-MD mode which can be increased by 3 hours by using an additional AA alkaline dry battery connected externally. To record for 24 hours, the device was modified to be powered by a rechargeable NiMH 1.2V D cell fed in through the internal battery contacts. The power supply consists of a DC-DC converter which converts the 1.2V supplied by the battery into 5V. This is electrically isolated from the battery by the integrated circuit IC4 [See Appendix B]; this was necessary so as to eliminate severe interference induced into the microphone circuit as a result of a ground loop between the motor drive and the microphone circuits. The microphone derives power from the isolated 5V supply via R8 and R9 [Appendix B].

2.2.1.4 Portable case

All the components of the recording device needed to be enclosed in a portable, convenient and secure case which could be easily accessed by clinical staff for maintenance. The designed case is closed using four tamper-proof screws and two security key locks. The device is fitted with an extendable strap which fits across the patient's body such that it passes from one shoulder to the opposite hip.

2.2.1.5 Event marker

The aim of this research component was to develop a method to enable the correlation of patient activities with coughing episodes. For the hardware development, this required a feature which could record a recognisable signal during the audio recording. In order to maximise the range of activities that can be monitored and to prevent limiting the device to specific applications, the event marker was designed such that it requires the patient to audibly describe their activities. The event marker consists of a button on the exterior surface of the case which can be pressed by the patient prior to the activity description. This generates a one second burst of signal which is attenuated and fed into the second channel of the microphone [See IC1 and IC2 of Appendix B]. In this way, a single event marker button is used to highlight the activity whilst the software will locate the markers and the associated descriptions.

2.3 DEVELOPMENT OF HACC SOFTWARE

2.3.1 Processing software

The original software has previously been described in Section 1.6. Improvements to software were made using Matlab® version 6.5. All software that was written for this system is given in Appendix A and is referred to in the text. All other functions are Matlab functions and can be found in the following toolboxes: PLS_Toolbox version 3.5 (Eigenvector Research Inc., Manson WA), Signal processing toolbox version 6.0 (Mathworks Inc., Natick, MA), HOSA toolbox version 2.0.3 (Developed by A. Swami, downloaded from the Matlab File Exchange) and Neural network toolbox version 4.0.2 (Mathworks Inc., Natick, MA). The programs were executed under Windows XP on a 1.50 GHz Pentium(R)M PC with 256 megabytes of RAM and on a 2.66 GHz Pentium(R) 4 with 512 megabytes of RAM fitted with a Sound Blaster Audigy 2 ZS Platinum Pro Audio Card (Creative Technology Ltd.).

2.3.2 Commercial software

The "design of experiment" work was achieved using the Stat-Ease Design-Expert version 4.0 software (Stat-Ease Inc., Minneapolis, MN, USA). Sony SonicStage version 3.4 (Sony Corporation, Tokyo, Japan), as supplied with Sony audio products, was used to transfer audio recordings to the PC and convert them to WAV format for processing. Creative Wave Studio version 5.0 (Creative Technology Ltd., Singapore) was used for cropping audio files and splitting the channels of the audio recordings.

2.3.3 Data collection

The aim of this section was to collect audio data for developing and testing of the software, in addition to the recordings collected previously and described in Section 1.6.2.2.1. For these studies, patients wore the cough recorder as illustrated in Figure 2.2 and described in Section 2.2.1.4.

Figure 2.2 Illustration of the way in which the cough counting hardware is worn by the patient under study

2.3.3.1 Cough studies

A number of audio recordings were required in order to develop the processing software. The cough recordings were comprised of two groups; short cough studies and 24 hour studies.

For the short cough studies, recordings of chronic cough patients, ambulatory within the hospital grounds, were made. A summary of all the short cough studies and their filenames is given in Table 2-1. Patients partaking in the 24 hour cough studies were set up with the recorder in a clinic appointment and then advised to carry out their normal activities. Patients were asked to remove the device from their person for showering and bathing and to hang the strap over their headboard during the night such that the microphone was closest to their head. A summary of all the 24 hour recording cough studies and their filename prefixes is given in Table 2-2.

Each 24 hour recording has 8 associated files consisting of 3 hour portions. The study set included two chronic cough patients with GORD, one cystic fibrosis patient both on and off cough medication and a further six cystic fibrosis patients, three on and three off the cough suppressant medication. For the description code, CF were patients with a cough due to cystic fibrosis, whilst GORD were patients with GORD

as the cause for their cough. Patients were asked to press the event marker following a bad bout of coughing and to audibly describe the activity being carried out prior to the cough episode. In addition to these studies, audio data collected in Section 1.6.2.2.1 were also used for software development; a summary of these files is given in Table 2-3.

Table 2-1 Summary of cough studies and filenames

COUGH STUDY	FILENAME	MALE / FEMALE	RECORDING DURATION (HH:MM)
C1	19-07-2005	Female	00:30
C2	08-09-2005	Female	01:00
C3	30-09-2005	Female	01:45
C4	13-10-2005	Female	01:27
C5	14-10-2005_1	Female	03:05
C6	14-10-2005_2	Female	01:34
C7	14-10-2005_3	Female	01:57
C8	20-10-2005	Female	02:34

Table 2-2 Summary of 24 hour cough studies and filenames

COUGH STUDY	FILENAME PREFIX	MALE / FEMALE	DESCRIPTION CODE
C9	06-Mar	Female	CF
C10	14-Mar	Male	CF
C11	29-Mar	Female	CF
C12	12-Apr	Female	GORD
C13	03-May	Male	CF
C14	09-May_on	Male	CF
C15	09-May_off	Male	CF
C16	11-May	Male	CF
C17	12-May	Female	CF
C18	13-May	Female	GORD

Table 2-3 Summary of original one hour cough studies and filenames

COUGH STUDY	FILENAME	MALE / FEMALE
A1	JLT25	Male
A2	RMT33	Male
A3	MST39	Male

2.3.3.2 Data processing

The aim of this work was to prepare the data for data processing. Cough recordings were transferred to the PC. 24 hour recordings were divided into 8 tracks of 3 hours each and denoted by a filename suffixed "_part1-8". Shorter recordings are divided appropriately such that portions are no longer than 3 hours. The files are then converted to WAV format to be compatible with Matlab for further processing.

The WAV files consist of two channels; the first channel contains the data collected from the patient microphone and is used for cough analysis. The second channel contains information for the patient activity marker and is used for event monitoring. Matlab reads only the first channel of an audio file; therefore the second channel is isolated and saved as a separate mono WAV file for further processing.

2.3.3.3 Test dataset

In order to create a dataset with a sufficient variety of cough and non-cough sounds, a short audio file was created containing sound events and coughs from different subjects. This control dataset was created by compiling short sections of several cough recordings into one audio file. Sections were between 5 and 8 minutes in duration and were chosen to contain a balance of a sufficient number of coughs and other sounds. A summary of the test dataset is given in Table 2-4.

Table 2-4 Summary of the test dataset audio file

PORTION	ORIGINAL FILENAME	MALE / FEMALE	DURATION (MM:SS)	COUGH COUNT
P1	14-Mar_part1	Male	05:00	40
P2	29-Mar_part1	Female	06:00	14
P3	09-May_off_part1	Male	05:00	3
P4	12-Apr_part1	Female	05:40	13
P5	19-07-2005	Female	05:00	17
P6	14-10-2005_2	Female	05:00	118
P7	11-May_part1	Male	08:00	3
P8	13-10-2005	Female	05:30	75

2.3.3.4 Test sounds

In order to identify spectral features of various sounds, individual sound events from the file C1 were extracted for further study. These sound events are summarised in Table 2-5 and include coughs, various upper airway sounds and equipment sounds.

Table 2-5 Summary of individual sound events isolated from the file C1

SOUND	SOUND TYPE
T1 - T58	Coughs
T59 - 65	Speech
T66	Laughter
T67 - 86	Other sounds e.g. equipment sounds
T87 - 91	Breathing

2.3.4 Sound Identification

2.3.4.1 Cough sound acoustics

The aim of this work was to study the characteristic features of various sounds in order to find a method for distinguishing cough from all other sounds. Sounds from the cough studies A1 and C1 were used for this purpose. Time-amplitude plots,

spectrograms and power spectra of cough sounds, vocal sounds and other noises were produced and studied. The time-amplitude plots were produced by plotting the signal against time, the Matlab function *specgram* was used to produce the spectrograms, whilst the power spectra were produced by plotting the square of the absolute value of the FFT coefficients.

2.3.4.2 Development of data pre-treatment

2.3.4.2.1 Identification of sound events

The first step toward data pre-treatment was to identify valid sound events within the cough recordings and then correctly label them as coughs or non-coughs for the subsequent training and validation step [as described next in Section 2.3.4.2.2]. Audio recordings were initially processed by the function *coughgui* which locates portions of sound in the recording [as described in Section 1.6.2.1.1] and creates a GUI to display the information. Using the GUI, representative cough sounds were manually selected and labelled to be used for training the neural networks. Suitability is determined by the clarity of the sound and by its ability to add relevant variation to the dataset. All other sounds are classed as non-coughs.

The threshold levels for determining the validity of sound events were set depending on the individual recording and were determined by use of the SNR.

2.3.4.2.2 Training and validation

The next step was to create a function which automatically divides the data, previously labelled, into sets for training and validation for the neural network. The function created was *dataprep* [See Appendix A]. Cough sounds were initially separated from non-cough sounds to ensure a fair distribution of each class into both training and validation datasets. One third of the data was selected for validation by taking every third point whilst the remaining two thirds were used for training. If there are more non-cough sounds than cough sounds, the non-cough dataset is reduced so that they are almost equal. A separate matrix was created to contain labelling information for the data, where '1' represented a cough and '0' represented a non-cough.

A random selection of sound events from the cough studies A1 and C1 were listened to following the dataset division to validate this classification process.

2.3.4.2.3 Feature extraction

It was considered appropriate to use the existing software as a starting point for this work. Therefore the feature extraction stage was based on the initial method as described in Section 1.6.2.1.2 with further modifications included where necessary.

2.3.4.2.3.1 Mel-frequency cepstral coefficients (MFCC)

The optimum window size for the MFCC calculation needed to be ascertained. The window size was initially set to 256 samples. This essentially divides each sound event into 0.023 second portions and lists the MFCC's calculated for each window. In order to capture all the MFCC's into a one-row information matrix, the minimum window size was first established by determining the size of a random selection of coughs. Following this the MFCC window was set to 4864 samples and any sound events smaller than this were padded with zeros so that all outputs were of uniform size.

2.3.4.2.3.2 Linear predictive coding (LPC)

The value of the LPC coefficients in addition to the MFCC's, as had previously been carried out in the existing method, was ascertained. LPC coefficients were originally calculated and added onto the end of the MFCC matrix such that the first 42 spectral coefficients were from the MFCC calculation and the remaining 14 were from the LPC calculation. A study of these coefficients showed that they possessed very little variation between coughs and non-coughs, and inclusion of these was considered unnecessary.

2.3.4.2.4 Cough in audio recordings

It was considered necessary to assess the effect that the proportion of coughs contained within a recording has on the ability of any system to accurately identify the cough events, given a system efficiency value. The assessment was carried out using a contingency table which is based on Bayesian statistics and takes into account

the system prediction efficiency and the proportion of coughs to other sound events to give a cough recognition accuracy.

2.3.4.2.5 Principal component analysis (PCA)

The aim of this research was to perform data reduction on the spectral coefficient data set so that only the data containing maximum variation remained. Following the application of MFCC to the data, the number of columns in the resultant spectral coefficients was 42, consisting of 14 coefficients along with 14 delta coefficients and 14 delta-delta coefficients; whereas the number of rows was variable and directly dependent upon the duration of the sound event and the window size employed by the MFCC function. However, the resultant data matrices for each sound event must be of uniform size for subsequent presentation to the neural network.

PCA was carried out on the spectral coefficients using the singular value decomposition (SVD) function to produce the eigenvectors and corresponding eigenvalues. A number of different methods for the application of PCA to the data were tried.

2.3.4.2.5.1 Reduction of the loadings

An eigenvalue plot was produced for the spectral coefficients of both coughs and non-coughs following PCA in order to establish the amount of variance possessed by each coefficient. The loadings that appeared to possess little variance were then removed from further processing.

2.3.4.2.5.2 Data reduction to multiple rows

Following on from the previous step using the PCs with the most variation, the scores were reduced to create a uniform dataset size according to the number of frames in the smallest sound event.

2.3.4.2.5.3 Data reduction to single row

To capture maximum variance and information in one row of data, such that all relevant data for each sound event can be passed to the neural network together, the PCs with the maximum variance were ordered on one row. SVD was performed on

each sound event and the first PCs were multiplied by their corresponding eigenvalues to scale according to variance.

2.3.4.2.5.4 Data reduction to uniform size

Additionally, prior to SVD, the feature vector was multiplied by its transpose to create squared matrices of equal size for all the sound events. SVD was then carried out and multiplied by the corresponding eigenvalues. The square root of the data was taken and the first 2 to 5 loadings were lined up from left to right to form one row of data. Scores were also reduced in both cases according to the percentage variance they represented.

2.3.4.2.6 Correlation coefficients

It was considered necessary to investigate if there was any correlation within the frequency spectrum as the frequencies progressed through the sound event.

The correlation coefficients of the first 15 spectral coefficients were calculated and compiled into a one row matrix. A reduced number of coefficients were used so as to reduce the size of the final dataset; the first 15 coefficients appear to possess the most variance of all 42. The training data was then constructed in the usual way, with each row this time representing one sound event. PCA was carried out and the scores plot produced. In addition, the correlation coefficients were plotted onto coloured surface plots to make any correlation visually apparent.

2.3.4.2.7 Higher order spectral analysis (HOSA)

2.3.4.2.7.1 Voice data collection

In order to study the variation of vocal sounds between males and females, audio data was collected. Recordings were made of a dialogue as spoken by two males, two females and a male and a female. The dialogue was used to introduce a variety of words and speech characteristics whilst maintaining a similarity between the different recordings.

Further to this, a recording was made of a male and female speaking a variety of vowel sounds to enable study of the speech formants if necessary.

2.3.4.2.7.2 Voice separation

HOSA was applied to samples of speech from both males and females to attempt to separate different voices. Various functions from the Matlab HOSA toolbox were employed [See Appendix A for methods]. The outputs from HOSA were inspected to identify any features that could potentially be used for voice separation.

2.3.4.2.7.3 Cough separation

HOSA was applied to the cough and non-cough data to attempt to identify significant differences between them in order to enable separation. HOSA was carried out on a test sample of sound events consisting of eight coughs and eight other sounds as isolated in Section 2.3.3.4. Script code for these calculations can be found in Appendix A. Values for the mean, variance, skewness and kurtosis, the linearity and Gaussianity data and the three fundamental frequencies of the sound were all calculated and compiled. A LVQ network with 13 input neurons was created, one for each value, with 2, 4 and 6 hidden neurons. Following training with this data, the same data was presented to the network for simulation.

2.3.5 Optimum data pre-treatment

Using the cough studies C5-C7, the data pre-treatment techniques described previously were assessed for their performance. Following this, an optimum set of data pre-treatment steps were compiled and used to treat the data prior to the application of the ANN.

2.3.6 Sound Separation

The aim of this section was to develop a series of ANNs in order to optimise the pattern recognition step and allow the recognition of cough events from all other sounds.

2.3.6.1 Development of neural network

2.3.6.1.1 Feed-forward (FF) neural network

A FF neural network with one hidden layer was created for later training with pre-treated data from the cough studies. Several parameters of a FF neural network were tested which resulted in several versions of the network. Table 2-6 lists the parameters and the levels at which they were tested.

The number of input nodes in the input layer was variable and dependent upon the pre-treatment the data had undergone. The output layer had one output node which gave values of closer to '0' for a non-cough and closer to '1' for a cough.

Table 2-6 Parameters of the FF neural network and the levels at which they were tested

PARAMETERS	LEVELS TESTED
Hidden neurons	8, 10, 12, 15
Training function	Sequential order incremental training function
Learning method	Gradient descent method
Performance function	Mean squared errors
First transfer function	Competitive, tansig, logsig
Second transfer function	Purelin, tansig, logsig

2.3.6.1.2 Learning vector quantisation (LVQ) neural network

A LVQ neural network was created for later training with pre-treated data from the cough studies. The LVQ network was created with two neurons in the hidden layer. The number of input nodes in the input layer was variable and dependent upon the pre-treatment the data had undergone. The output layer had two output nodes which gave a value of 1 in the first column to represent a cough and a value of 1 in the second column to represent a non-cough.

2.3.6.1.2.1 Testing the neural networks

The aim of this experiment was to test the ability of the designed neural network to distinguish between two groups of data by use of a simulated dataset. Two pure spectral peaks (approximate Gaussian curves) were plotted using 41 data points (c.f. 42 coefficients for the mel cepst signal processing coefficients). The first peak had its maximum at 10 (arbitrary units) and a range of 0-20 while the second peak had its maximum at 30 and a range of 20-40 [See Figure 2.3]

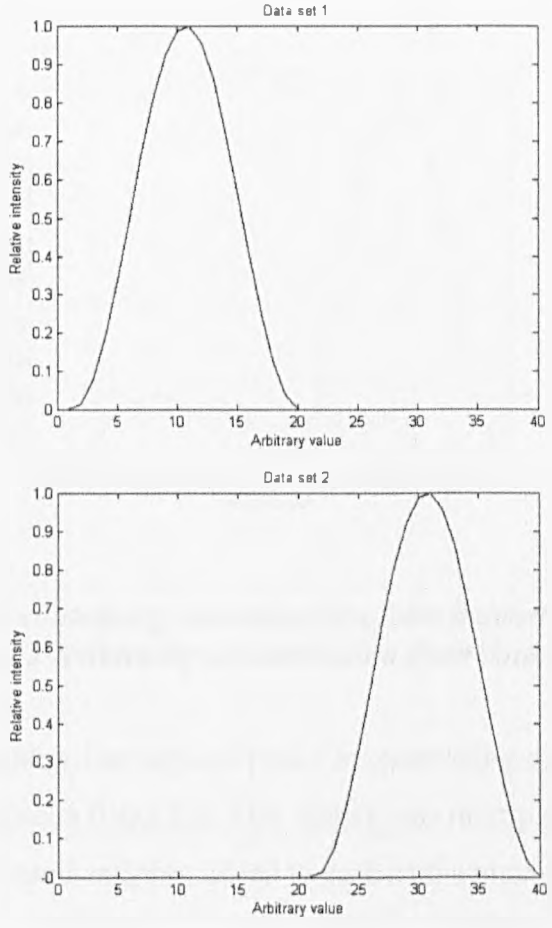


Figure 2.3 Two spectral peaks with peak maxima at 10 for data set 1 (top) and 30 for data set 2 (bottom)

To add some simulated contamination into the dataset, the pure Gaussian peaks were multiplied by 0.2 to give minor peaks. These were added to the major peaks to create two new spectra consisting of the major of one peak and the minor of the other [See Figure 2.4].

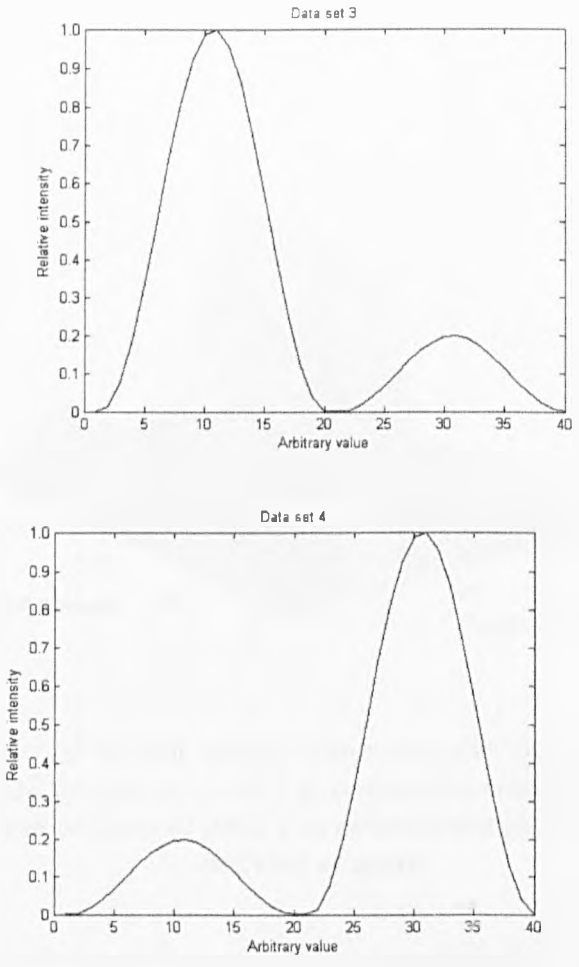


Figure 2.4 Dataset 1 containing contamination from dataset 2 (top) and vice versa with dataset 2 containing contamination from dataset 1 (bottom)

Noise was added to these four sets of peaks by generating a random matrix of size (1,41) of numbers between 0 and 1.0. This matrix was multiplied by 0.02 to make the numbers sufficiently small and then added to each of the pure spectra.

A total of 400 spectra were then generated, 100 of each of the noisy spectra and each with slightly different values. This was achieved by creating a matrix of random numbers between 0 and 0.1 of size (1,100) and multiplying each number by the noisy spectra of each peak [See Figure 2.5].

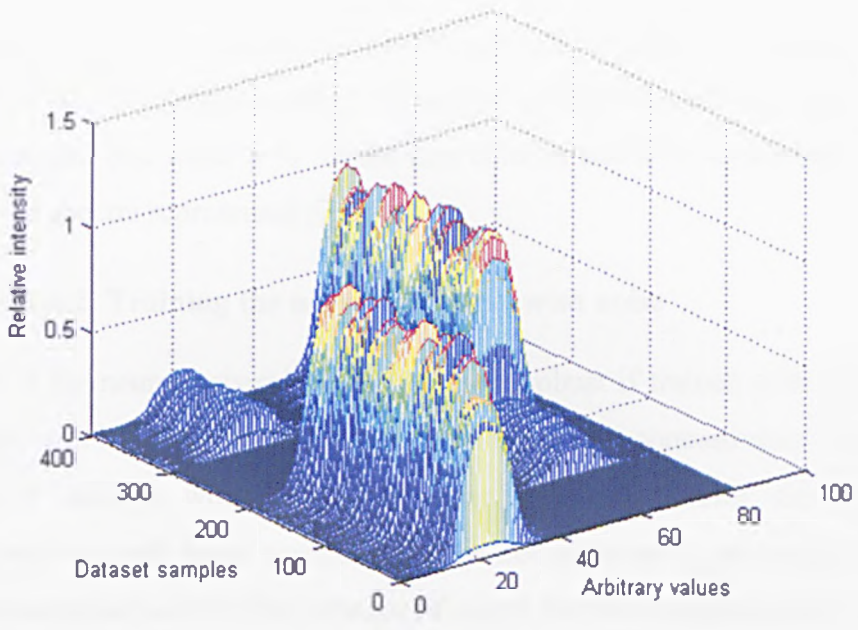


Figure 2.5 Mesh plot of the 400 spectra comprising the dataset: 100 of peak 1, 100 of peak 1 plus a small amount of peak 2 as contamination, 100 of peak 2 and 100 of peak 2 and a small amount of peak 1 as contamination. All the peaks had the addition of noise.

In order to correctly train and validate the neural network, the dataset was split into two groups; 100 validation spectra (sampled at regular intervals throughout the dataset to get an even spread of the spectra) and 300 training spectra.

PCA was applied to the training data set in the form of SVD. The first five PCs were then used to represent the variation in the dataset. The first five loadings values were multiplied by the training spectra to give the dataset necessary to train the neural network. The validation spectra were treated in exactly the same way by being multiplied by the same values.

A LVQ neural network was then created. Five input neurons were created for the five PCs and three neurons were specified for the hidden layer. The training function was defined as *trains*, which selects samples from the training dataset in sequence in order to train the network. All these parameters are as defined in the network used for the original dataset.

The target classes were defined as simply '1' or '2' for the first and second peaks respectively. A binary matrix was created to pass these targets to the network such that a '1' in the first column and a '0' in the second column indicated the spectra represented peak one while a '0' in the first column and a '1' in the second column indicated the spectra represented peak two.

2.3.6.1.2.2 Training the neural networks with noise

To decide if the neural network would be more robust if trained with noisy data, it was trained with both the original data and the same data contaminated with artificial noise. A FF network with one hidden layer and a logarithmic sinusoid transfer function between each layer was created. The neural network was trained using the gradient descent method for 1000 epochs of cough and non-cough data as prepared by the function *dataprep*. This was followed by another 100 epochs of the same cough and non-cough data, modified by the addition of random noise. A further 1000 epochs of cough and non-cough data, without added noise, completed the training.

2.3.6.1.2.3 Testing the neural network's suitability to the data

The aim of this experiment was to test the suitability of the neural network to the data requiring classification. The cough and non-cough spectral features of 20 sounds taken from the test sounds group were separated into two datasets. Each value in the cough dataset was multiplied by various factors from 10 to 100,000 so that it was substantially larger than the non-cough data. The performance of the neural network was then tested at the original scale and at the exaggerated values. A LVQ network was employed with default properties as set by the Matlab function *newlvq* and three neurons in its one hidden layer.

Following on from the results of this previous test, the same spectral features were then used and the magnitude effects were tested on the non-cough dataset.

The results appeared to indicate that the network was not recognising magnitude between groups; a simulated dataset was therefore produced, each data vector possessing the same basic pattern but with two different magnitudes in the dataset.

Finally, the real dataset was pre-treated to leave only absolute values and only the first PC was used. The neural network was re-tested with this new dataset, still with the non-cough data scaled up to the same orders of magnitude.

2.3.6.1.2.4 Number of epochs

The number of epochs used for presentation of the data to the neural network was tested at varying levels from 20 to over 1000. The optimum number of epochs to prevent over-training was then established.

2.3.6.1.2.5 Optimising network training using design of experiment

The aim of this experiment was to determine the optimum network training parameters. Design of experiment (DoE) was used to investigate the optimum combination of the number of PCs used to represent the data and the number of neurons in the hidden layer of a network. Table 2-7 lists the levels tested for each of the factors under study.

Table 2-7 Factors under study and the levels used for each in the optimisation of the neural network training parameters

FACTORS	LEVELS
Number of PCs	3, 4, 5, 6
Number of hidden neurons	4, 6, 8, 10, 12, 14, 16, 18, 20

The data consisted of cough and non-cough feature vectors from the audio file C5 which had been squared, decomposed by SVD, scaled by multiplying each eigenvector by the variance each represented and then square rooted. The first rows of the resultant matrix, according to the number of PCs being tested, were lined up from left to right to form a single row. The response to be determined was the cough validation accuracy achieved by the neural network.

The neural network was a LVQ network as described in Section 2.3.6.1.2.

2.3.6.2 Network training

In order to create an automated system, software was created to perform complete, automatic ANN training and validation using the pre-treated data. Script codes, *lvqnet* and *ffnet*, for the automatic training of the neural networks can be found in Appendix A. The automatic process firstly takes the name of the file to be used for training, separates the data into training and validation and applies the desired data pre-treatment [See Section 2.3.4.2]. If there is no existing ANN, a new one is created; otherwise an existing one is recalled. The prepared data is then presented to the neural network for training. After each set of training epochs, the validation data is presented to the network. The classifications given by the ANN for the validated data are compared to the actual values and a validation accuracy is subsequently calculated. For accuracy, three validation accuracies are determined; the percentage of coughs and non-coughs correctly classified and a total percentage of sound events correctly identified. Each validation accuracy is then compared to the one given on the previous run; unless it is the first run in which case the network is trained once more. If the accuracies are equal to or better than the previous, the network is assumed to have either reached a plateau or is still improving and the current network is saved for further training. Another cycle of training then ensues. However, if the accuracies have worsened, then the network is in danger of being over-trained and so the current training changes are ignored and the previous network saved as the final version.

Table 2-8 Neural network training data details

Audio File	Cough Count	Non-cough Count	Epochs
A1	32	28	25
A2	19	14	50
A3	11	6	50
C1	20	29	50
C4	140	37	50

2.3.6.3 Network simulation

Continuing with system automation, software was developed to automatically apply the pre-treatment stage to a previously unseen audio recording and then use the saved

neural network for classification of the sound events. This function created is *netprocess* as given in Appendix A. This file automates the process of presenting a new file to a previously trained neural network for classification. The audio recording is initially processed by the function *coughgui* to identify portions of sound in the recording [as described in Section 1.6.2.1.1 and Figure 1.15]. Data pre-treatment is then applied exactly as for the network training data to extract the spectral coefficients representative of the sound events. The current ANN is recalled and the resulting data is presented to it. The outputs are then classified as coughs or non-coughs dependent upon their values.

2.3.6.4 Analysis of audio recordings

The next stage was to apply the developed software to sections of the 24 hour audio recordings collected as described in Section 2.3.3.1 and listed in Table 2-2. The neural network was created and trained according to Section 2.3.6.2 and Table 2-8. The audio recordings C9_part1, C10_part1, C10_part2, C11_part1 and C12_part1 were pre-treated as described in Section 2.3.5 and then processed using the neural network.

During this work, it was considered necessary to develop the ANN training step. Experiments into various training conditions were made. It was also attempted to retrain the network using some of the interfering sounds encountered on these recordings. The GUI was used to scroll through the audio recordings and to allow identification of a selection of non-cough sounds that were repeatedly being wrongly classified as coughs and an equal selection of coughs. The network was then trained with the isolated sounds and their correct classes.

2.3.7 Additional data pre-treatment

From the results of the work carried out in the previous section, it was considered necessary to find an alternative approach to the data processing in order to achieve cough recognition. The aim of this section was to further investigate the possibilities of data pre-treatment in order to obtain data that the neural network could distinguish with greater accuracy.

2.3.7.1 Frequency filtering

The aim of this experiment was to use the information gathered from Section 2.3.4.1 to exclude any unwanted frequency information from the audio recordings and thus enhance the data pre-treatment stage.

2.3.7.1.1 Application of a Butterworth filter

A narrow-band high-pass frequency filter was used to eliminate the low frequencies present in many interfering sounds, whilst retaining the higher frequencies possessed by coughs. A Butterworth filter [See Section 1.3.3.4] with a pass band beginning at 14700 Hz and stopping at 15300 Hz was applied to each minute of the audio file C1. The remaining signal was then analysed for its high frequency content. The presence of a frequency with a threshold intensity of over 0.0002 was counted as a positive event and a list of occurrences in the file was compiled. Following the location of a positive event, the file was skipped forward by 0.3 seconds to avoid the multiple classification of a single sound event. Several of the parameters were tested at varying levels, these are summarised in Table 2-9.

Table 2-9 Summary of parameters tested for the frequency filter and their values

PARAMETER	LEVELS TESTED
Skip value	0.1, 0.2, 0.4
Filter frequency (kHz)	11, 12, 14, 15
Threshold level	0.0001, 0.0002, 0.002, 0.007, 0.02

The start time of the sound event is defined as the point that the high frequencies defined by the filter rise above the threshold intensity. The end point is defined by taking the average intensity of each 100 sample portion from the start of the event until it falls below a 0.0001 threshold. The end of the final portion still above the intensity threshold is defined as the end of the event.

Sounds that are classified multiple times due to having above-threshold frequency intensities 0.3 seconds into the sound are identified by having the same end times.

These sound events are combined, taking the earliest start time as the actual start time.

To eliminate some high frequency microphone noises or sector boundary errors (SBEs), these sounds were analysed and found to have very short durations. A check was included for sound event duration and any found to be lower than 2000 samples (0.18 seconds) were discarded.

2.3.7.1.2 Application of FFT

The audio recording was initially processed by the function *coughgui* to locate portions of sound in the recording [as described in Section 1.6.2.1.1 and Figure 1.15]. FFT was then applied to each sound event and the Fourier transforms between 8,000 and 15,500 Hz were taken. To identify similarities or differences between the data, four cough sounds and six other sounds, selected from the sound events isolated as described in Section 1.6.2.1.1, were all squared by their transpose and then processed using SVD. All scores were plotted on the same space. The frequency band selected for analysis was also tested at 6,000 to 15,000 Hz.

2.3.7.1.3 Covariance

From the results of the previous experiment, it was considered necessary to further investigate the variance within the data following frequency filtering. A covariance calculation was performed on both groups of sounds. The Matlab function *cov* was used to calculate the covariance of test sounds as selected from Table 2-5. Following the results of this, the FFT filtering step and the covariance calculation was applied to all the sound events in the file C1.

The results of this allowed a threshold value to be established for the covariance level of cough events.

2.3.7.1.4 Validation of frequency filtering

The aim of this experiment was to test the performance of the frequency filtering data pre-treatment on the test dataset [Section 2.3.3.3]. The functions *fft_process*, *fft_features* and *fft_coughid* were created to automatically process the data in this way

[See Appendix A]. Frequency filtering was carried out as described above in Section 2.3.7.1.2 above followed by the covariance calculation and the application of a covariance threshold.

A minimum cough duration was set to exclude sound events that were too small to be coughs from the results.

2.3.7.1.5 Application of an ANN

It was considered useful to test out the ANN on the sound events defined by the previous stage. Results obtained for the testset were divided equally into training and validation groups. A LVQ neural network was then created, trained and validated.

2.3.8 Additional features

2.3.8.1 Graphical representation of results

The aim of this work was to create a function to automatically summarise the results of the cough counting and represent them graphically. The function *plotcough* [See Appendix A] summarises the cough frequencies for each of the eight three-hour portions in a 24 hour audio recording into 15 minute sections. The results are then plotted on bar charts in two forms for ease of data interpretation, the cough count and the time spent coughing. The function *combineplot* then groups the information for all eight plots to produce a 24 hour cough count summary. The method was validated using the cough studies C9 and C10.

2.3.8.2 Double cough detection

The aim of this research was to solve the problem of the software incorrectly recognising multiple coughs as single events. To avoid double or triple coughs being miscounted a step was included to allow the operator to make the final decision. The function *plotcough* [See Appendix A] takes all sounds events over one second in duration and plays them to the operator in sequence. After each sound event, the program takes the operators decision, inputted simply as a number, before progressing to the next sound event. The operator can request repeated playback at

any time. At the end of the sequence, the number of cough events is recalculated and displayed graphically.

2.3.9 Patient activity monitoring

The aim of this research was to develop the software to be used with the event marker. The software was required to locate the use of the event marker on the second channel of the audio recording and identify the corresponding patient description on the first channel. Compilation of the activity information is to be carried out by the operator.

2.3.9.1 Spectral analysis

In order to identify the occurrence of the event marker within the audio recording, a characteristic feature of the marker needed to be identified. The marker sound was studied for its spectrographic properties. A portion of the audio signal from the second channel of the file C10 which contained a marker sound, as determined by listening to the signal, was isolated. The FFT was calculated and the resulting spectrogram plotted using the function *specgram*. Studying the frequency intensities revealed the frequency content of the signal for further analysis.

2.3.9.1.1 Application of frequency filter

Using the results of the previous experiment, the aim was to design a function to automatically process an audio recording and identify the use of the activity marker. The function *signal* was created [See Appendix A]. Each minute of the 3 hour audio file is processed in sequence and is initially filtered. A narrow band pass Butterworth filter removes all frequencies outside of a 14,300 to 14,900 Hz range. The *specgram* function is then used to calculate the amount of each frequency within this filter range. The resulting output is a measure of the intensity of this marker frequency in each sample. Thus, samples exceeding a threshold of 0.003 of the relative amplitude determine the location of the marker. Marker times are compiled and converted from sample numbers to time for output.

2.3.9.1.2 Validation of "signal" function

After development, the *signal* function required validation to determine its effectiveness at locating the use of the event marker. The second channel of the audio recording C10 was listened to and a summary of the start times of the marker was made. These times were then compared to the results from the processing of the same recording by *signal*.

2.3.9.1.3 Compilation of patient activities

Following the success of the previous experiments in locating the use of the event marker, the aim of this experiment was to establish the activity description on the first channel of the audio recording. The function *activity* [See Appendix A] uses the marker times compiled by *signal* to locate the corresponding portion on the first channel of an audio recording which contains the patient's description of activity. The section of recording beginning at the start of the marker and ending two seconds after the end of the marker is played to the operator using the Matlab function *soundsc*. The operator inputs the activity as spoken by the patient and the function compiles a summary of the activities and the times they occurred.

CHAPTER 3

RESULTS & DISCUSSION

3.1 AIMS

The aim of this work was to create a 24 hour ambulatory monitoring device along with a method for the computerised processing of audio recordings to automatically identify coughs.

- Monitoring should be carried out for 24 hours in order to study the temporal / diurnal patterns of cough frequency.
- The monitoring device should also be an objective measure such as audio recordings to avoid the inaccuracies introduced into the system by subjective counting.
- The recording device should be ambulatory to allow the patient to carry out normal daily activities in their own environment in order to gain a true representation of the cough events.
- Finally, an automatic method for the analysis of the recording is required to make the system feasible and convenient to use.

3.1.1 Assessment of existing system

In the assessment of the existing methodology, data was reprocessed to establish the methods in which the system worked and also to obtain a measure of the capabilities of the system. Using the GUI, the average sensitivity was calculated to be 0.80 with a range of 0.55 to 1.00 while the specificity was 0.96 with a range of 0.92 to 0.98. At this stage, the cough events recognised needed to be counted by a human listener. For this purpose, the GUI was used. However, in this procedure only events classified as cough have to be listened to. Using the system it was possible to identify coughs in an hour long recording in an average time of 1 minute 35 seconds, a reduction of 97.5% in counting time.

Reproducibility of repeated analysis is 100%. The average percentage of false positives compared to true positives was calculated to be 20%. False positives were caused by similar sounds such as laughter, loud bangs and other subjects coughing.

The results of this initial work therefore showed the potential of automatic cough identification using speech processing techniques and neural networks. However some limitations have been highlighted at this stage:

1. Due to the current hardware limitations, the system is only capable of short recordings, in a relatively controlled environment and with subjects only ambulatory to a limited degree. This is an obvious limitation with regard to the need for 24 hour ambulatory and objective cough counting but also means that this developed software has not been fully tested in such conditions. Adequate hardware needs to be developed in order to carry out such recordings and then enable the development of the software if necessary.
2. The study showed that the method is not subject-specific in its cough classification, and will recognise any coughs that are audible on a given recording. Improving the counting accuracy is best achieved, therefore, by excluding the non-subject coughs from the recording, for instance by using a different microphone with a lower sensitivity. This will ensure only high-amplitude sounds occurring close to the microphone will be detected, thus discerning the subject's coughs from ambient coughs. This modification will also diminish problems with the increased background noise which will be encountered in 24 hour ambulatory recordings.
3. The original use of the PNN meant that all training had to be carried out at once which did not allow for any additional training later on. It would be advantageous to have the ability to train the network at a later stage as different cough types and other sounds are encountered. For this, an automatic data-pre-treatment, data selection and network training would be necessary.
4. The recordings were of smoking subjects and thus contained a large number of coughs in a short time period. The subjects were also confined to the clinical environment with a limited amount of ambient noise to be detected on the audio recording. Thus, a sample of all background noise and cough samples could be used to train the PNN, with minimal possibility of further extraneous sounds causing interference.

In order to fulfil the original aims of creating a 24 hour ambulatory recording device together with software capable of analysing the resultant data, several improvements to the existing system were required.

In addition to providing solutions to the problems of subject non-specificity and limited neural network training, the hardware must also be re-designed and the software must be modified to be suitable for processing the extended recordings.

3.2 Development of Hardware

The hardware used for the initial studies was a lapel microphone connected to a DAT recorder, which had limited recording duration due to power constraints and a limited amount of data storage. To meet the new requirements, a new recording system had to be developed.

3.2.1 Developments for 24 hour recording

The aim of this work was to develop a hardware device capable of making ambulatory, 24 hour recordings for objective cough counting. Research over the years has shown the need for such technology [See Section 1.2.2]. Subjective scoring of cough has frequently been proven to be unreliable. Animal models and induced-cough models have been shown to be inaccurate for assessing patient cough responses and therapeutic efficacies. 24 hour recording durations are necessary to study the diurnal changes of cough and an ambulatory device is required to study the patient during normal everyday activities and in their own environment.

The ability to carry out 24 hour, ambulatory recordings depended on the combination of several requirements:

1. Recording media capable of storing 24 hours of audio data
2. Power supply capable of recording for 24 hours
3. Low sensitivity microphone
4. Patient activity marker button
5. Contained within a portable, convenient and secure case

3.2.1.1 Recorder

There are a limited number of commercially available recorders that have the capability to carry out 24 hour ambulatory recordings. For the cough counting device, the most important features are adequate data storage, as it would be unfeasible to interrupt a study to change the recording media, and a compact and portable design for maximum convenience to the patient. A 24 hour power supply would be beneficial but not absolutely necessary as extra power could be provided with slight modifications.

Three possible options were MiniDisc (MD) recorders, solid state memory recording devices, also known as flash memory and hard disk drive (HDD) recorders.

MD technology was announced by Sony in 1991 and introduced in 1992, and is capable of storing any kind of binary data. Originally intended to replace the compact disk (CD) for sale of music albums, MDs are now primarily used for recording and are fitted with a USB interface which can be used to upload recordings to a computer.

The audio on a MD is compressed using the ATRAC format. ATRAC is a psychoacoustic lossy audio compression scheme, which means that although decompression of the compressed signal will not yield the original signal, it is designed to maintain a high quality of sound, audibly identical to the original.

MDs were recently improved by the development of high-density MDs (Hi-MDs) and Sony's latest compression, ATRAC3plus. Hi-MDs are capable of storing up to one gigabyte (Gb) of audio data, which is approximately 34 hours when using ATRAC3plus mode of recording at 64 kilobits per second (kbps).

The recording media is therefore adequate for 24 hour recordings, although the mechanical action of the disk means power consumption is relatively high and the supplied battery was not capable of long recording durations. At this time, flash memory devices and HDD recorders were still very much in their infancy, very expensive and not widely used. In addition, although one of their main advantages is large data capacity, the amount of storage they offer is actually superfluous to the design requirements. While this could be viewed as an opportunity to utilise the much larger memory capabilities in order to store the required 24 hours in a less

compressed format, again this is an unnecessary extra. Therefore, due to the features and capabilities of the Hi-MD recorders, the Sony Hi-MD MZ-RH10 was selected to form the basis of the 24 hour recording device.

3.2.1.1.1 Data storage

The recording media were filled with a non-audio buffer of a size equivalent to approximately 10 hours of ATRAC3plus compression such that 24 hours worth of space remained on the disk. This successfully limited the recording time to the desired 24 hours. This served an important purpose in that when the disk reaches full capacity, the recorder automatically performs the data-save function and stops recording; whereas if the recorder were to continue for an unnecessary 34 hours, power supply would be compromised and a power failure would result in the entire recording being lost.

3.2.1.1.2 ATRAC compression

Due to the high degree of compression being used in order to record for the 24 hour period, it needed to be ascertained that the ATRAC3plus method of compression did not significantly affect the recording compared to the ordinary PCM. The quality of data recorded in both Linear PCM and ATRAC formats were therefore directly compared. Spectrograms of the two examples are shown in Figure 3.1. The colours represent frequency intensities; comparison of the two plots suggests that there is no reduction in intensities of the frequencies recorded in the ATRAC format. The two plots are also very similar in clarity and contrast, indicating that spectral resolution also appears to be maintained. This comparison shows that the ATRAC method of compression is very similar to the PCM recording and will therefore produce recordings of equally high quality.

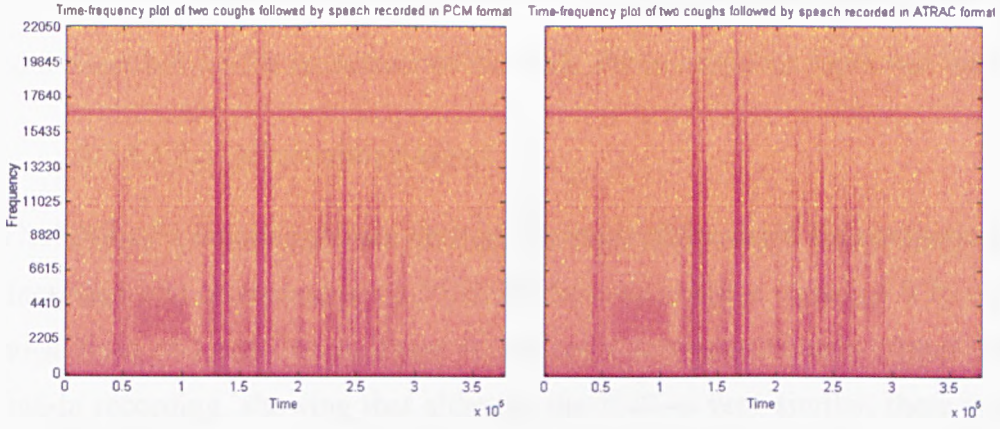


Figure 3.1 Spectrogram of a sample of coughs and speech recorded in PCM format (left) and ATRAC format (right)

3.2.1.1.3 Signal input

There are two options for signal input into the recorder; microphone input and analogue line input. The aim of this study was to determine which of these two inputs would be the optimum to use for the 24 hour subject recordings.

3.2.1.1.3.1 Line-input

The SNR of a line-in recording was calculated from a direct comparison with the microphone input to be 58.03 dB from an average signal of 0.25 and an average noise level of $3.09E^{-4}$(Au). The SNR of cough study C5 was calculated to be 74.62 dB from an average signal of 0.50 (Au) and average noise level of $9.29E^{-5}$(Au).

A major problem with the line-input is the automatic introduction of track marks for every period of no-sound encountered, where a period of no-sound describes a situation when the input falls to 4.8 mV[118]. This can result in hundreds of tracks per hour which leads to a lengthy combination step and media-to-PC transfer time. This was resolved by introduction of a continuous tone into the recorder such that it never encounters a period of true silence. As long as the tone is of a small, limited frequency range that is outside the frequency region under study, it should contribute no interference to the recording. This was proved to successfully reduce the line-in recording to a single track of data; however, the constant tone generator appeared to consume a high level of power and subsequently reduced the available recording time

from 24 hours down to just a few. This was a problem for the hardware that had to be addressed, but for the remainder of the work, the microphone input was used.

3.2.1.1.3.2 Microphone-input

The SNR of a recording made through the microphone input for comparison with the line-input was calculated to be 55.92 dB from an average signal of 0.96 and average noise level of $1.53E^{-3}$(Au). These values are clearly significantly higher than for the line-in recording, showing that although the SNR is very similar, there is actually a much greater degree of noise. The SNR of cough study C10 was calculated to be 73.07 dB from an average signal of 0.45(Au) and an average noise level of $1.01E^{-4}$(Au). The line-input clearly possesses a higher SNR than the microphone input and also has a baseline during quiet periods that is five times smaller. However, it carries with it the disadvantage of frequent track splitting. An advantage to both of these recording inputs is that they both possess greater SNRs than the old hardware which was calculated at 57.12 dB for cough study A1 from an average signal of 0.24 and an average noise level of $3.37E^{-4}$(Au).

The use of attenuators was experimented with to reduce the amount of noise picked up on the microphone input. However, the use of attenuators does not improve the SNR, it simply reduces the intensity of all sounds; thus noise and interfering sounds such as speech may be reduced, but they are done so to the detriment of the cough events. It was decided that changing the threshold values of detection, as described in Section 1.6.2.1.1, is a simpler and less permanent step to achieve the same result.

3.2.1.2 Microphone

The chosen microphone was a dual channel, cardioid, capacitor microphone. The dual channel aspect allowed the use of the first channel for the recording of coughs from the subject while the second channel could be used for the event information. A further application for the second channel was that of background noise subtraction in case of extreme interference on the recording. However, it was decided that this was an unnecessary addition for two reasons: firstly, it would be a lengthy procedure to perform a background subtraction on the entire audio recording; secondly, it would be

unlikely that a patient would be in such a noisy environment for the entirety of their study, and in which case it would be unfeasible to have to listen to the recording in order to identify when background subtraction would be necessary, as this element of manual processing is what the system design is aiming to avoid.

The capacitor microphone has a cardioid pick-up pattern, which is omnidirectional therefore does not need exact positioning to avoid the exclusion of sounds from certain directions. An omnidirectional microphone is a necessity with this application as once the patient has left the clinic, their activities over the 24 hour study period will almost certainly mean that the position of the microphone is moved to a certain extent. However, the fact that the microphone does not pick up sound from behind, means that a reasonable amount of unnecessary sound will be excluded from the recording.

Capacitor microphones are highly sensitive; however their sensitivity can be attenuated by applying a greater charge over the capacitor. This ability to restrict the level of sound that the microphone detects is not achievable with some other microphones such as the dynamic or electret types as their sensitivity is fixed. The capacitor also possesses a fast signal response and the capability of a high sampling rate such that a wide range of frequencies can be studied.

The frequency response of up to 20,000 Hz is adequate for capturing maximum information regarding the frequency content of the cough; and due to the Nyquist frequency and the sampling rate of 44,100 Hz, it would only be possible to study frequencies of up to 22,050 Hz. In comparison, contact microphones, or more specifically throat microphones, are used in areas with a high level of ambient noise and therefore offer the capability of detecting a limited number of sounds; however, their frequency range is typically 300 to 3400 Hz which cuts out a lot of the higher frequency information useful to spectral coefficients analysis.

3.2.1.3 Power supply

The Sony MD-RH10 can record for up to 8.5 hours using its fully charged nickel metal hydride (NH-14WM) 1.2 V battery. It is supplied with an extra attachable case for one dry AA cell which provides enough power for up to three extra hours of

recording. Thus possessing the potential for 12.5 hours of recording, the system required modifications in order to be used for 24 hour recordings.

As the internal battery provided so much power, it was decided to consolidate this by using an extra battery pack fed into one of the other two inputs; the mains charging input or the input for the additional battery. A feature of the MD which must be considered at this point is the need for "system file writing" at the end of each recording; responsible for ensuring that the recorded data is fully saved and readable in the future. This step requires an additional amount of power which, if interrupted can result in the loss of the entire recording. The recorder safeguards against such a problem by possessing a function which constantly measures the voltage of its power supply. If the voltage drops below a set threshold, the recorder assumes the battery power is failing and stops the current recording in order to have enough power left to save data. However, when powering the device through the mains input, the device assumes that the power will be constant and of unlimited supply so this feature is not active. Supplying the additional portable power through the mains input would therefore pose more of a risk of data loss in case of power interruption. This risk could be reduced by ensuring that the extra battery has far more power than is required, such that the disk would reach full capacity and induce the data-saving step well before the power diminishes. However, it was discovered that if the recorder received any power through the mains input, it bypassed the use of the internal battery in favour of this supply. This would then take the requirements of the extra battery up from 11.5 to in excess of 24 hours, and would also need to supply the high voltage of 3 V expected by the device.

The next option was to consolidate the internal battery by supplying power through the additional power input, as the system was designed for, but replacing the AA cell with one capable of 11.5 hours of recording. Due to the anticipated high use of the device, it was considered much more cost-effective and environmentally friendly to utilise rechargeable batteries; a nickel metal hydride 1.2 V D cell possessed adequate power for this application. However, the expected voltage of the additional battery input was 1.5 V compared to the 1.2 V of the internal rechargeable battery. Thus, the recorder required the additional battery to supply a higher constant voltage to

continuously operate without provoking the data-saving step. Since the selected battery possessed more than enough power for a 24 hour recording, the battery was connected via the internal battery contacts thus replacing the existing power supply completely with a much larger battery [See Figure 3.2].

The resulting device was proven to successfully make 24 hour recordings.

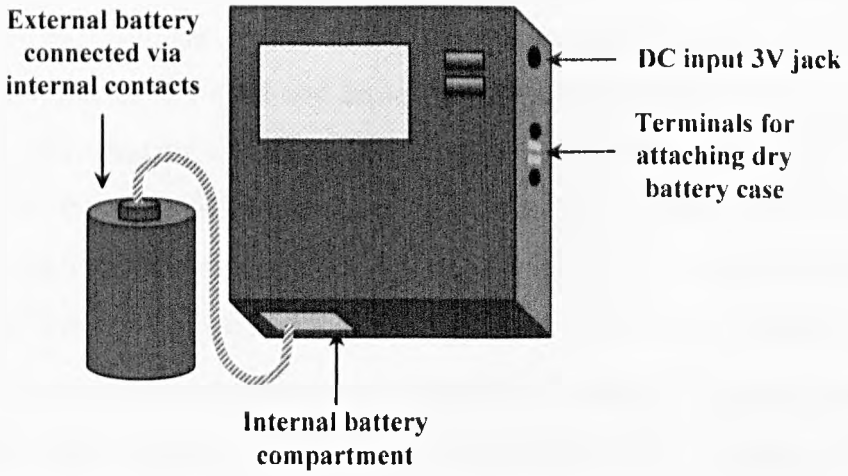


Figure 3.2 Diagram of the modified power supply for the device in order to achieve 24 hour recording capabilities

3.2.1.4 Portable case

In order to make the device suitable for patient use, a portable, convenient and secure case was required to contain all the components of the cough counter including the recorder, the microphone and the extra power supply.

While the case needs to be easily accessed and maintained by the clinic staff, it must remain secure and inaccessible to the patient. This was achieved by complete enclosure of the system into a compact case, lockable by four tamper-proof screws and two security locks [See Figures 3.3 and 3.4]. A small grill was included in case the second channel was to be used for background recording for subsequent noise cancellation. A relatively large button, for ease of use, was fitted to the top of the case to serve as the patient activity marker. An adjustable strap is attached to either side of the case and is designed to fit across the body of the patient such that the mid-point of the strap sits on the shoulder and the device sits on the opposite hip. The set up for audio data collection was optimised to determine the best microphone position for

recording coughs. The microphone is attached to the device's strap such that it is positioned on the centre of the patient's chest pointing upwards toward the mouth. Attaching the microphone to the strap means that its placement is consistent both during the 24 hours and also between patients; thus, the microphone's position is not dependent upon the patient's choice of clothing, for example when using lapels or collars for attachment. The fitting of the strap removes any need for the patient to hold the device, limits the chance of the microphone moving and also means that the entire device can be removed and replaced easily by the patient for example when showering, changing or sleeping. The strap is adjustable i.e. can be extended or shortened for the patients comfort. During the night, the strap can be placed over a headboard such that the microphone is closest to the patients head, or can simply be placed on a bed-side table, again with the microphone close to the patient.

The resulting set up was successful in recording the subject's upper airway sounds, easily transferable between users and was widely accepted by patients for 24 hour use.

3.2.1.5 Event marker

The event marker consists of a button on the external surface of the case which records a characteristic marker sound on the second channel of the microphone. Patients are instructed to register more severe cough episodes by pressing the event marker button and audibly describing the activity they were carrying out prior to the cough. Along with the objective counting of cough, it was considered advantageous to have an additional monitoring feature which enabled the correlation of patient activities with coughing episodes.

As cough is a symptom of the underlying cause, studying the relationship of cough to certain patient activities can contribute important information to a diagnosis. As discussed in Section 1.2.2, a patients subjective scoring of cough can be unreliable and often no patterns to coughing episodes are recognised. A more objective system is therefore required to study such correlations. One cause of cough which can be identified in this way is GORD. In up to 75% of patients with GORD, cough is the sole presenting symptom; therefore the absence of heartburn and other associated

symptoms is not necessarily enough to refute a diagnosis. In GORD sufferers, cough reflexes are stimulated by the occurrence of reflux into the oesophagus and typically occur when the oesophageal sphincter (OS) is opened. Activities responsible for the opening of the OS include eating and the subsequent release of gas from the stomach, and any movement which uses the diaphragm, as this puts direct pressure on the OS and causes it to open. Such movements include talking, due to excessive use of the diaphragm for phonation, driving, bending and getting out of bed in the morning. Other triggers of coughing episodes could include the patients contact with certain allergens. Thus, by recording certain activities of the patient and correlating them to the temporal cough counts, cough patients can be studied and potentially diagnosed. Carrying out this study of cough and its relationship to trigger factors alongside the objective 24 hour cough counting clearly requires subject input. In order to make the system as automatic as possible, it was considered ideal to use the recording device to record the necessary activity information. It would then be possible to compile all the information at once. One option for achieving this objective would be to have several buttons, each with a characteristic marker, which the patient presses to indicate their current activity. However, the maximum number of activities which could be studied would be set at the time the hardware was developed, with no room to expand or change as studies showed new information. In addition to this, the studying of a sufficient number of activities would create a complicated system for the patient to use. It was therefore decided to utilise the fact that the upper airway sounds of the patient were already being recorded in the cough study and instruct the patients to audibly describe the activity they had been carrying out prior to the coughing episode. In this way, a single event marker button could be used to highlight the activity whilst the software could be developed to locate the markers and the associated descriptions. Furthermore, used in this way, the event marker can then be applied to many additional situations; for example in drug efficacy studies, the event marker can indicate the administering of medication.

The resulting device was a working marker which was easily used by patients and successfully inserted a characteristic marker onto the second channel of the recording.

3.2.2 Hull Automatic Cough Counter (HACC)

The final device, called the Hull Automatic Cough Counter (HACC) is shown in Figures 3.3 and 3.4. All the desired features for the device including adequate data storage, power supply and the patient activity marker have been included. All the components of the device are enclosed in a secure case with a shoulder strap for the patient's comfort and convenience and optimal positioning of the microphone. The device is fully operable by patients and has been demonstrated to work as intended.

Figure 3.3 The exterior of the device

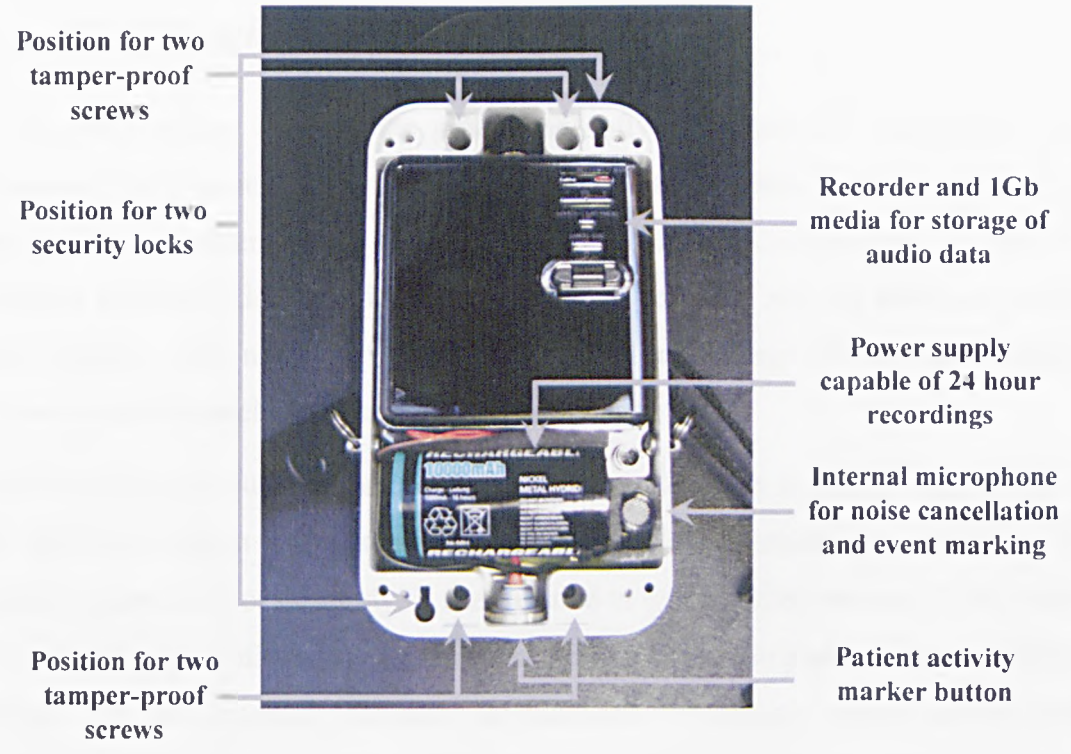


Figure 3.4 The interior of the recording device

3.3 Development of Software

The aim of this section was to develop the software for the automatic counting of cough in audio recordings. As a method for computerised processing of audio recordings already existed, it was initially decided to attempt to further develop and modify this software for application to the new, extended recordings. In order to develop the software, a dataset of cough studies collected with the new hardware was required.

3.3.1 Data collection

Data was collected using the recording device as described in Section 3.2. The resultant data from the 24 hour cough studies are very different to those used to develop the original method as they contain a much lower concentration of coughs and a much larger amount of extraneous background sounds.

3.3.2 Cough studies

In order to develop and test the processing software, a series of recordings were collected. The first stage was the collection of short, ambulatory audio recordings of chronic cough patients. The purpose of these initial recordings was to test the software following the modifications to both the software and the hardware, whilst still working with manageable file sizes. Eight recordings of varying durations between approximately 30 minutes and 3 hours were made.

Following this, the study duration was increased to 24 hours to test the capabilities of the hardware and to fully assess the modified software under test conditions. To maximise the utility of the study, it was decided to test the effectiveness of the system for multiple uses; both for its use in GORD diagnosis and in pharmaceutical efficacy studies. For this purpose, the study set included two chronic cough patients with GORD, one cystic fibrosis patient both on and off cough medication and a further six cystic fibrosis patients.

The result of the experiment was a successful collection of audio recordings. The developed hardware achieved a 100% success rate of recording for 24 hours. Patient feedback regarding the comfort and convenience of the device was positive.

3.3.2.1 Data processing

The audio recordings were then transferred to the PC and prepared for further processing. The files existed on the MD media as two compressed ATRAC3plus files; one of length 999:59 minutes, as this is the maximum value the counter can reach, and the other of approximately 440:00 minutes, which in total made up the 24 hours of data. The data was transferred via USB to the PC using the Sonicstage software. The data was required to be in WAV format in order to be compatible with Matlab; however, this expands a 24 hour stereo audio file from 0.65 Gb to 14.2 Gb. To avoid memory constraints associated with processing such large files, the ATRAC files are initially divided into three-hour portions, which when decompressed will approximate to 1.8 Gb. The files are all named according to the date of the original

file and are suffixed to show to which three hour portion of the eight created they belong.

Matlab reads only the first channel of a stereo audio file; the data can therefore remain in stereo format for the cough recognition work without any interference from the second channel. However, in order to carry out the event marker processing on the second channel, the first channel needs to be removed. This was achieved by creating a mono file based on the second channel alone for each three-hour portion of WAV data.

The result was 16 WAV files for each 24 hour recording; eight stereo files for cough recognition processing and eight mono files for the event marker processing. At this stage, all files were in the appropriate format to be read by Matlab.

3.3.2.2 Test dataset

In order to test the system on a variety of coughs sounds a test dataset was created, consisting of a short audio file containing a variety of sound events and coughs from different subjects. In this way, the analysis methods could be tested on a variety of coughs and other sounds rather than restricting each analysis to a single subject. This was considered necessary to avoid limiting each analysis to one subject and tailoring any modifications to the one set of results only to then have to address a different set of problems and modifications for the next subject. The resulting file was of a manageable size for repeated analysis and allows a wide range of cough types to be studied at once.

3.3.2.3 Test sounds

In order for individual sounds and coughs to be studied, a variety of cough and non-cough sounds were isolated as separate events for both spectral analysis and for use in assessing analysis methods. For the purpose of this work, a cough will be defined as an explosive sound separated by a fall of sound level to below threshold, regardless of the presence or absence of a preceding inspiratory effort. The result was a set of 58 coughs and 33 other sounds including speech, mechanical sounds, breathing and laughter.

3.3.3 Sound identification

3.3.3.1 Cough sound acoustics

The first step towards finding a method for the automatic identification of cough was to visually inspect the features of various graphically represented sounds in order to investigate whether or not there are any clear methods for distinguishing cough from all other sounds. While it is obviously necessary to find statistically relevant differences between coughs and other sounds, it is perhaps more important to find statistical similarities between coughs. This is because it would be impossible to create one group to fit the vast amount of non-cough sounds which would potentially be encountered during a 24 hour ambulatory recording. Thus, it is not going to be feasible to simply perform a statistical measure, such as cluster analysis, to determine which group a sound event is most likely to belong. A better approach to the problem would be to find some characteristic possessed only by coughs which could then be checked for in order to classify the sound.

Two possibilities exist for cough sound analysis: one is the registration of acoustic pressure accompanying cough in the form of a time-amplitude wave, also known as a tussiphonogram; the second is the analysis of the frequency content of the cough sound.

Several methods of analysis have been studied by other groups including the use of spectrograms, pressure plots, cepstral coefficients, zero-crossing rates and neural networks [See Section 1.3.1]. For this study, analysis was carried out in both the time and frequency domains.

3.3.3.1.1 Time-domain analysis

Observations were initially made on coughs alone to attempt to find characteristics that could then determined as absent from other non-cough sounds.

As discussed in Section 1.3.1, literature has reported that cough can be characterised as a composite of two or three phases: an expulsive phase occurring at the moment of glottal opening, responsible for the high intensity sound; an intermediate or steady state phase which is lower in amplitude and associated with a steady flow of air

through the open glottis; and occasionally a third voiced phase which is caused by the partial closing of the vocal chords vibrating toward the end of the cough. While this qualitatively describes an almost uniform pattern to the cough sound, quantitative uniformity is practically impossible. In addition to this problem, literature also reports that cough changes its sound in disease[49], thus making the potential variation between cough sounds even greater. An example of this is shown in Figure 1.2, which is a summary of the variety in cough sound patterns, intensities and durations for different diseases.

Observation of the coughs in the time-domain immediately shows that there is no uniformity to either the overall duration of coughs or the duration of each cough phase; neither within or between patients. Even the presence of a third phase is a variable occurrence within an individual, as shown in Figures 3.5 and 3.6, the former is a two phase cough while the latter has three phases; both are coughs from the same patient.

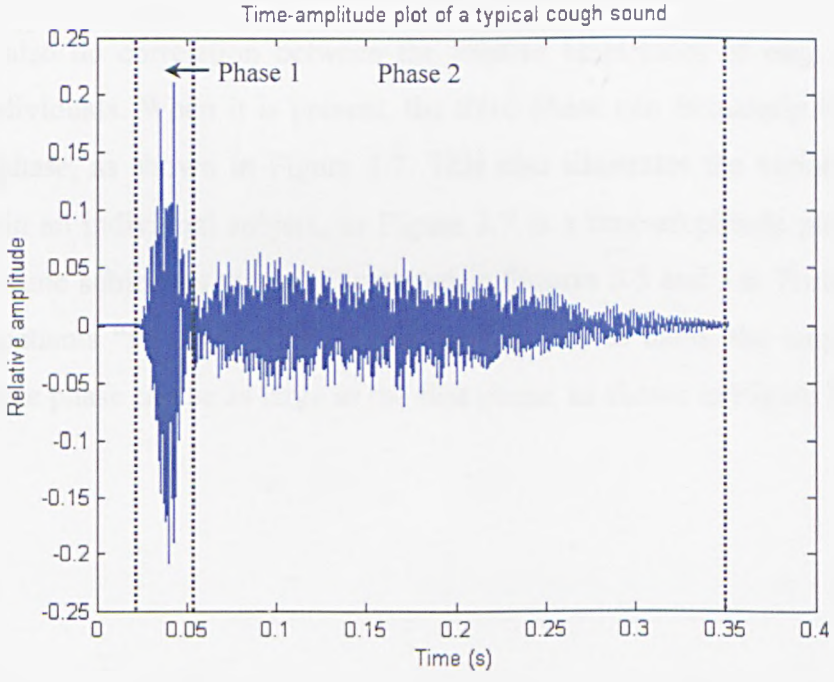


Figure 3.5 Time-amplitude plot of a cough sound illustrating the absence of the third cough phase

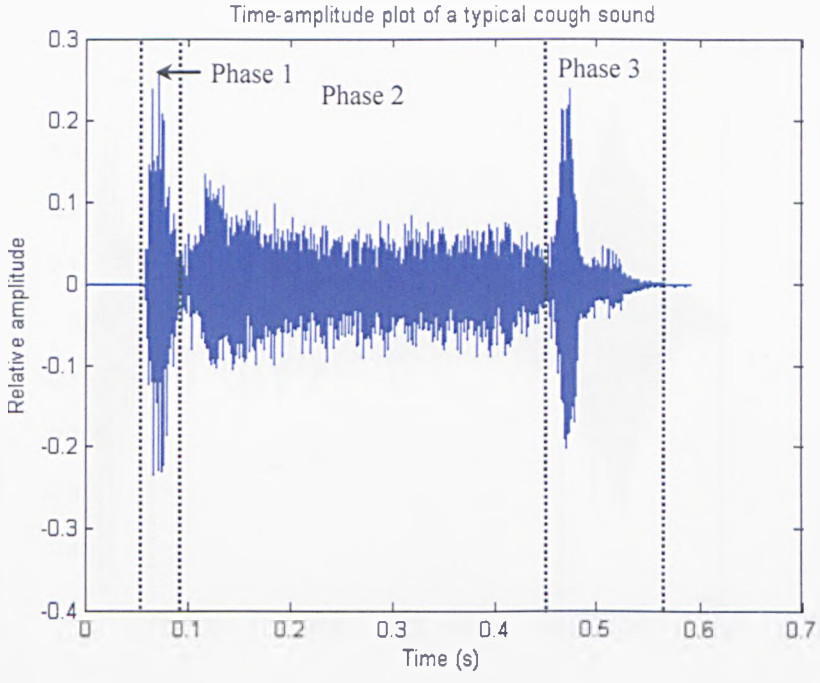


Figure 3.6 Time-amplitude plot of a cough sound illustrating the presence of a third cough phase

There is also no correlation between the relative amplitudes of each cough phase within individuals. When it is present, the third phase can frequently be larger than the first phase, as shown in Figure 3.7. This also illustrates the variation of cough type within an individual subject, as Figure 3.7 is a time-amplitude plot of a cough from the same subject as the one illustrated in Figures 3.5 and 3.6. Furthermore, due to some patients "voicing" their coughs, usually due to habit, the amplitude of the intermediate phase can be as large as the first phase, as shown in Figure 3.8.

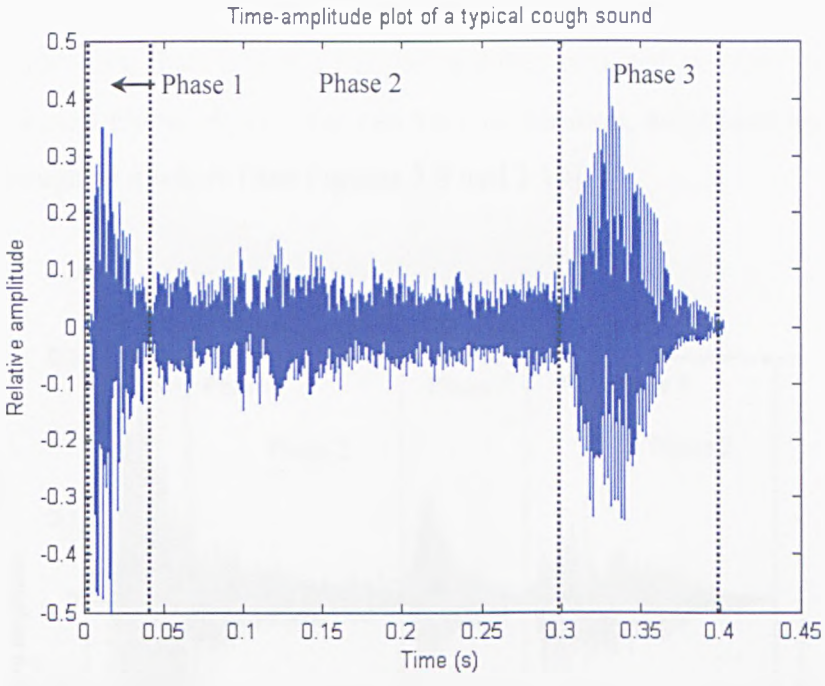


Figure 3.7 Time-amplitude plot of a cough sound, illustrating the larger amplitude of the third phase in relation to the first phase

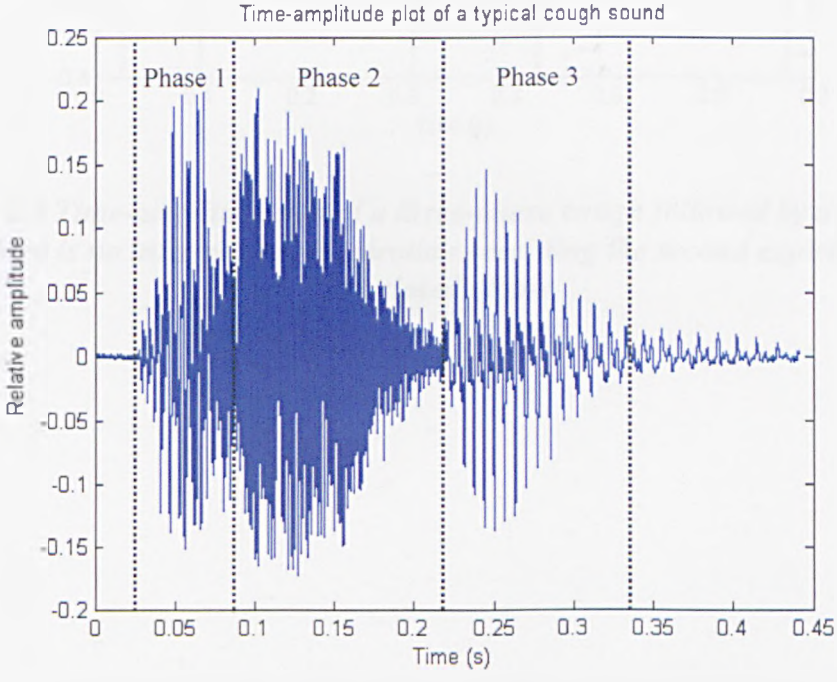


Figure 3.8 Time amplitude plot of a cough sound, illustrating the large amplitude of the second phase in relation to the first and third phases

Some patients have multiple expirations against the closed glottis for only a single inspiration, meaning that their third phase is either replaced or followed by what is essentially, a first phase. Again, this can vary in duration, amplitude and positioning within the cough as a whole [See Figures 3.9 and 3.10].

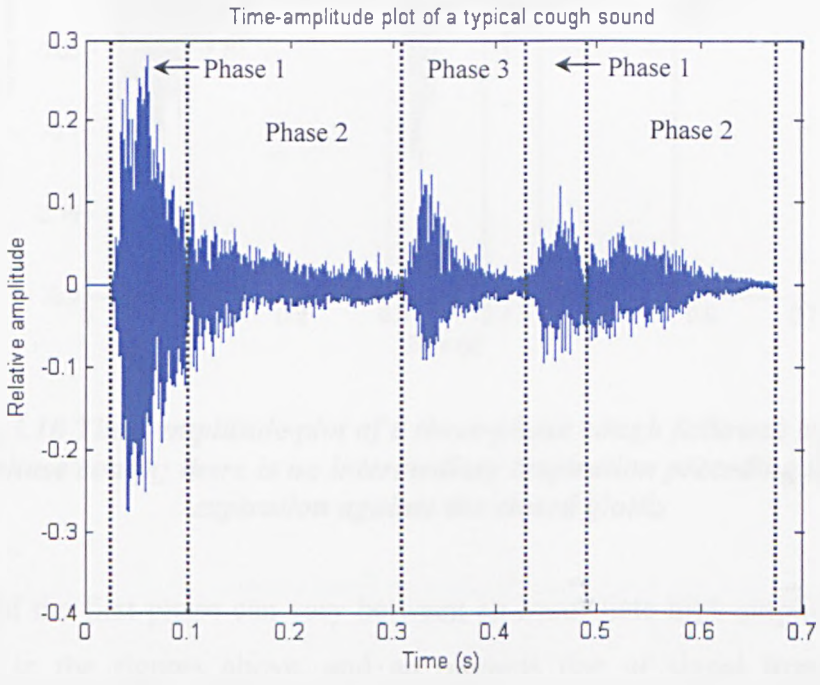


Figure 3.9 Time-amplitude plot of a three-phase cough followed by a two-phase cough; there is no intermediary inspiration preceding the second expiration against the closed glottis

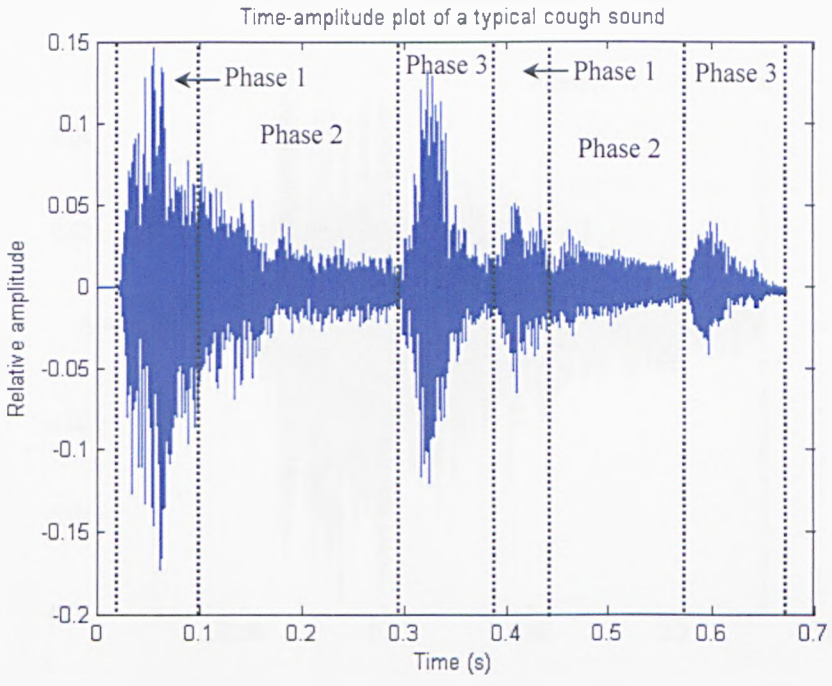


Figure 3.10 Time-amplitude plot of a three-phase cough followed by a second three-phase cough; there is no intermediary inspiration preceding the second expiration against the closed glottis

The start of the first phase can vary between an immediate high amplitude onset, as illustrated in the figures above, and an obvious rise of signal from baseline to relatively low amplitude. This is illustrated in Figures 3.11 and 3.12. This instance usually occurs when the expiration against the closed glottis is less powerful, such as when the subject does not have a large enough, or even any, inspiration prior to the expiration.

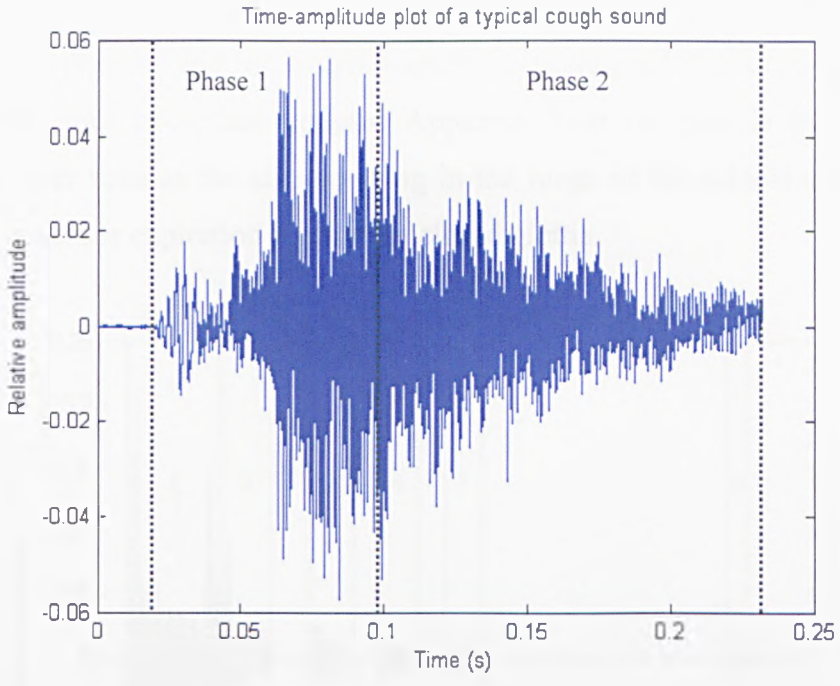


Figure 3.11 Time-amplitude plot of a two-phase cough following a minimal inspiration, the onset of the amplitude is not as rapid as in other coughs and the amplitude does not reach as high a level

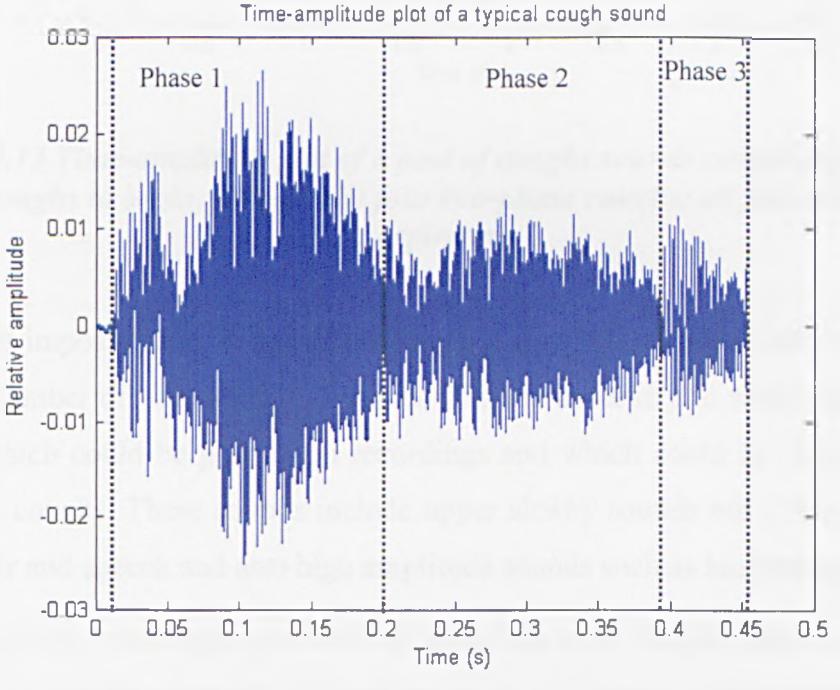


Figure 3.12 Time-amplitude plot of a three-phase cough following a minimal inspiration, the onset of the amplitude is not as rapid as in other coughs and the amplitude does not reach as high a level

Cough events may also occur in peals of cough sounds following a single inspiration. Figure 3.13 shows a peal of cough sounds, consisting of two three-phase coughs followed by four two-phase coughs. Apparent from the plot is the decrease in amplitude over time as the air remaining in the lungs of the subject diminishes and results in a weaker expiration against the closed glottis.

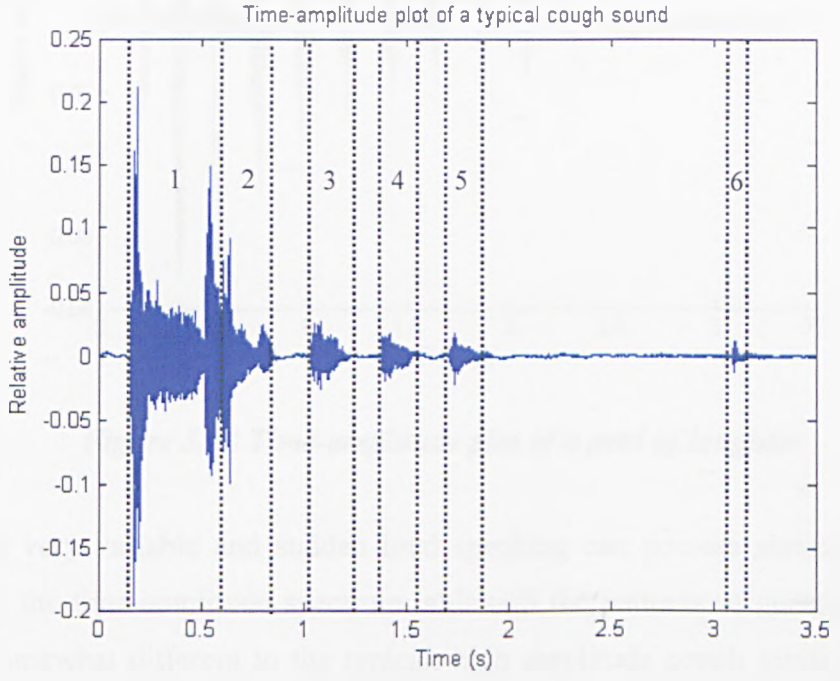


Figure 3.13 Time-amplitude plot of a peal of coughs sounds containing two three-phase coughs to begin, followed by four two-phase coughs; all following a single inspiration

While it is impossible to study all non-cough sounds in the same way, simply due to the vast number of possibilities, it was considered necessary to study some common sounds which could be present on recordings and which could be characteristically similar to coughs. These sounds include upper airway sounds other than cough such as laughter and speech and also high amplitude sounds such as loud bangs.

Analysis in the time-domain showed that one of the most similar sounds to cough was laughter. As it is characteristic for laughter to occur in peals, a three second period of laughing can generate 12 high-amplitude spikes, as shown in Figure 3.14, which could cause a massive overestimation of cough frequency if these were to be mistaken for peals of coughing.

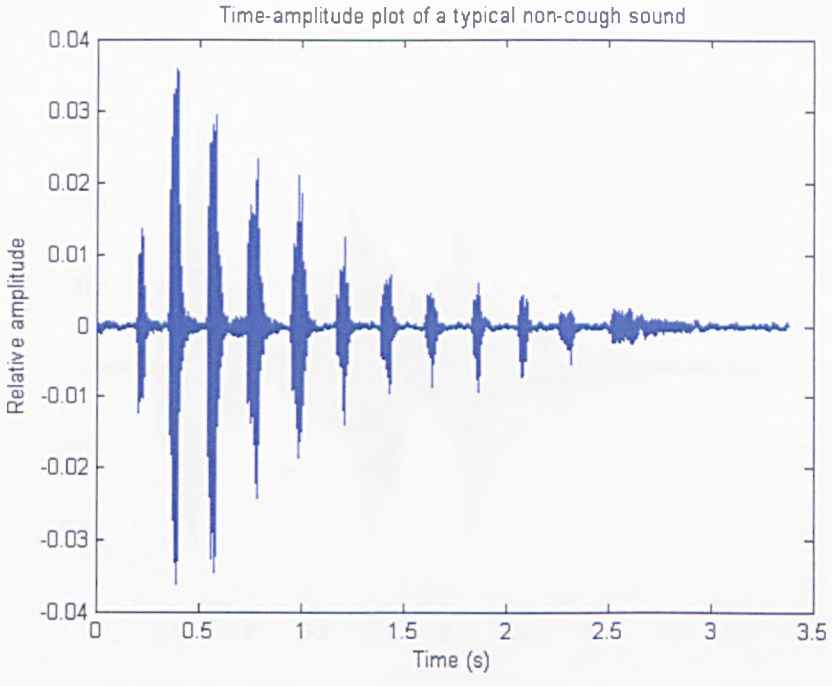


Figure 3.14 Time-amplitude plot of a peal of laughter

Speech is very variable and sudden loud speaking can possess similar patterns as coughs in the time-amplitude spectrum, although the patterns of general speech are usually somewhat different to the typical, high amplitude cough event as discussed previously. However, speech can be very similar to the low amplitude coughs shown in Figures 3.11 and 3.12 especially when single words are spoken. Figure 3.15 shows the time-amplitude plot of a portion of the words "What's your name?" as spoken by a female speaker. The patterns possess a number of similarities to the time-amplitude plots of cough sounds.

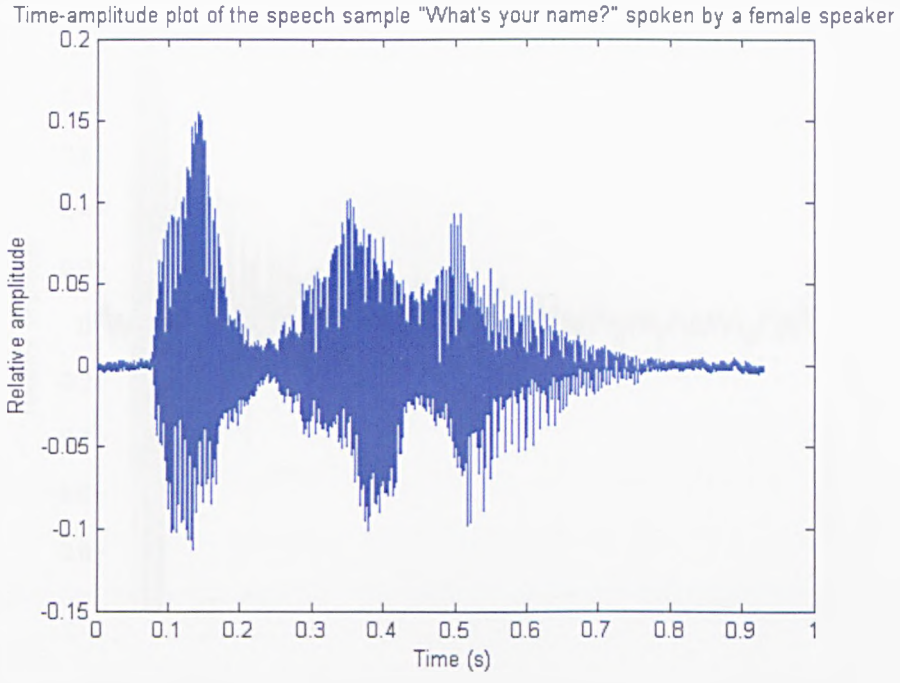


Figure 3.15 Time-amplitude plot of the words "What's your name" spoken by a female speaker; illustrating the similarities of some speech to some of the cough events discussed previously

Loud sounds such as a door banging shut typically contain the rapid onset of a high amplitude signal that is also associated with a cough event. However, the gradual decrease in amplitude is not overly similar to that in a cough sound, the RMS of the signal for the bang would appear almost exponential in shape over the course of the decay. Due to the introduction of voice into cough sounds, they tend not to be quite so uniform. A time-amplitude plot of such a sound is illustrated in Figure 3.16.

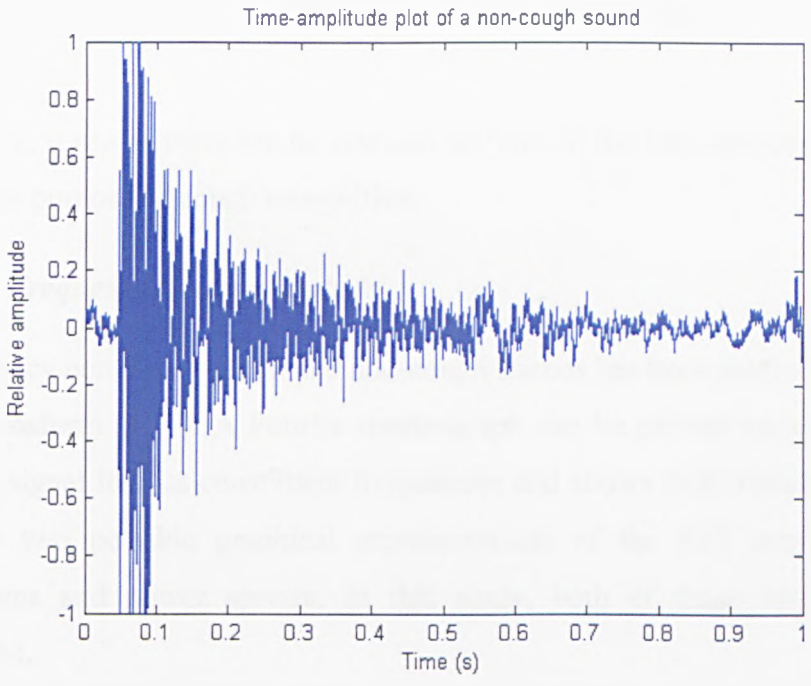


Figure 3.16 Time-amplitude plot of the sound of a door banging; illustrating the rapid onset of high amplitude signal that is also associated with cough sounds

The results of this analysis show that features present in the time-domain of cough sounds are not adequate to characterise cough sounds as unique with respect to all other sound events. Coughs contain a large amount of variation in several aspects such as duration, amplitude and general patterns; so much so it would be difficult to group so many dissimilar sounds together. In addition to this, the coughs are not sufficiently dissimilar to other sounds that have potential to be present in the audio recordings. Other groups have used the time-domain in the study of cough sounds. In 2006, Murata *et al.* visually inspected the time and spectrographic features between voluntary productive and non-productive coughs with a degree of success[60]. However, this is a very different application to the one faced in objective cough counting; the process was not automatic and due to a lack of other sounds in the recording only the two cough types were being studied. In addition, Widdicombe concluded that the tussinophonogram may be of value for studying the mechanisms of airway pathology[119]. Both these studies go to further support the fact that the time amplitude plots of coughs vary sufficiently between coughs of different types. This is

not a desired feature for this application where a characteristic feature needs to be determined.

In summary, it seems there are no relevant features in the time-domain that could be used for the purpose of cough recognition.

3.3.3.1.2 Frequency-domain analysis

The frequency content of cough and non-cough sounds has been studied using the fast Fourier transform (FFT). A Fourier spectrograph can be plotted which deconstructs the cough signal into its constituent frequencies and shows their variation over time. There are two possible graphical representations of the FFT output; frequency spectrograms and power spectra. In this study, both of these forms have been investigated.

3.3.3.1.2.1 Frequency spectrograms

A frequency spectrogram is a three dimensional representation which plots time on the abscissa, frequency on the ordinate and the intensity represented by colour, ranging from red being the highest amplitude to blue being the lowest. Figure 3.17 shows the spectrogram of the two-phase cough represented in Figure 3.5. From the spectrogram, it can be seen that the majority of the high frequencies occur in phase one. This is then followed by a gradual reduction in the intensity of the higher frequencies over phase two. Where a third phase is present within the cough sound, phases one and two take on a similar pattern as before, whilst phase three has an increase in the intensities of the lower frequencies, and a slight increase in the higher frequencies, as shown in Figure 3.18.

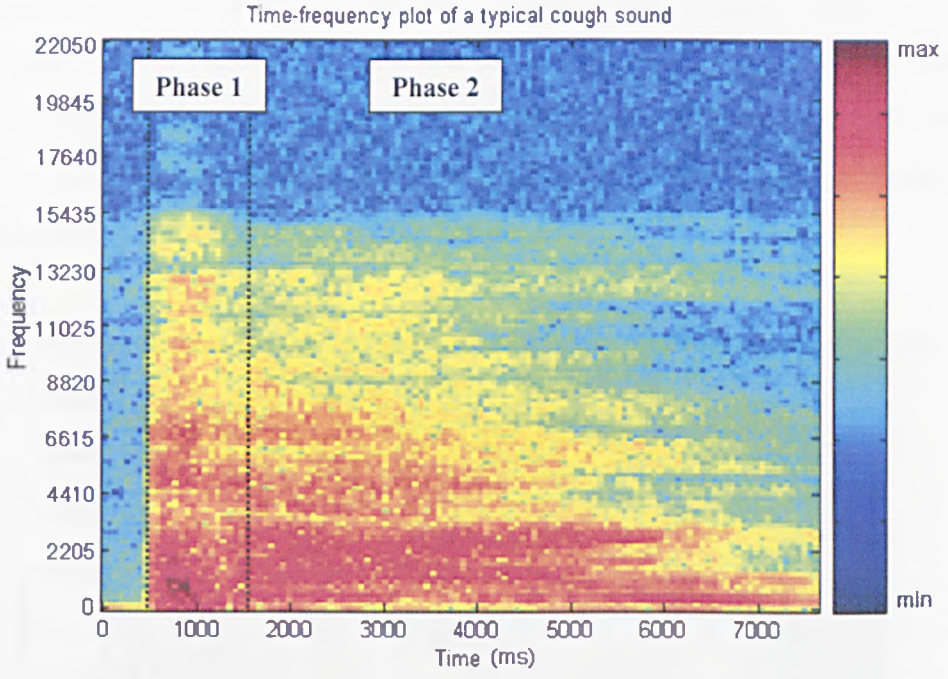


Figure 3.17 Frequency spectrogram of a cough with two phases to the sound

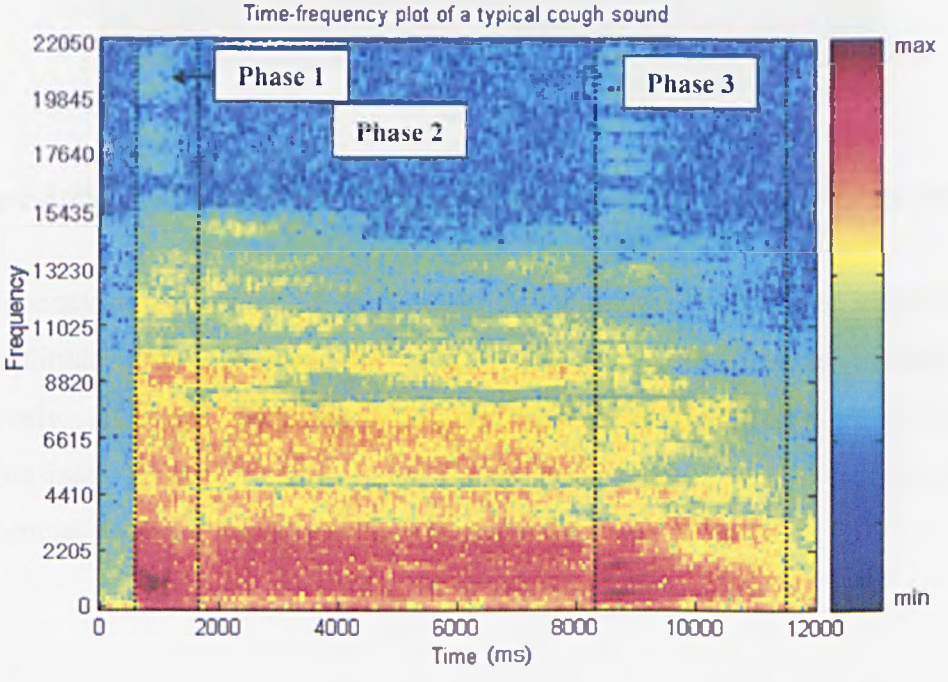


Figure 3.18 Frequency spectrogram of a cough with three phases to the sound

Studying the differences of the frequency spectrograms between patients shows a very similar pattern. Figure 3.19 shows a three phase cough sound from a different patient, which still has the higher intensity high frequencies in the first phase, a decline in high-frequency intensities over the second phase and an increase in low frequency intensity for phase three [See Figure 3.19].

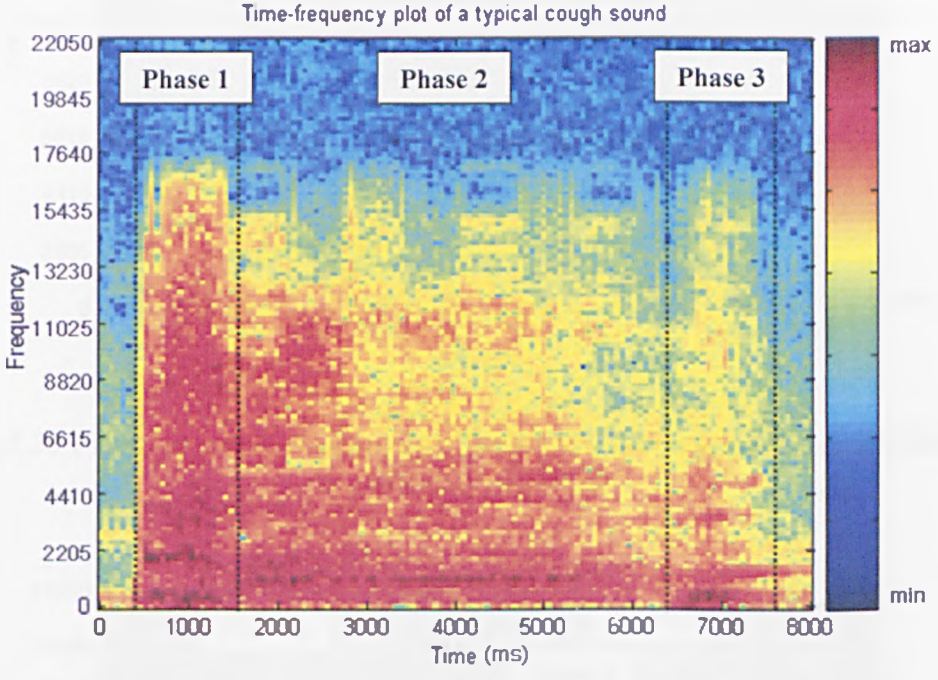


Figure 3.19 Frequency spectrogram of a cough with three phases to the sound

The frequency spectrograms of the low amplitude coughs represented previously as time-amplitude plots in Figures 3.11 and 3.12 are shown in Figures 3.20 and 3.21 respectively. Although with these coughs the higher frequencies don't achieve maximum intensity until later on in the cough event, there is still an obvious change in frequencies over the duration of the cough event.

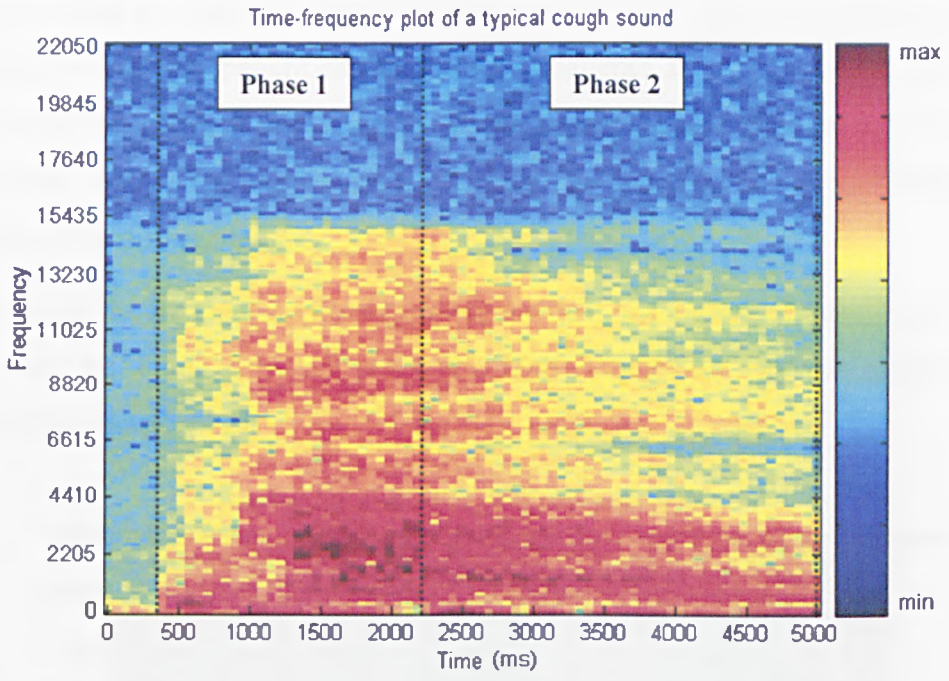


Figure 3.20 Frequency spectrogram of a two-phase, low amplitude cough following a minimal inspiration

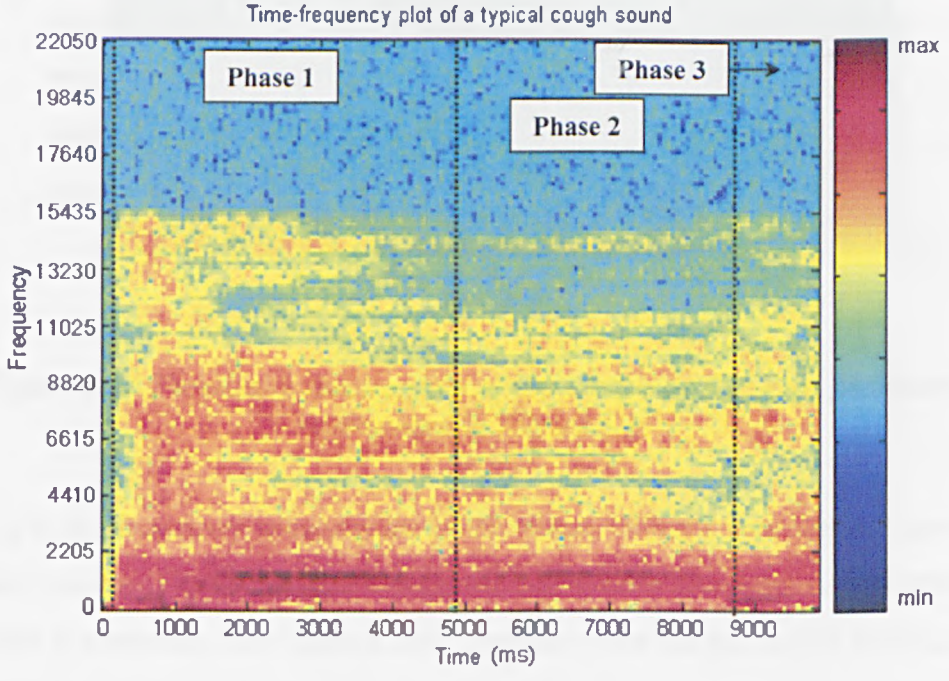


Figure 3.21 Frequency spectrogram of a three-phase, low amplitude cough following a minimal inspiration

Although there are patterns visible in the spectrograms both for different coughs belonging to the same patient and for different patients, there remains no uniformity or quantitative patterns in order to perform statistical comparisons. However, comparison with other sounds such as speech and laughter and loud bangs does highlight some differences.

Inspection of the frequency spectrogram of the peal of laughter discussed previously shows that the sound does not reach the high frequencies that the cough events achieve [See Figure 3.22].

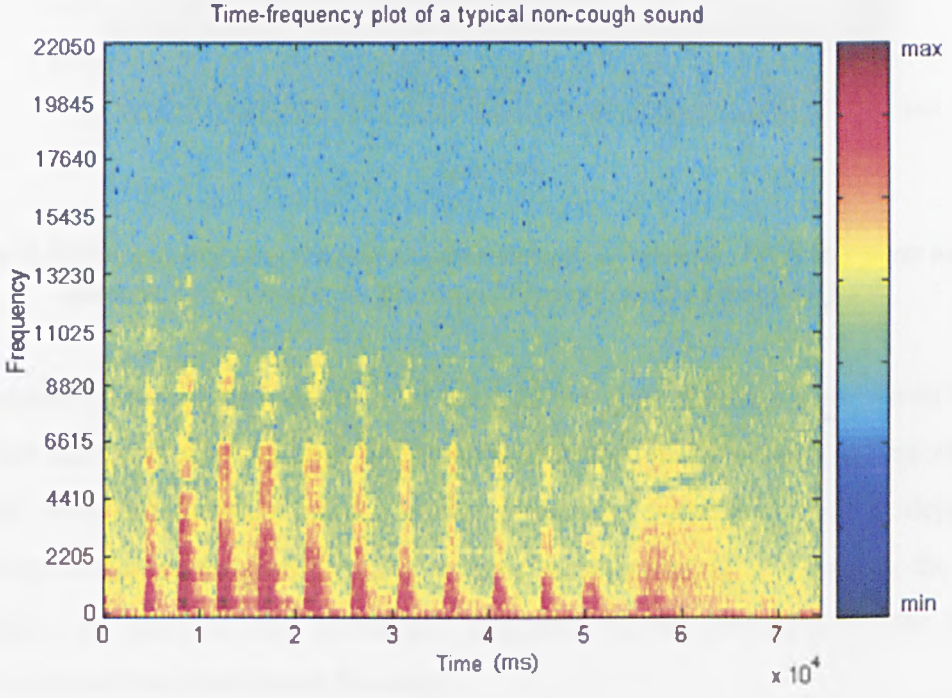


Figure 3.22 Frequency spectrogram of the sample of laughter, previously represented in Figure 3.14

Looking at the frequency spectrogram of the words "What's your name?" spoken by a female speaker [Figure 3.23], although the signal does contain a certain amount of the higher frequencies, they remain fairly constant over the portion of speech, unlike in the cough events.

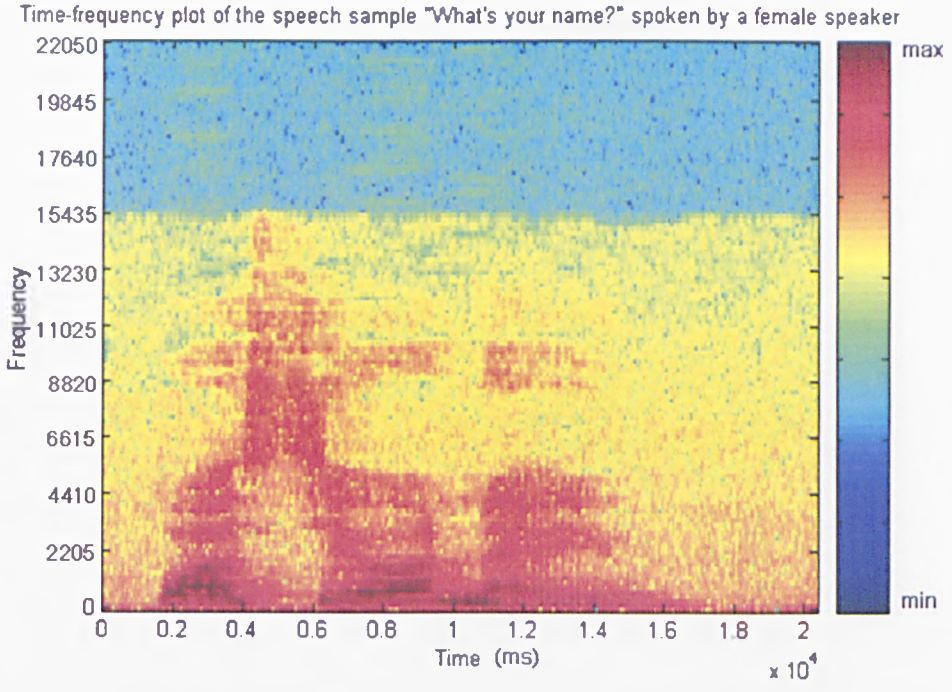


Figure 3.23 Frequency spectrogram of the sample of speech, "What's your name", spoken by a female, as previously represented in Figure 3.15

The frequency spectrogram of the loud bang of the door shutting is shown in Figure 3.24, this both contains the high frequencies that the cough events possess and has variation over the duration of the event. However, there is a certain degree of uniformity to the spectrogram that the cough spectrograms do not possess; the main one being the gradual and consistent decrease in the intensity of the higher frequencies over the duration of the event.

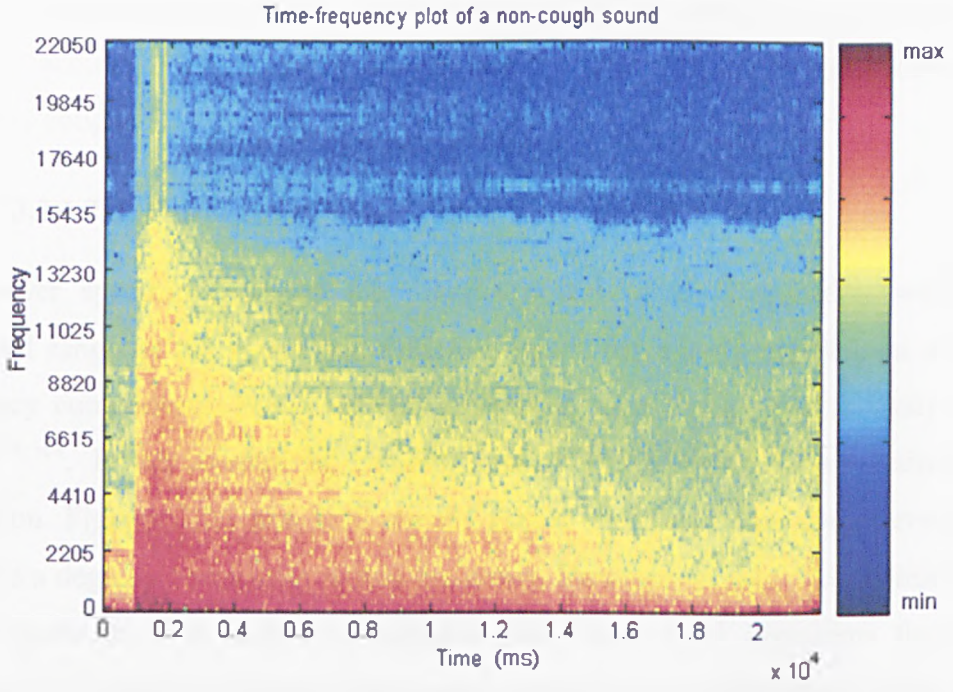


Figure 3.24 Frequency spectrogram of the sample of a door banging, previously represented in Figure 3.16

In summary, it appears from inspection that the frequency spectrograms contain little quantitative information which could be used to recognise coughs and distinguish them from all other non-cough sounds. However, they do contain more qualitative information than the time-amplitude plots studied previously.

Spectral analysis for the study of cough sounds has been the choice for several research groups[53-55, 57, 58, 60, 61] for a variety of applications. However, the majority of these studies applied the FFT to isolated cough sounds and only had a very specific aim such as identifying differences between two sound types. This is very different to the application of automatic cough recognition in recordings.

Since it is apparent that spectral features could qualitatively distinguish between coughs and non-coughs, it seems the calculation of the cepstral coefficients should be continued. In addition, in 1992, Thorpe *et al.* showed cepstral coefficients to be one of the most effective methods for distinguishing between two types of cough[56]. Due to the lack of quantitative characteristics between the groups, a more general method of pattern recognition will be necessary for the classification stage. It would therefore seem that neural networks are the most appropriate method of pattern recognition to

apply in this case. A LVQ neural network was applied successfully by Van Hirtum in 2002 to achieve 96% accuracy in the discrimination between spontaneous and voluntary coughs[59].

3.3.3.1.2.2 Power spectra

The power spectrum presents the relative power of all frequencies within the measured range of the signal and therefore yields the energy distribution amongst frequency components. Power spectra have been used in the past to study cough sounds[44, 59]. Power spectra of coughs, speech and laughter were produced for inspection. Figures 3.25 and 3.26 are the power spectra of two cough events and illustrate a degree of similarity within the group. However, Figure 3.27 represents the power spectrum of a sample of laughter and Figure 3.28 represents the power spectrum of a portion of speech; these power spectra are not dissimilar to those of the cough events.

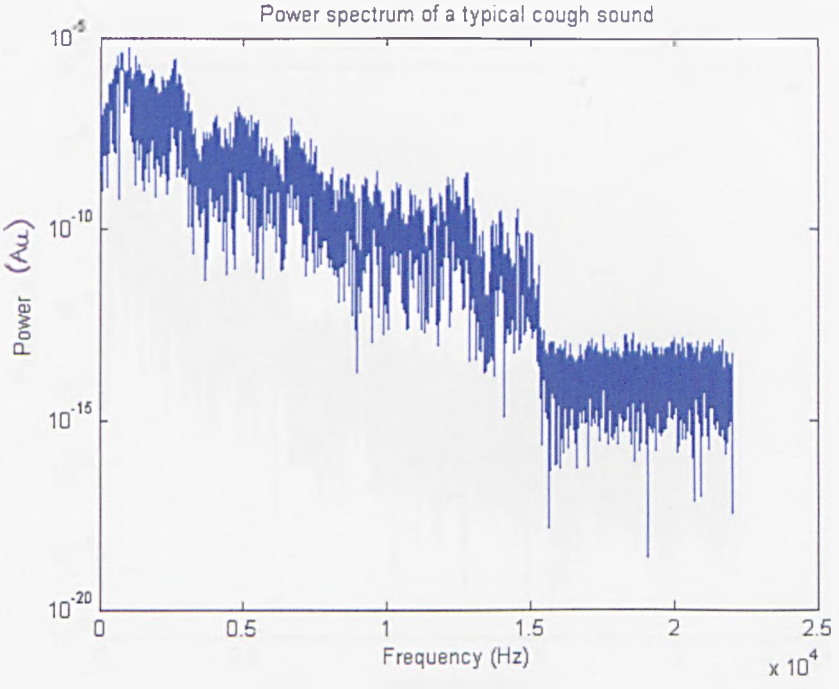


Figure 3.25 Power spectrum of a typical cough sound

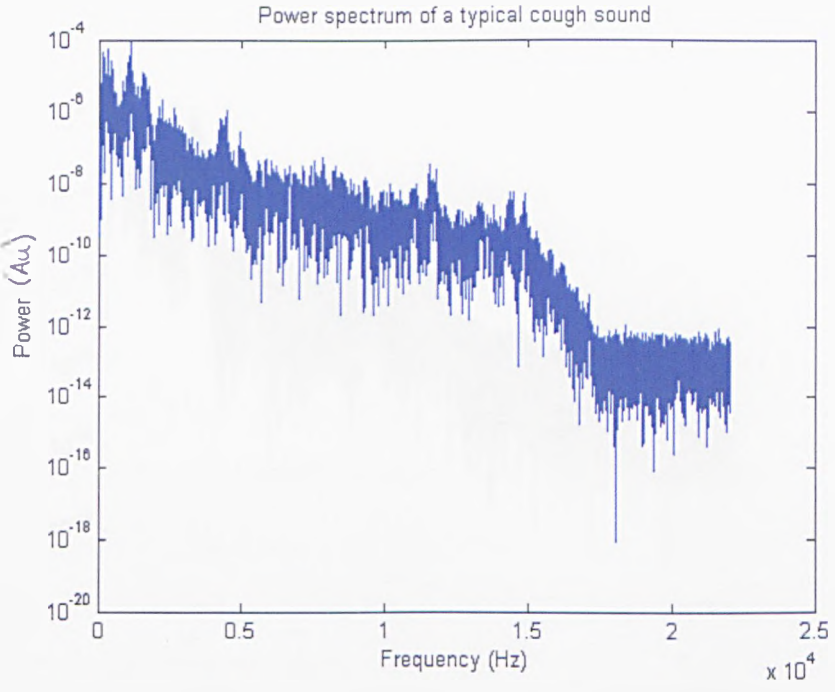


Figure 3.26 Power spectrum of a typical cough sound

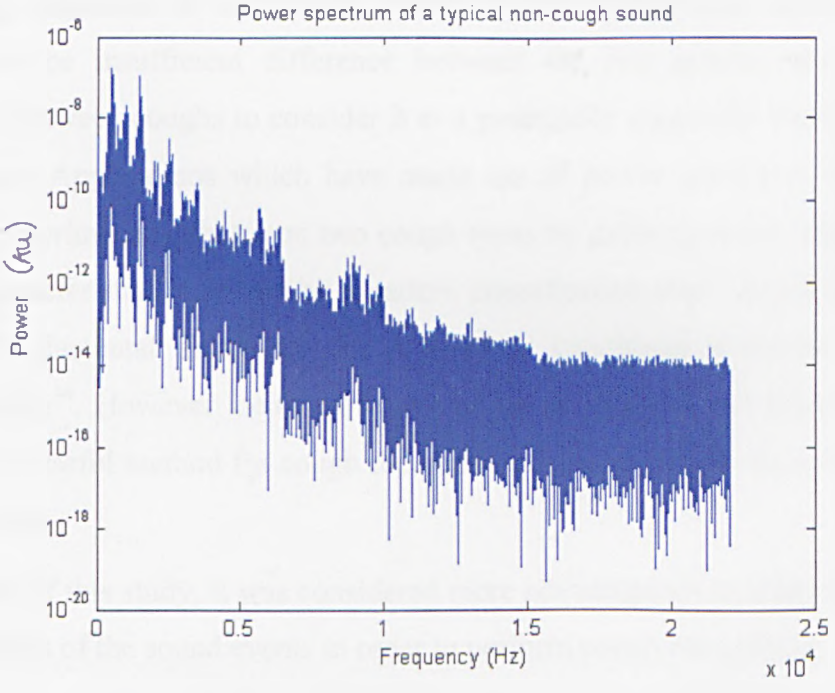


Figure 3.27 Power spectrum of a sample of laughter

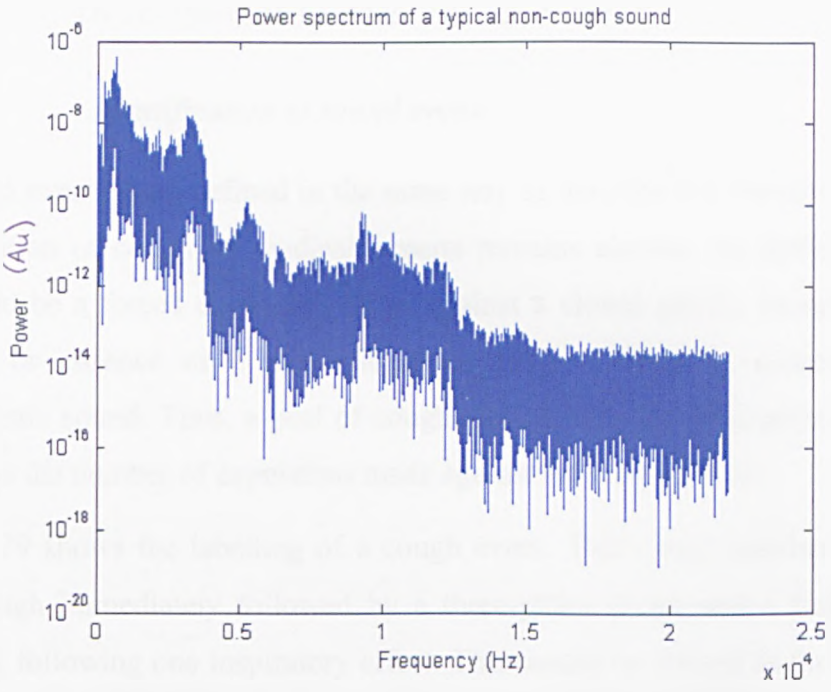


Figure 3.28 Power spectrum of a typical speech sound

Following inspection of the power spectra of both coughs and non-coughs, there appears to be insufficient difference between the two groups, and insufficient similarity between coughs to consider it as a potentially successful method for cough recognition. Applications which have made use of power spectra in the past have included discrimination between two cough types by defining power spectral density (PSD) characteristics followed by a pattern classification step[59] and also calculation of the fundamental frequency as part of a conditions check-list for cough identification[44]. However, neither of these methods indicates that the power spectra would be a useful method for cough recognition amongst an infinite amount of other sound events.

As a result of this study, it was considered more advantageous to study the frequency spectrograms of the sound events in order to perform cough recognition.

3.3.3.2 Development of data pre-treatment

3.3.3.2.1 Identification of sound events

The sound events were defined in the same way as described in Section 1.6.2.1.1. As the definition of cough by medical experts remains elusive, the definition for this work is to be a forced expiratory effort against a closed glottis, irrespective of the presence or absence of a prior inspiratory effort, which is associated with a characteristic sound. Thus, a peal of coughs following a single inspiration would be counted as the number of expulsions made against the closed glottis.

Figure 3.29 shows the labelling of a cough event. This cough consists of one two-phase cough immediately followed by a three-phase cough and a final two phase cough, all following one inspiratory effort. This would be classed as three coughs, as illustrated by the labels. The decrease in relative amplitude following the second phase of the first cough identifies the end of this cough, while a subsequent rise indicates the beginning of the second.

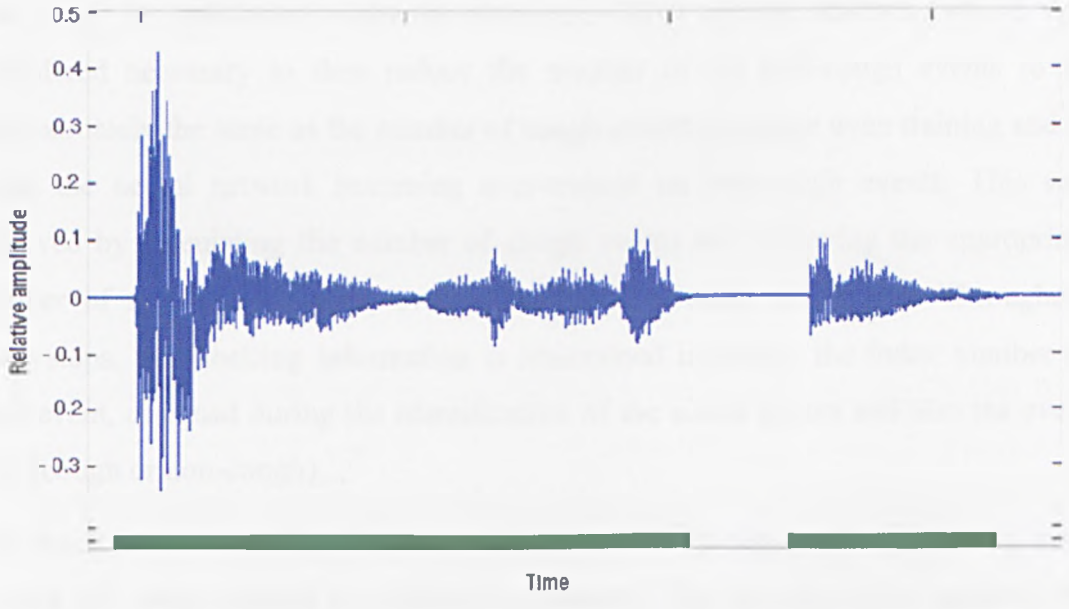


Figure 3.29 Illustration of a cough event showing the labelled region

Since the audio recordings made by the new hardware have a relatively low baseline in comparison with the original recordings, it was considered necessary to re-optimise

the threshold levels. Due to the variety of recordings made which possess a range of SNRs, it was not considered appropriate to fix the threshold levels at set values. Instead, these were altered as the different recordings were processed and will be detailed where necessary.

3.3.3.2.2 Training and validation

The aim of this experiment was to create a function which automatically divides the labelled data into training and validation sets for use by the neural network. Data is labelled manually using the GUI as described in Section 1.6.2.1.3. As non-cough sounds make up the majority of the sound events, it was first decided to separate the sounds into coughs and non-coughs in order to be able to generate an equal distribution of the event types. Separation was achieved by use of the manually assigned labels. Following this, the training and validation groups were established. It is important to have a separate dataset for validation in order for the neural network to be properly tested on data not previously presented to the network. It was considered appropriate to have two thirds of the data for training and the remaining third for validation. To achieve this, every third point of each group (coughs and non-coughs) was taken for validation while the remainder made up the training data. It was considered necessary to then reduce the number of the non-cough events to be approximately the same as the number of cough events to ensure even training and to avoid the neural network becoming over-trained on non-cough events. This was achieved by calculating the number of cough events and removing the appropriate number of non-cough events, evenly-distributed through the dataset. Throughout these steps, the labelling information is maintained including the index number of each event, assigned during the identification of the sound events and also the event type (cough or non-cough).

The result was a working function which successfully separated coughs and non-coughs into both training and validation datasets. The operation was validated by taking the outputted datasets from cough studies A1 and C1 confirming that each cough and non-cough had been appropriately labelled.

3.3.3.2.3 Feature extraction

The next stage was to calculate the spectral coefficients of the sound events in order to create the feature vectors. It was decided to continue with the format used in the existing software as a starting point for this work. Neither MFCC method nor LPC have previously been documented as being applied to the analysis of cough sounds, with the exception of a study by Thorpe *et al.* in which cepstral coeffcients were compared to several other methods in their ability to distinguish asthmatic and non-asthmatic coughs[56]. It was confirmed that these methods could be used effectively in such an application. Cepstral coefficients have also been widely and successfully used for the purpose of speech recognition.

3.3.3.2.3.1 Mel-frequency cepstral coefficients (MFCC)

The calculation of the MFCCs has been previously described in Section 1.3.3.2.1. The MFCCs represent the amount of each frequency present at each windowed portion for the duration of the sound event. The function calculates the frequency components of each window and presents the result as a row of coefficients; the results of each window are then stacked. For this reason, the resulting data are two-dimensional with a uniform length for the dimension representing the individual frequencies and a variable length dimension depending on the duration of the sound event under analysis. As the length of a window is 256 samples (where a sample is 1 / 44100 sec), with an overlap of 128 samples, one second of audio data can generate approximately 344 windows. Considering that cough events are highly variable in duration, no uniform duration size can be assigned for analysis purposes.

In the existing software, data were presented to the neural network in this two-dimensional format, with each successive window being presented in sequence[48]. However, the neural network used was not one that recognised two-dimensional data; it was therefore taking each row of coefficients as a separate dataset. This is effectively losing the important temporal information associated with the frequency change over the duration of a cough event.

As the coefficients are inherently of variable size, a method was required to reduce the matrices of the various sound events down to a uniform duration.

One option was changing the window size of the MFCC calculation to a sufficiently large value such that cough events would be captured by just one window. Any events smaller than this set window size were then padded out with zeros to maintain uniformity. However, following investigation into this, it transpired that as the method only calculated one value for each frequency, it was actually giving an average value of the frequency components over the cough duration. This resulted in non-cough sounds with the same frequency content as a cough, although different temporal patterns, possessing similar coefficients to those of a cough; thus leading to incorrect classifications. This method was therefore as lacking in temporal information as the first approach.

The next option was the generation of the MFCCs in the original way and the reduction to a uniform matrix by use of PCA [See Section 3.3.3.2.6].

A further option was to investigate the use of correlation coefficients in order to study the change in frequency content over the course of the sound event and also to generate a uniform matrix [See Section 3.3.3.2.7].

3.3.3.2.3.2 Linear predictive coding (LPC)

It was considered necessary to investigate the value of using LPC coefficients in addition to the MFCCs. A contribution chart for the 56 spectral coefficients used in the existing software along with a plot of the values of each of the spectral coefficients is shown in Figure 3.30.

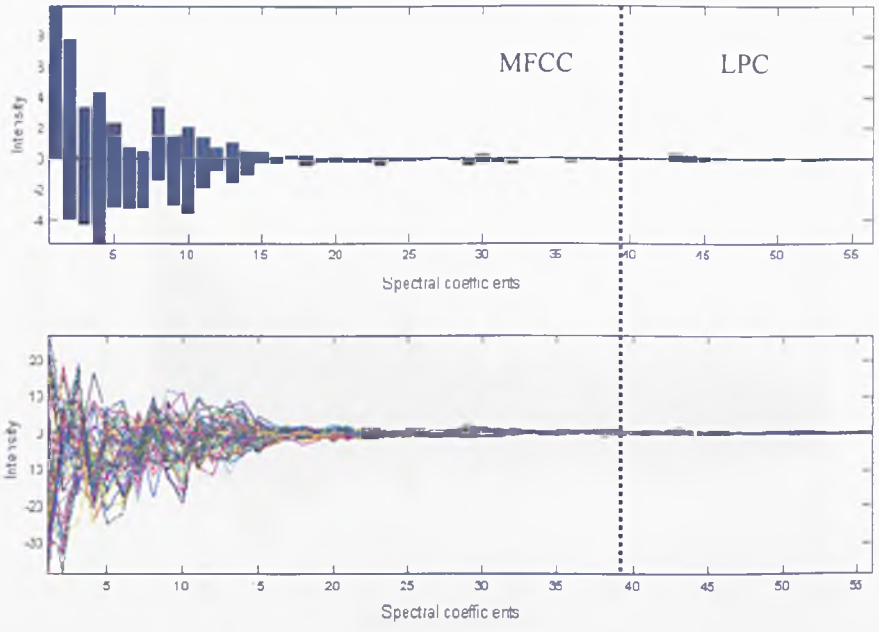
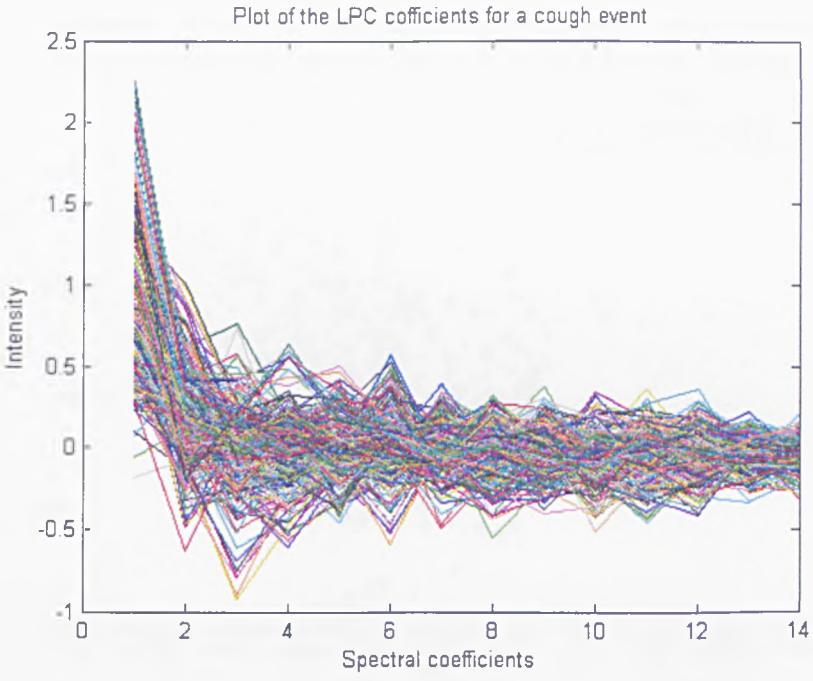


Figure 3.30 A contribution chart representing the contribution of each spectral coefficient (top) and a plot of the values of each spectral coefficient (bottom) for an example sound event; the first 42 coefficients are calculated by MFCC while the last 14 are calculated by LPC

The final 14 coefficients which are calculated by LPC are apparently not contributing any useful information to the description of the sound event. The problem with a data matrix in which each sample has a common factor, for example the final 14 coefficients possessing very little variation, is that it introduces the possibility that the neural network will use this similarity as a common feature to classify both coughs and non-coughs as belonging to the same group.

In addition, Figure 3.31(a) shows the LPC coefficients calculated for a cough event, while Figure 3.31(b) shows those calculated for a non-cough event; there is virtually no difference between the two groups. The final 14 LPC coefficients were therefore shown to be superfluous and not included in any further calculations.

a)



b)

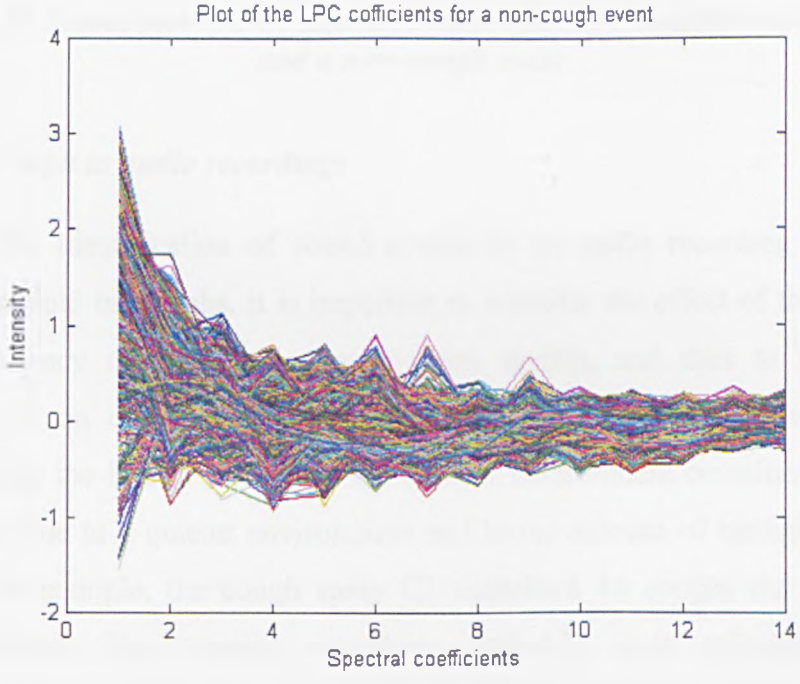


Figure 3.31 LPC spectral coefficients for a cough event (top) and LPC spectral coefficients for a non-cough event (bottom)

As a further test of similarity between the cough and non-cough coefficients, PCA was carried out on the data and the scores plots produced [See Figure 3.32]. The separation between the two groups is evidently not adequate.

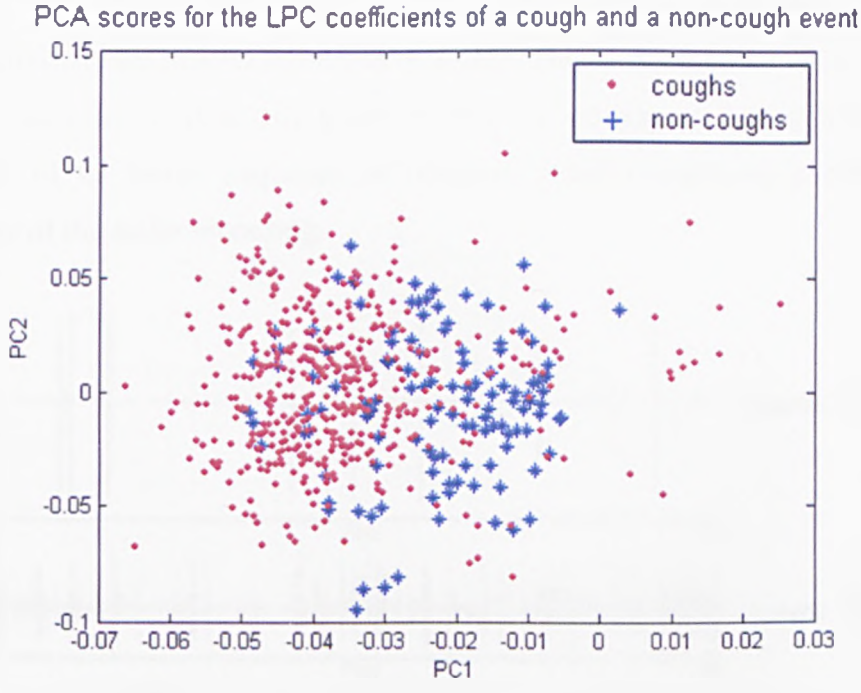


Figure 3.32 Scores plot of PC1 against PC2 for the LPC coefficients of a cough and a non-cough event

3.3.3.2.4 Cough in audio recordings

Following the identification of sound events in an audio recording, of which a proportion should be coughs, it is important to consider the effect of this proportion on the efficiency of any applied recognition system, and thus to highlight the importance of an effective data pre-treatment stage. The cough studies C1-C8, recorded using the line-in method in the clinical environment contained as much as 40% coughs due to a quieter environment and lower amount of background sounds detected; for example, the cough study C1 contained 44 coughs out of 98 sound events detected. The original recordings (A1-A3) were calculated to have approximately 6% coughs due to a greater degree of background noise; an example is the A1 study which contained 53 coughs out of 849 sound events. This is illustrated in Figure 3.33 (a) and (b) which are the time amplitude plots of 20-minute sections of C1 and A1 respectively. Both contain approximately 40 coughs, although there is a smaller amount of background noise in (a) and the coughs make up the majority of the higher amplitude peaks, whereas in (b) there are many more additional sounds. The 24 hour recording C9 contained as low as 1% coughs; this is both due to the fact

that the patient did not cough frequently (23 coughs in one hour) and also the fact that a high degree of background noise was recorded. This is represented in Figure 3.33(c) and the magnified portion (d) which shows the coughs to be of much smaller amplitude, of far lower frequency of occurrence and comprising a much smaller percentage of the audio recording.

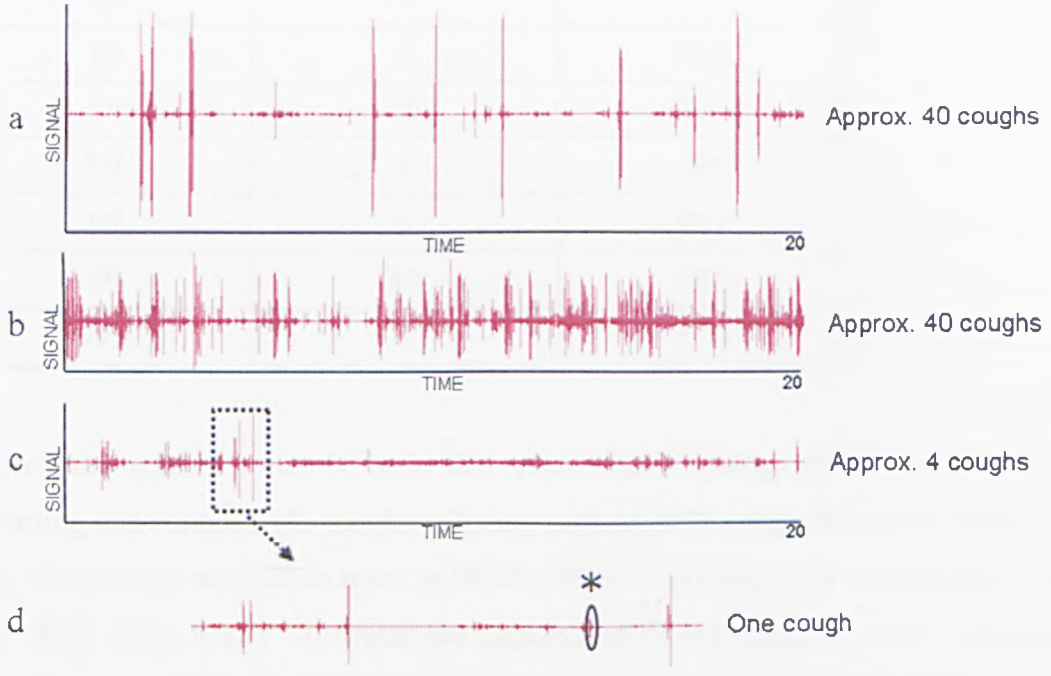


Figure 3.33 Time-amplitude plots of 20-minute portions of three different audio recordings: C1 (a), A1 (b) and C9 (c) and a magnified portion of C9 (d) illustrating the large differences between studies

Regardless of the reasons for the varying percentages of cough present in different recordings, the effect of this figure on the cough detection accuracy should not be overlooked. Using a contingency table, the actual accuracy of cough recognition, taking into account system efficiency and percentage composition, was calculated [Table 3-1].

Table 3-1 Summary of the percentage of coughs present in an audio recording and the theoretical predicted accuracy of detecting a cough correctly at various system efficiencies

SYSTEM EFFICIENCY	COUGH COMPOSITION (%)	COUGH ACCURACY (%)
80	1	3.9
80	6	20.3
80	40	72.7
99	1	50
99	6	86.3
99	40	98.5
99.8	1	83.4

From these results it can be seen that even with a system efficiency of 99%, a recording that contains 1% coughs will only achieve 50% cough detection accuracy at best. In fact, the recognition must be 99.8% efficient even to get a cough accuracy of over 80%. This clearly illustrates the importance of removing as many non-cough sound events prior to the pattern recognition stage as possible in order to increase the percentage of coughs and thus achieve improved cough accuracy results.

3.3.3.2.5 Data pre-treatment

Frequently used data pre-treatment methods include mean-centring, auto-scaling or smoothing techniques such as Savitsky-Golay smoothing. However, it was considered that, for this application, techniques which smooth or remove variation in the data would be inappropriate. The ultimate aim is to carry out an effective separation step on the two classes of data; coughs and non-coughs. To remove variation or smooth the signal would remove characteristics which could potentially serve to highlight the differences between the two groups. To confirm this theory, a comparison was made between the PCA scores of cepstral coefficients pre-treated by mean-centring, auto-scaling and Savitsky-Golay smoothing and PCA scores of the untreated cepstral coeffcients. As expected, there was a lesser degree of separation of the cough and

non-cough datasets achieved following data pre-treatment compared to the untreated cepstral coefficients.

3.3.3.2.6 Principal component analysis

The feature extraction step creates a large amount of data, not all of which will be valuable to the pattern recognition step. The purpose of applying PCA is to identify and remove the coefficients possessing the least useful information to the dataset. In this way, PCA both reduces the data to a more manageable size whilst also removing the data which will add the least value to the dataset. The desired result following this data-pre-treatment stage is two groups of data which are maximally correlated within the group and maximally different between groups, thus rendering the pattern recognition stage much simpler. PCA has been used in many applications however, due to the limited methods used for the analysis of cough sounds it has not been extensively studied for this purpose[59].

Following MFCC calculation, the data for each sound event consists of 42 columns containing the coefficients and a variable number of rows dependent on the duration of the sound event (one row for every window of 256 samples).

3.3.3.2.6.1 Reduction of the loadings

An eigen value plot was produced for the spectral coefficients of both coughs and non-coughs following PCA in order to establish the amount of variance possessed by each coefficient [See Figure 3.34].

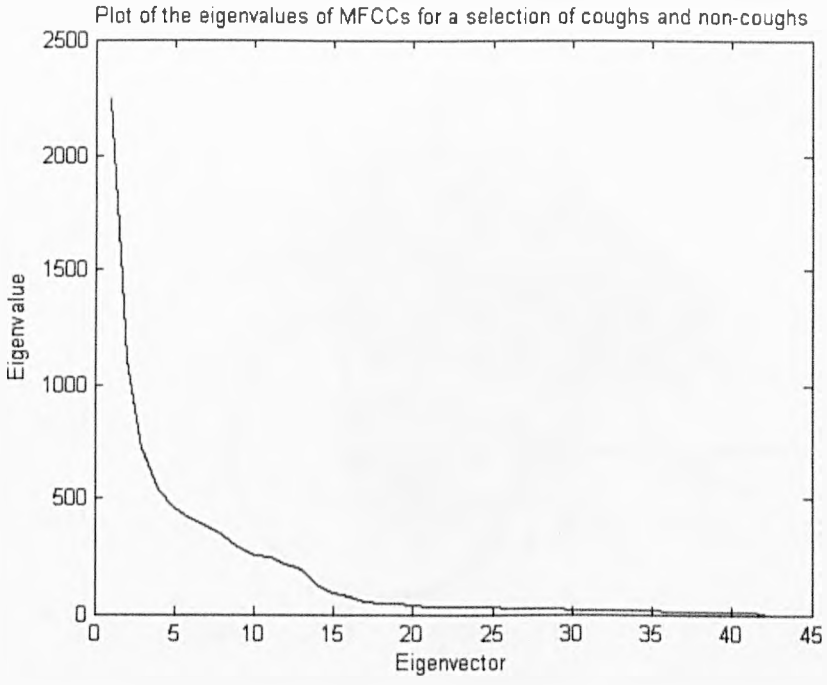


Figure 3.34 Eigen values of the MFCCs for a selection of coughs and non-coughs; this shows that the first 15 PCs possess the most variance

Removing the loadings possessing the least variance avoids similar data occurring in both datsets, which would then increase the possibility that the neural network will find a characteristic common to both groups. From the eigen value information, it was decided to use the first 15 PCs. The scores matrix of the first 15 PCs was plotted for the coughs and non-coughs [See Figure 3.35]. This plot shows a reasonable degree of separation between the two groups, with a small amount of overlap. However, there is not a sufficient amount of distance between the two groups to make them appear separate. This suggests that simply applying the loadings reduction to the MFCCs is not enough to create two distinguishable groups.

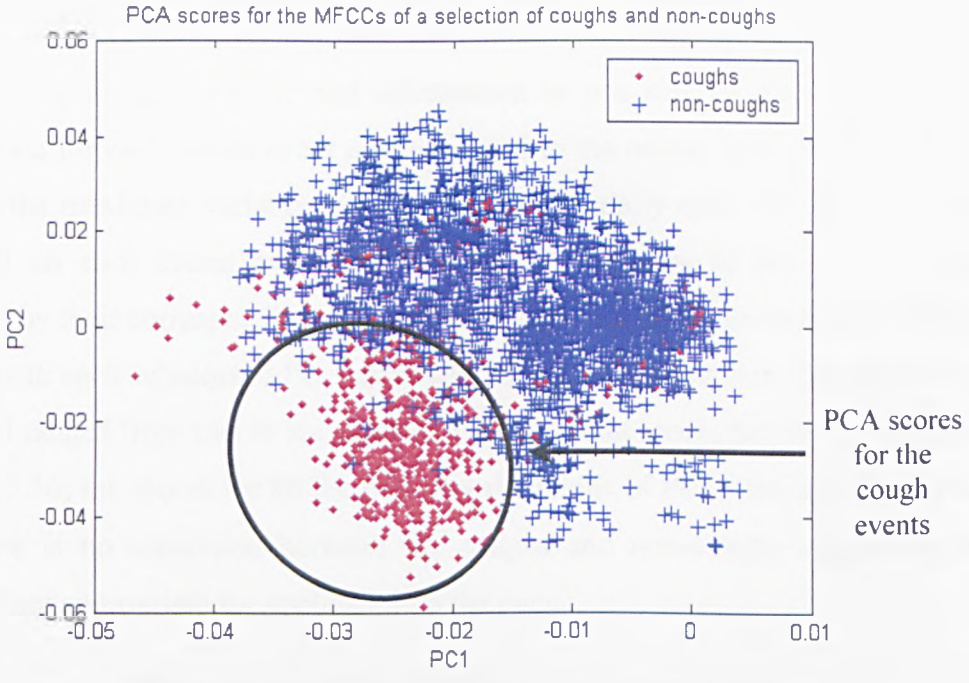


Figure 3.35 PCA scores of the MFCCs calculated from a selection of coughs and non-coughs

3.3.3.2.6.2 Data reduction to multiple rows

At this stage, the dataset was still in a non-uniform state due to the multiple rows of coefficients for each sound event, the number of which is time-dependent. To instil uniformity into the dataset for presentation to the neural network, the loadings were reduced to create a uniform dataset size according to the number of frames in the smallest sound event. However, the shortest sound event gave rise to only one row of coefficients, which resulted in very little information for the longer duration events, thus leading to an unsuccessful result.

3.3.3.2.6.3 Data reduction to uniform size

To capture the maximum variance in the dataset whilst also introducing uniformity, the MFCC data was squared by its transpose such that the resultant data matrix was 15 by 15. SVD was then performed on the data followed by calculation of the square root. The loadings were then multiplied by the corresponding eigen values for scaling. However, once again this results in a matrix that has multiple rows for presentation to the neural network.

3.3.3.2.6.4 Data reduction to single row

To capture maximum variance and information in one row of data, such that all relevant data for each sound event can be passed to the neural network together, the PCs with the maximum variance were ordered sequentially onto one row. SVD was performed on each sound event and the first 15 loadings of the first PC were multiplied by their corresponding eigen value to scale according to variance. This was repeated with each subsequent PC, which was added to the same line. The numbers of PCs tested ranged from two to six. A plot of the resultant scores for two PCs is given in Figure 3.36; the scores for all the other combinations of PCs were identical to this plot. There is no separation between the coughs and non-coughs suggesting this method is not appropriate for application to the data.

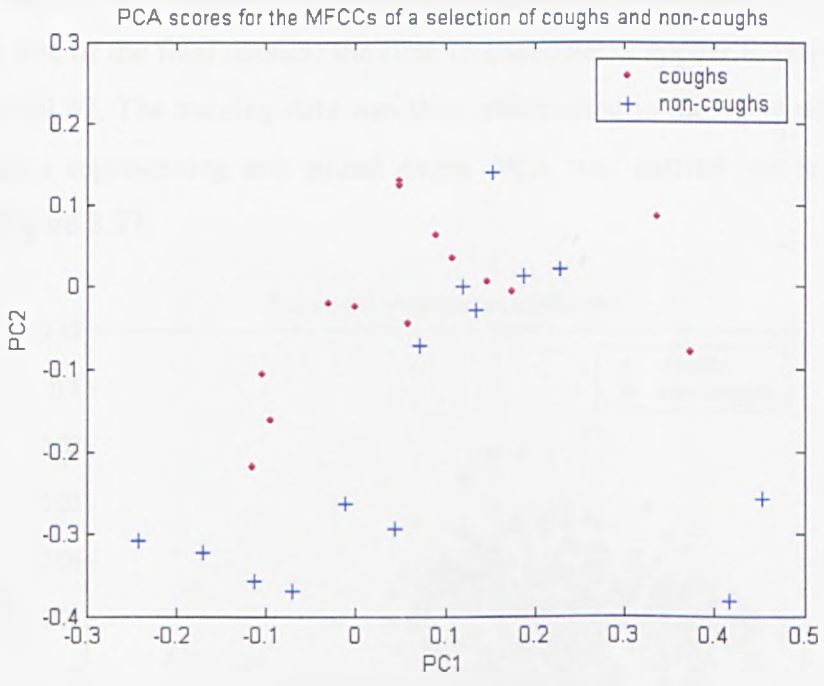


Figure 3.36 PCA scores of two PCs represented on one row of data

3.3.3.2.7 Correlation coefficients

The aim of this experiment was to investigate if there was any correlation within the frequency spectrum over the duration of a cough event. One of the problems with the calculation of MFCCs is that the outputs are divided into separate rows for each frame of the sound event; thus the temporal relationship of the frequency content

within a sound event is lost. Correlation coefficients provide a solution to this problem by identifying any correlation within a matrix.

Calculating the correlation between two variables gives a summary of the strength of the linear association between the variables. The correlation coefficient is a number between -1 and +1. A positive correlation indicates that the variables move in the same direction, while a negative correlation means that they move in the opposite direction. The closer the correlation coefficient is to either -1 or +1 indicates the strength of the correlation. Correlation coefficients were calculated to identify any correlation within the frequency domain between frequencies as they progressed through the sound event.

The correlation coefficients of the first 15 spectral coefficients were calculated and compiled into a one row matrix. A reduced number of coefficients were used so as to reduce the size of the final dataset; the first 15 coefficients appear to possess the most variance of all 42. The training data was then constructed in the usual way, with each row this time representing one sound event. PCA was carried out and the scores shown in Figure 3.37.

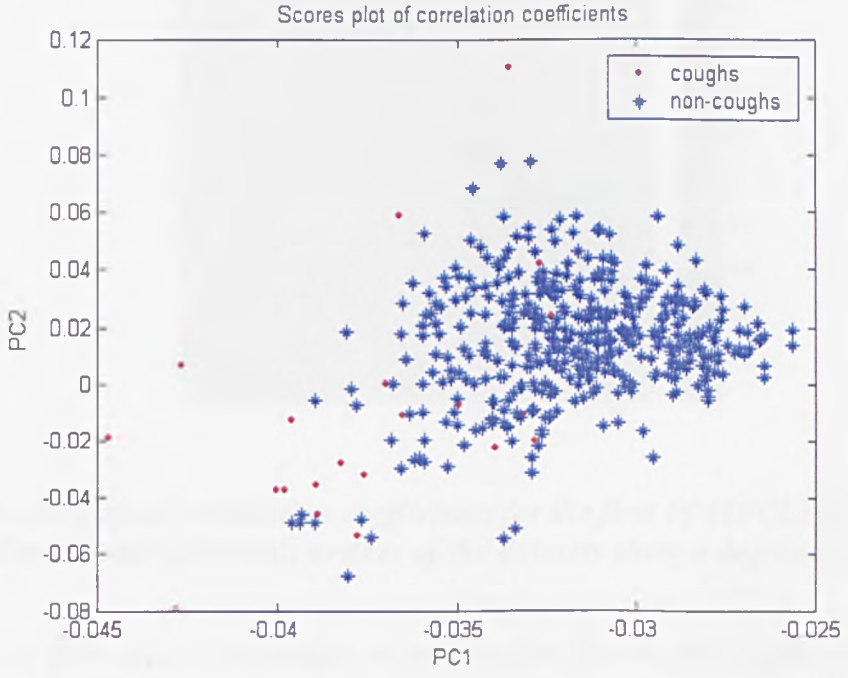


Figure 3.37 PCA scores plot of PC1 against PC2 for the correlation coefficients

Again, the cough event data possesses much more variation than the non-cough data, making the problem harder to solve. However, there appears to be more distinction between the two groups.

Coloured surface plots representing the relationship between the correlation coefficients were produced.

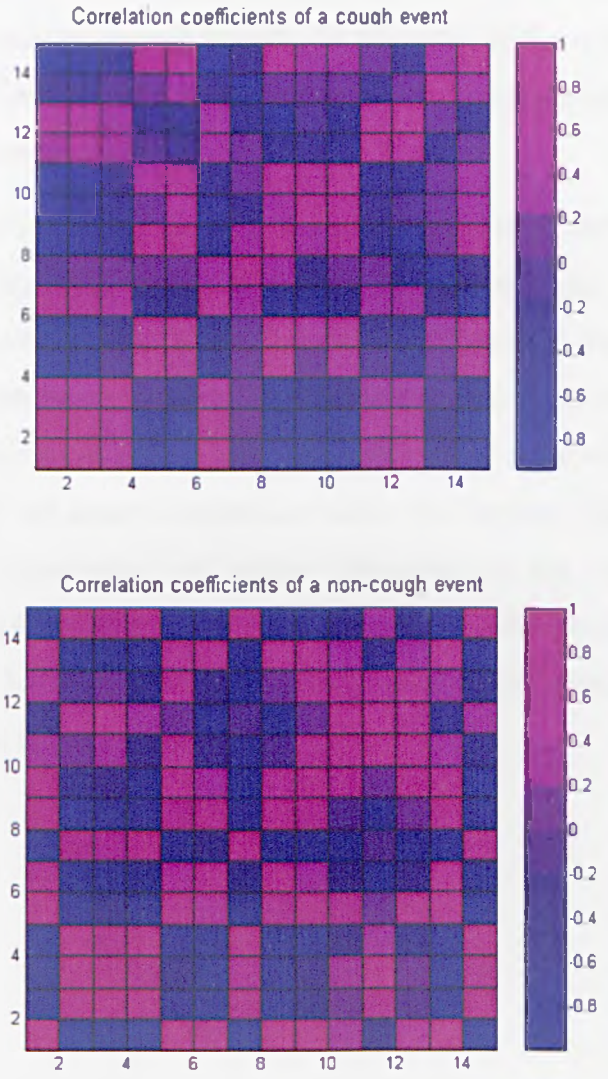


Figure 3.38 Typical correlation coefficients for the first 15 MFCCs for a cough (top) and non-cough (bottom); neither of the datasets show a degree of correlation

None of the plots, either for coughs or non-coughs showed any significant degree of correlation; where correlation is indicated by a cluster of positive correlation (i.e. values close to +1) on the diagonal, whilst the remaining coefficients have relatively

low values. In addition, the PCA scores plots do not show any significant separation of the two datasets to provide adequate training data to the neural network.

3.3.3.2.8 Voice separation

As speech sounds are one of the most common of the interfering sounds present in the cough studies, it makes sense to apply a technique for cough recognition that also has success with differentiating speech sounds. In this way, if the technique is adept at handling speech differences, it has more chance of distinguishing between speech sounds and cough sounds.

The initial stage involved taking the FFT of a sound event such that the output was an intensity value for each frequency at each time interval. The intensity of each frequency was then averaged for the entire duration of the sound event giving a vector of mean values for frequency intensity. Applying this to a male and a female sound gives the output as shown in Figure 3.39. As expected, the male voice contains much higher intensities of the lower frequencies while the female voice contains lower amounts of these frequencies and higher intensities of the higher frequencies. Applying PCA to 14 male and 14 female vocal sounds gave the PCA scores illustrated in Figure 3.40. Thus, from frequency information alone, male and female voices show potential for separation.

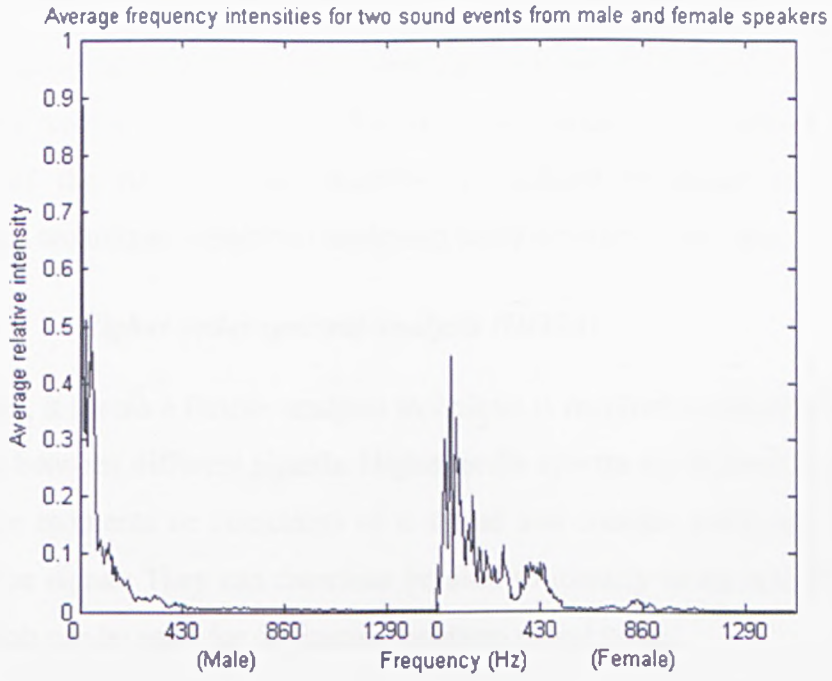


Figure 3.39 Frequency intensities averaged over a voiced sound event for a male and female speaker; the male voice contains high intensities of the lower frequencies whilst the female voice is spread over a wider range and contains more of the higher frequencies

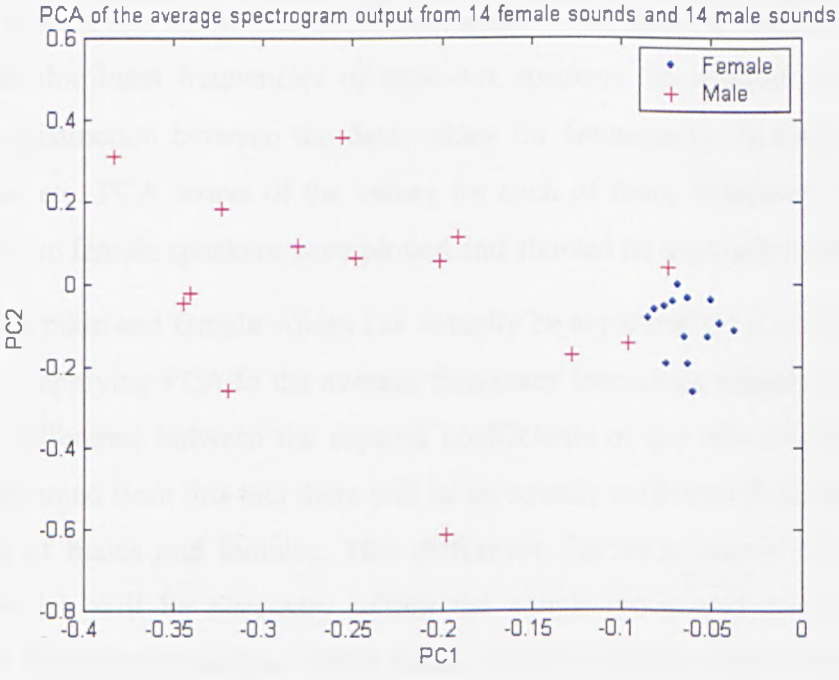


Figure 3.40 PCA scores of the average frequency intensities for 14 male sounds and 14 female sounds

While it was demonstrated that PCA could separate a mixture of male and female voices, it was not able to successfully separate speakers of the same sex. PCA scores of same-sex voices pre-treated in the same way were superimposed due to the similarity of the data. It was therefore considered necessary to try a more sophisticated technique, capable of analysing more features of the data.

3.3.3.2.9 Higher order spectral analysis (HOSA)

At this stage, it seems a further analysis technique is required to identify key spectral differences between different signals. Higher-order spectra are defined in terms of the higher-order moments or cumulants of a signal and contain additional information regarding the signal. They can therefore be used to identify characteristic features in signals which can be used for distinction between sound types.

3.3.3.2.9.1 Voice

HOSA was initially used to separate the same vowel sounds as spoken by a male and female speaker. Using the dominant frequencies alone, this appeared achievable as there was clear separation between the two. However, attempting to apply the same technique to same sex voices was not as successful. There was virtually no distinction between the dominant frequencies of same-sex speakers. In addition, there was no patterns or distinction between the data values for Gaussianity, linearity, skewness and kurtosis etc. PCA scores of the values for each of these calculated components for each of two female speakers were plotted and showed no separation or clustering.

The fact that male and female voices can actually be separated by a relatively simple technique of applying PCA to the average frequency intensities suggests that there is sufficient difference between the cepstral coefficients of the two sound types. It is therefore assumed from this that there will be an equally sufficient difference between the coughs of males and females. This difference further increases the amount of variation which will be necessary within the cough group and thus adds to the problem of finding a technique which will be able to classify such a highly varied dataset.

3.3.3.2.9.2 Cough

HOSA was used in combination with a LVQ neural network to attempt to distinguish coughs from non-coughs. A selection of HOSA calculations were made and a neural network was designed, having a node for each HOSA result such that the first four nodes were always used for the linearity test data, the next four were always used for the Gaussianity test data, the next three for fundamental frequencies and the final three for the remaining data [As described in Section 2.3.4.2.7.3]. To start with, the same data was used for training as for validation, with the intention of creating separate datasets if the method appeared to have a degree of success. This re-presentation of the same data to the neural network should have given it an advantage and should therefore achieve good separation, if it was going to be possible. Following the training, the network identified all eight coughs correctly, however, it also labelled six of the non-coughs as coughs, resulting in a validation accuracy of 55%. Considering the fact that the data had already been presented to the network, thus necessitating no generalisation, this suggested that the HOSA technique was not going to be a successful one for this application.

A further point is that HOSA is not an easy technique to optimise since it requires user input following the observation of power spectra.

3.3.4 Optimum data pre-treatment conditions

Several techniques had been attempted at this point to achieve a suitable degree of separation between the two groups, however, so far total separation had not been successful. From the results of the previous experiments it was decided to calculate the MFCCs of the sound events as carried out in the existing method. These were then reduced by SVD, as described in Section 2.3.4.2.5.1 to give a data matrix for application to the neural network. Although the data has not been shown to be fully separable by PCA, ANNs have an ability to recognise patterns in data that are not always obvious when viewed by other methods. It was therefore decided to assess the performance of the ANN on the existing data, which is at the highest degree of separation that can be attained.

3.3.5 Sound Separation

Following the pre-treatment of the audio data, the individual sounds require classification as either coughs or non-coughs. For this process, a pattern recognition step is required in order to identify characteristic features in the data which will enable separation. For this application, ANNs were chosen for several reasons. Firstly, it would be impossible to create reference patterns for all the sound events that could be encountered in a 24 hour study in order to carry out an exact pattern comparison stage. For this reason, the ANNs ability to generalise is highly beneficial as it allows it to recognise data that is similar to the reference data with which it has been trained, without it having to be exact. Secondly, due to the non-uniformity of cough events and given the large amount of variation within the two datasets, statistical analysis would be highly difficult; an ANN is able to identify patterns and characteristics in the data which would be otherwise unobservable by simple inspection. Thirdly, the process of applying the ANN can be automated and should be relatively efficient. Finally, the use of a PNN in the existing work was shown to be highly successful indicating a high chance of success with the new data.

ANNs have been successfully applied to pig cough recognition by Van Hirtum *et al.*, several times since 2001. The group have employed self-organising maps (SOMs) and PNNs to the calculated PSD data with success rates of over 90%[111-114].

3.3.5.1 Development of neural network

A PNN, as used in the existing data, is limited to one round of training and must therefore be presented with an entire database at once. This type of training is not concurrent with the ability to update the neural network at points in the future if an unusual cough type is encountered and the neural network has difficulty recognising it. It was therefore decided to change the ANN type; two types were evaluated, feed-forward (FF) and learning vector quantisation (LVQ).

3.3.5.1.1 Feed-forward (FF) neural network

The FF neural network was created by the Matlab function *newff* to design specifications according to desired numbers of hidden layers, number of neurons in

each hidden layer, transfer functions of each layer, training functions, learning functions and performance functions. The purpose of the neural network is ultimately to achieve the best convergence between the input and the target output. Thus, it is these parameters that require optimising in order to achieve the best performance. To determine the optimum number of neurons in the hidden layer, a balance needs to be found between the number of input nodes, which would result in too many connections and would lead to a lengthy training process, and the number of output nodes which would render the hidden layer ineffective. The transfer functions, as discussed in Section 1.4.3.1.2, each have merits according to the required application; for instance, the use of an ANN to solve a linear problem would only require a linear transfer function. Training functions and learning functions each have merits and drawbacks and are usually a compromise between time and performance. Performance functions assess the ability of the ANN to achieve the desired classification or recognition; different functions base performance on different factors.

Several parameters were investigated at various levels in order to assess the best architecture for the network. The network was trained with real data and then validated on additional unseen real data from the cough studies. However, as a result of these experiments, there was no clear optimum set of conditions that appeared to give the best results; in fact at this stage, the validation errors were relatively high and it was decided to try a different type of network.

3.3.5.1.2 Learning vector quantisation (LVQ) neural network

The LVQ neural network was created by the Matlab function *newlvq*. The number of neurons in the hidden layer was specified and the network was tested as described in the previous experiment. The cough study used for validation was C5. Comparison of the results from the neural network with those obtained by listening to the WAV file were positive; sensitivity was calculated at 84%, specificity at 95% and accuracy at 88%. These were considered to be highly acceptable for the application and a positive result. However, when attempting to obtain the same results from other cough studies, it was not shown to be repeatable. Cough study C5, as previously discussed, contained approximately 40% coughs and very little ambient noise due to the patient

being limited to the hospital grounds and spending a lot of time within the clinic. This result, therefore, does not reflect on the ability of the neural network to perform successful cough recognition on the 24 hour cough studies. Further testing of this is discussed in Section 2.3.6.4.

3.3.5.1.2.1 Testing the neural networks with simulated data

The neural network required testing to ensure that the type and architecture were suitable to solve the problem at hand, which was to distinguish between two groups of spectral coefficients. To achieve this, a simulated data set was generated. The simulated dataset bore no particular resemblance to the real data as it was simply intended to represent a dataset containing two clear sub-groups. The simulated data consisted of 400 spectra in total. 100 contained the first spectral peak while another 100 contained the second spectral peak; each with peak maxima at a suitable resolution and containing a reasonable amount of noise. A further 200 spectra were composed equally of each of the spectral peaks along with the other spectral peak at a much smaller magnitude to represent contamination. The parameters of the data were maintained as close to those of the original data set as possible. One quarter of the data was removed by sampling at regular intervals throughout the dataset to form the validation dataset. The remaining data was to be used for the neural network training.

PCA was then applied to the training data set in the form of SVD. It was decided to use the first five PCs to represent the variation in the data. The first five loadings values were then multiplied by the training spectra to give the dataset necessary to train the network. The validation spectra were treated in exactly the same way by being multiplied by the same values.

A LVQ neural network was then created. Five input neurons were created for the five PCs and three neurons were specified for the hidden layer; this was to reproduce the network architecture used for the spectral coefficients. All of these parameters are as defined in the network used for the original dataset.

One epoch of training, which is the presentation of each data point to the network once, yielded a validation error of 38%, while four epochs yielded 2% error, therefore 98% accuracy [See Figure 3.41]. Further training lead to an increase in the validation

error while the training error continued to improve; this is a typical result of the network becoming over-trained. At over-training, the network learns to recognise the training data so well it loses its ability to generalise for new data; thus, while the training error decreases, the validation error increases. It is therefore important to monitor the outputs and errors of the neural network during training to prevent this occurrence and to avoid attributing a poor validation result to the wrong cause. At over-training, the network is no longer suitable for use and must be re-trained from the start.

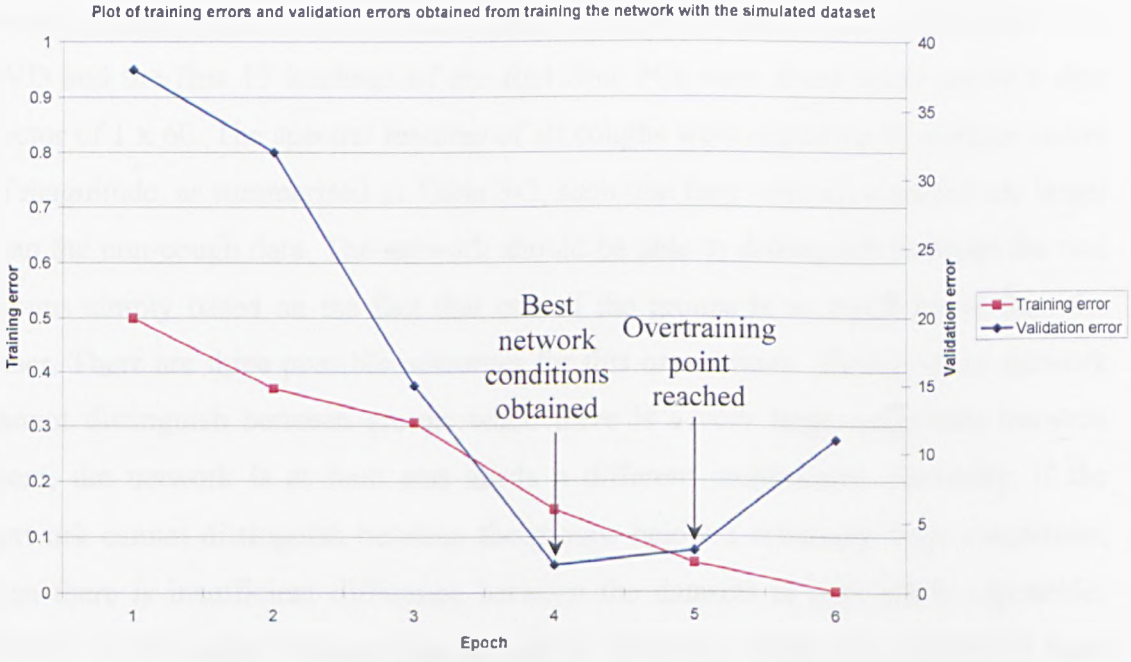


Figure 3.41 Plot showing the decrease in validation error before reaching a minimum and increasing again; at the same time the training error gradually decreases regardless of the validation error

This experiment showed that the neural network is working and is capable of classifying two groups of different spectral information to a high degree of accuracy.

3.3.5.1.2.2 Training the neural networks with real data contaminated with artificial noise

To allow the network to generalise from data it receives and tolerate noise in real data, it can be useful to train the neural network with both the original data and the

same data contaminated with artificial noise. A FF neural network with one hidden layer and a logarithmic sinusoid transfer function between each layer was created. Applying this to the cough study C9, there was almost no difference in results obtained from the network trained on data with and without noise added sequentially and those obtained from the network trained on data alone.

3.3.5.1.2.3 Testing the neural network's suitability to the real data

A further test for the neural network is its suitability to the data that is to be classified. Twenty cough and non-cough spectral features taken from the cough study C9_part1 were equally separated into training and validation datasets. Data was reduced by SVD and the first 15 loadings of the first four PCs were lined up to create a data vector of 1 x 60. The spectral features of all coughs were scaled up by various orders of magnitude, as summarised in Table 3-2, such that they were all a magnitude larger than the non-cough data. The network should be able to distinguish between the two groups simply based on the fact that one of the groups is so much larger than the other. There are three possible outcomes for this experiment: Firstly, if the network cannot distinguish between groups when there is a very large magnitude between them, the network is at fault and needs a different architecture. Secondly, if the network cannot distinguish between the groups below a relatively large magnitude, then there is insufficient difference between the datasets to ever allow separation. Thirdly, if the scalar between the datasets is relatively small, the network is most probably inadequately trained or requires small alterations to its architecture. The performance of a LVQ neural network was then tested at these values and the cough, non-cough and overall validation accuracies were calculated.

When using the original data, the cough validation was shown to be 100% accurate while the non-cough validation accuracy was at 40%. However, for every magnitude greater than 1, the non-cough group was repeatedly classified with 100% accuracy while the cough group worsened [See Table 3-2]. This unexpected result could be due to the relative variations within each dataset. The fact that scaling up the cough dataset improved the recognition accuracy for the non-cough group suggests that by making the cough data significantly out of the non-cough data range, the non-cough data was much more easily classified. The results also show that in their original

state, the non-cough data recognition is hindered by the presence of the cough data. Ultimately, no matter how separate the cough data is to the non-cough sounds i.e. even if cough was the only dataset present, the inherent variation within the group makes it very difficult to be correctly grouped by use of a neural network. This indicates a significant difficulty for the application of neural networks to this problem.

Table 3-2 Results of testing the neural networks ability to classify cough and non-cough differentiated by scale; magnitude applied to cough dataset

Magnitude (x original data)	Cough validation accuracy (%)	Non-cough validation accuracy (%)	Overall validation accuracy (%)
1	100	40	70
10	0	100	50
100	40	100	70
1000	40	100	70
10,000	40	100	70
100,000	40	100	70

In order to further test the magnitude effect, the same experiment was carried out, this time scaling up the non-cough dataset. The results are summarised in Table 3-3.

Table 3-3 Results of testing the neural networks ability to classify cough and non-cough differentiated by scale; magnitude applied to non-cough dataset

Magnitude (x original data)	Cough validation accuracy (%)	Non-cough validation accuracy (%)	Overall validation accuracy (%)
1	0	100	50
10	100	0	50
100	100	0	50
1000	100	0	50
10,000	100	40	70
100,000	100	0	50

Observation of the scaled up data compared to the normal data shows a significant difference that the neural network would be expected to recognise. To test whether or not the network was performing classification based entirely on patterns and ignoring any magnitude information, a simulated dataset was created. All the data followed the same basic pattern and the two groups were differentiated only by magnitude. Applying the same LVQ neural network to the data gave 100% classification accuracy.

Two further questions that required an answer were, firstly whether or not the use of the first four PCs was introducing too much similarity between the datasets, thus making them harder to differentiate and secondly, whether the fact that the dataset contained both positive and negative values was affecting classification. To investigate this, only the first 15 loadings of the first PC were used, these were then squared and square rooted to find the absolute value of the spectral coefficients. The results are summarised in Table 3-4.

Table 3-4 Results of testing the neural networks ability to classify cough and non-cough differentiated by scale; magnitude applied to non-cough dataset following further data pre-treatment

Magnitude (x original data)	Cough validation accuracy (%)	Non-cough validation accuracy (%)	Overall validation accuracy (%)
1	0	100	50
10	100	40	70
100	100	40	70
1000	100	40	70
10,000	100	40	70
100,000	100	40	70

Although an improvement is seen over the previous attempts, the network is still unable to identify the differences between the two datasets, even when one is 100,000 times greater than the other. These results appear to indicate that the data is variance based, having been treated with PCA, and that the neural network is not recognising magnitude differences. However, the magnitude information could be as important as

the variance in performing cough recognition, and could be incorporated by using RMS plots as a further input to the network, along with the frequency information.

3.3.5.1.2.4 Number of epochs

The number of epochs used for presentation of the data to the neural network was tested. The Matlab function which coordinates the training of ANNs gives a plot of the progressive training errors. Once the training errors reach a minimum value and either remain at that level or fluctuate slightly above it, it is considered that over-training has occurred. Observation of these dynamic plots therefore gives an indication of when training has reached a maximum number of epochs. Table 3-5 and Table 3-8 summarise the validation accuracies achieved from training with rounds of 25 and 100 epochs respectively. As is apparent from these results, training with 100 epochs gives slightly better validation accuracies, although there is no massive improvement. It has, however, been observed that there was no advantage in taking training above 200 epochs since any minimum training error for this data has been found well before this level.

3.3.5.1.2.5 Optimising network training using Design of Experiment

The aim of this experiment was to determine the optimum network training parameters. Design of experiment (DoE) was used to investigate the optimum combination of the number of PCs used to represent the data and the number of neurons in the hidden layer of a network.

The response to be determined was the cough validation accuracy achieved by the network. The resulting optimum conditions were 4 PCs used to train a network with 6 neurons in its hidden layer.

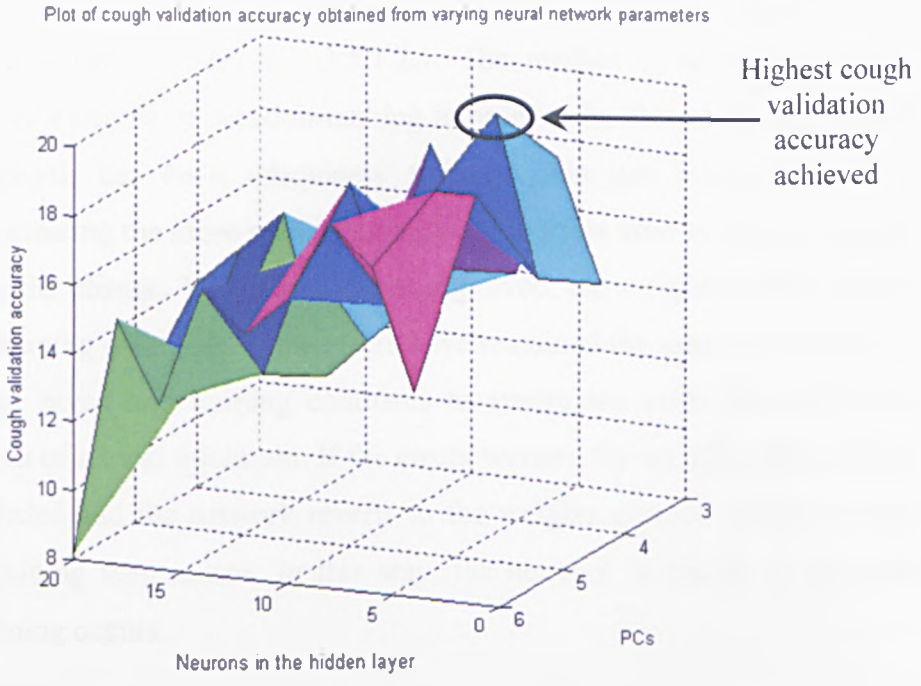


Figure 3.42 Validation accuracies obtained from the various combinations for the numbers of hidden neurons and PCs; the highest validation accuracy achieved relates to 4 PCs and 6 neurons in the hidden layer

3.3.5.2 Network training

The aim of this section was to create a function to automatically coordinate the training and validation of the neural network using the pre-treated data. The files *'lvqnet'* and *'ffnet'* [See Appendix A] create the LVQ and the FF neural networks respectively and automatically perform the training.

The entire dataset from an audio recording tends to contain a large amount of non-coughs compared to the relatively small number of coughs. The cough studies have shown the recordings to contain as little as 6% coughs, which means that the ANN could potentially classify all sound events as non-coughs and still achieve 94% accuracy. This has previously been explained in Section 3.3.3.2.4. For this reason, three separate validation accuracies are used to assess the performance of the ANN, a cough validation accuracy, which assesses the percentage of coughs have been correctly classified, a non-cough validation, which works in the same way for non-coughs, and an overall validation accuracy which includes all sound events.

A potential problem associated with training ANNs is the possibility of over-training, as described previously in Section 3.3.5.1.2.1. The method of automatic training therefore requires a degree of decision-making in order to be able to recognise when the neural network has been adequately trained. This has been achieved by automatically assessing the afore-mentioned validation errors after each set of epochs and monitoring the change. If the errors have improved, the weights of the network are saved and training continues. If the errors have remained the same, the weights of the network are saved and training continues to ensure the error has not simply reached a plateau or a local minimum. If the errors worsen, the weights of the current ANN are discarded and the network reverts to the weights created on the previous training run, training then ceases. In this way, the network is trained to the point before over-training occurs.

3.3.5.2.1 Random training

If an ANN is trained on several datasets sequentially, each for a reasonable number of epochs, it can adapt to the most recent dataset and essentially "forget" the earlier ones. To avoid this all the data must be presented completely randomly. The way in which training has been organised so far has presented one cough audio file at a time to the neural network. Therefore, to test whether this has had a detrimental effect on the network, all cough files were taken together and random training was performed. Results obtained were no different to those obtained from the sequential training as was previously carried out.

3.3.5.3 Network simulation

To further automate the system, an automatic method for the application of audio data to the saved neural network was created. The function *netprocess* takes the WAV file of a previously unprocessed audio recording and firstly identifies the location of the valid sound events. Working on 15 minute segments, these are then pre-treated in the same way as the training and validation data used to create the ANN; as described in Section 3.3.4. The saved ANN is then loaded and the data is passed to the network for classification. The ANN will give a two-column output where the first column represents the cough class and the second column represents the non-cough class; an

output of '1' in either column is a positive classification for that group. The results are then saved and can be inspected by use of the GUI, as detailed in Section 1.6.2.1.3. The GUI is set to load the new data as classified by the neural network and then graphically represent the decision output for each sound event. Additionally, the function summarises the cough count for each 15 minute portion processed for the purpose of the graphical representation [See Section 3.3.7.1].

3.3.5.4 Analysis of audio recordings

The results of the 24 hour cough studies, as obtained from the application of the neural network were analysed. In each case, the first three hour portion of the recording was initially used to assess the networks performance for that subject. The neural network had been trained on data as summarised in Table 3-5 to give Network 1. This was the starting point used for the cough study analyses.

Table 3-5 Summary of data used to train the neural network to give Network 1

Audio File	Cough Count	Non-cough Count	Epochs	Validation accuracy (%)
A1	32	28	25	85
A2	19	14	50	97
A3	11	6	50	88
C1	20	29	50	92
C4	140	37	50	88

3.3.5.4.1 Cough study C9

The number of coughs in the first 60 minutes of the audio recording was manually counted in order to provide a point of comparison to measure the performance of the network. The results of this count is summarised in Table 3-6. The cough count figures are given as the total cough count along with the individual figures of double and single coughs; since the method sometimes has difficulty determining the exact cough count due to counting double coughs as singles, this was considered useful as a point of reference for any underestimating of coughs.

Table 3-6 Aurally counted coughs in cough study C9

Time segment (mins)	Cough Count	Consisting of	
		Double coughs	Single coughs
0-15	7	3	1
15-30	7	2	3
30-45	7	3	1
45-60	2	1	0
Total	23	9	5

The initial results given by the network showed a vast overestimation of cough counts, with counts of over 100 coughs. The majority of the wrong classifications were laughter and loud vocal sounds. Figure 3.43 is a spectrogram of a sample of speech and laughter that, as expected from the results of the cough studies, is a common interference in this recording. Study C9 was of a female subject, and as is apparent from the spectrogram, the frequencies of the interfering sounds are relatively high and may therefore possess similar cepstral coefficients to the cough sounds.

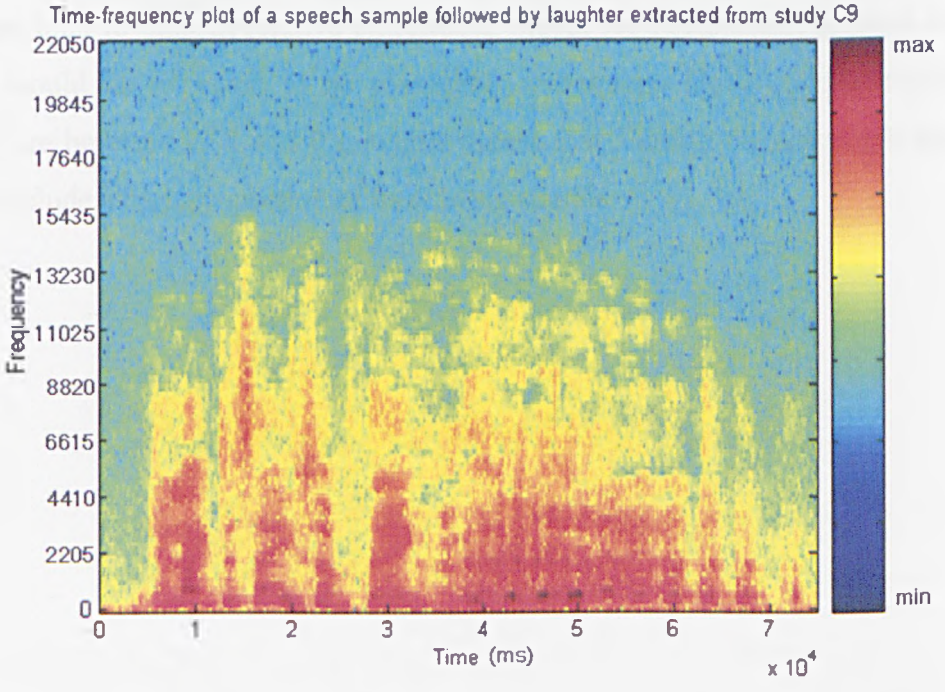


Figure 3.43 Frequency spectrogram of a sample of speech and laughter of the subject in the C9 study

It was considered that the network was possibly not being trained on enough non-coughs to represent the higher proportion they made up in the audio recording; it was calculated that this audio recording contains only 1% coughs compared to all other sound events. The network was re-trained with both twice and four times the number of non-coughs shown in Table 3-5, however this made no difference to the networks ability to correctly classify the coughs and non-coughs of this recording. It seemed the problem was the fact that the bank of non-cough sounds used to train the network was not broad enough to incorporate all other sounds. There is potential for the network to be periodically updated with "new" sounds, however, it needs to be established if this would provide a viable solution to the problem. To test this, five cough and five non-cough sounds were isolated and correctly labelled. These were then used to re-train the existing network, to give Network 2. The entire audio recording was then re-processed through the network. The results gave a cough count of 177 for the first hour alone, due to an increased number of vocal sounds being incorrectly classified as coughs.

Inspection of the interfering vocal and laughter sounds showed that they are generally of lower amplitude relative to the cough sounds, as illustrated in Figure 3.44. Voice averages 0.04 to 0.05 in relative amplitude; except for occurrences of loud shouting which would be expected to be reasonably infrequent. However, the majority of coughs are between 0.08 and 0.1, which means that altering the amplitude threshold could exclude a high proportion of interfering sounds.

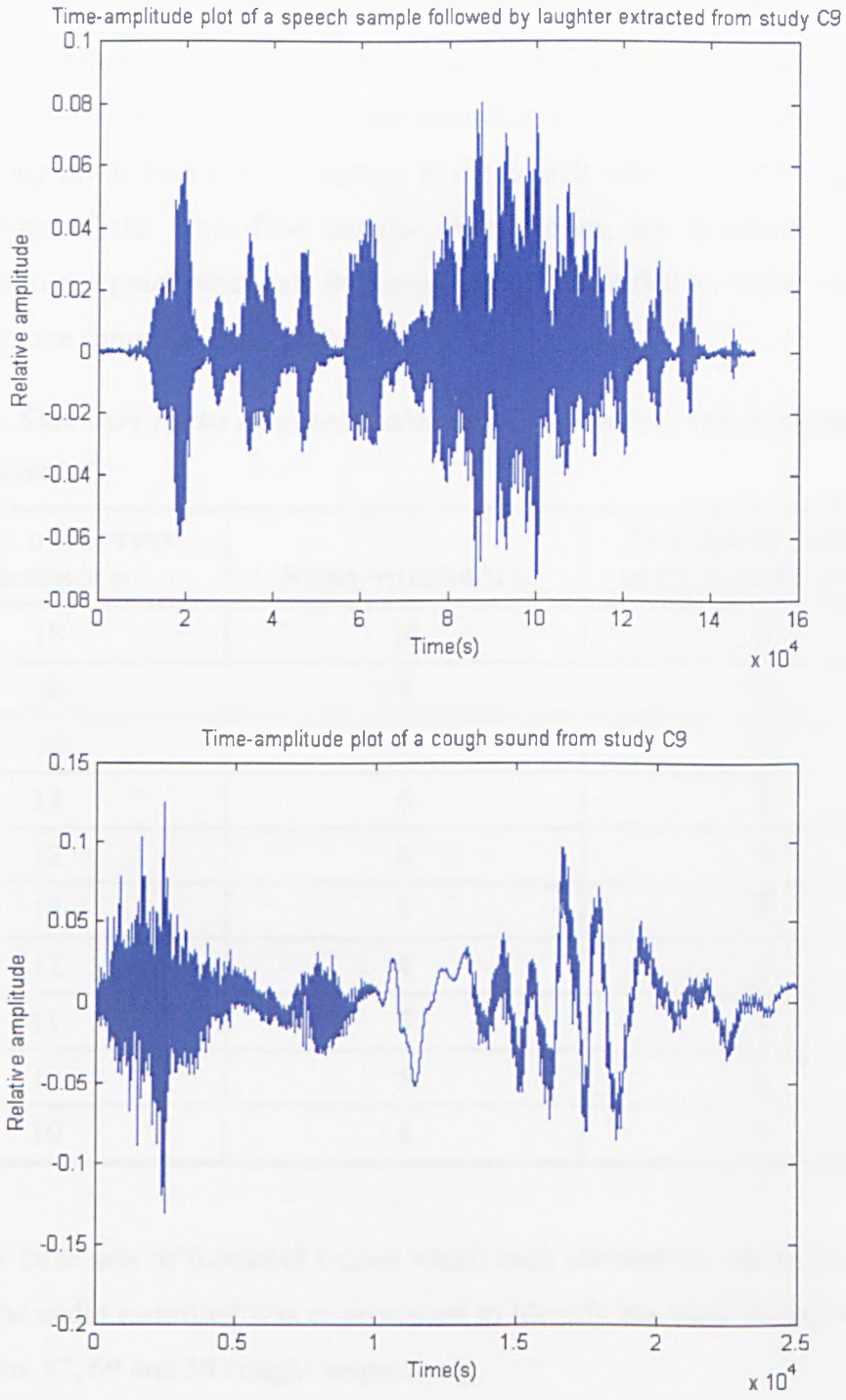


Figure 3.44 Time-amplitude plots of speech and laughter (top) and a typical cough (bottom) taken from the study C9

Detecting the coughs is the main objective, therefore the threshold levels need to be changed so that they are at the highest possible levels whilst not excluding any coughs. The threshold levels implemented are multiples of the baseline value; the baseline is calculated using a moving window and is set as the largest value out of the

minimum RMS for the window, or the default baseline value, set at 0.002 at this point. The threshold levels are therefore also theoretically moving values. However, analysis of a 30 minute portion of audio recording showed the minimum RMS value to be mainly lower than 0.002, resulting in the default value of 0.002 being used for the majority of the time. Five coughs were chosen, the threshold values were optimised to the point where all five coughs were identified as valid sound events. The results are summarised in Table 3-7.

Table 3-7 Summary of the threshold values specified and the resulting number of coughs detected

SIGNAL DETECTION THRESHOLD	NOISE THRESHOLD	NUMBER OF COUGHS DETECTED (OUT OF 5)
18	10	2
16	8	3
16	6	3
14	6	3
12	6	3
12	5	4
12	4	4
11	5	5
10	5	5
10	4	5

Using the three sets of threshold values which each allowed the detection of all five coughs; the audio recording was re-processed to identify the valid sound events. The results were 67, 69 and 50 coughs respectively.

This is still a large over-estimation compared to the actual results and by no means provides a solution. Defining the threshold limits so exactly to exclude certain sounds will also cause problems due to a natural variation of amplitude between subjects and audio recordings.

It was next decided to experiment with the number of epochs used to train the LVQ network [Network 3]. Increasing the number of epochs in one run of training from 25 to 100 gave the results as summarised in Table 3-8.

Table 3-8 Summary of LVQ network training to yield Network 3

File	Epochs	Cough Validation Accuracy (%)	Non-cough Validation Accuracy (%)	Training Error
A1	100	91	91	0.09
C1	100	100	75	0
A2	200	55	100	0.02
C5	100	97	65	0.13
A3	200	95	93	0.09
C4	100	95	59	0.12

Following the new training, the network gave an output of 19 coughs for the first hour of the audio recording. However, not all these were correct classifications. Some coughs were still missed, some double coughs were only picked up as singles and some laughter was still incorrectly classified as coughs. For the whole three-hour portion, 24 coughs were correctly counted as coughs, while 23 non-coughs were also counted as coughs.

The next step was to aim to reduce the threshold levels. Following experimentation, a multiple of 10 was used for the high signal threshold and a multiple of 2 for the noise threshold. Five sounds which were repeatedly being misclassified, including the subject's laughter, were isolated along with five coughs. These ten sounds were used to re-train the saved network; 60 epochs of training gave a result of 24 correctly classified coughs and 14 non-coughs incorrectly identified as coughs [Network 4].

It was decided at this stage to move onto the next cough study to ascertain if these problems were a feature of this study alone or of the new 24 hour cough studies in general.

3.3.5.4.2 Cough study C10

Study C10 was of a male subject so vocal sounds were expected to cause less interference than with the previous study. The first three hour portion of the audio recording was processed using the neural network trained on data as summarised in Table 3-8 [Network 3]. The threshold levels remained as for the processing of the previous cough study C9. However, these levels excluded a number of the lower amplitude coughs, therefore these levels were set to multiples of 6 and 1.5 of the baseline for the high signal threshold and the noise threshold respectively. This produced a count of 84 correctly classified coughs and five non-coughs incorrectly classified as coughs; four of these were loud speech sounds. A large number of coughs were incorrectly classified as non-coughs, and therefore excluded from the cough count. The reason for a large number of coughs being classed as non-coughs was perhaps due to the unusual pattern of the signal, a sample of which is illustrated in Figure 3.45 and Figure 3.46.

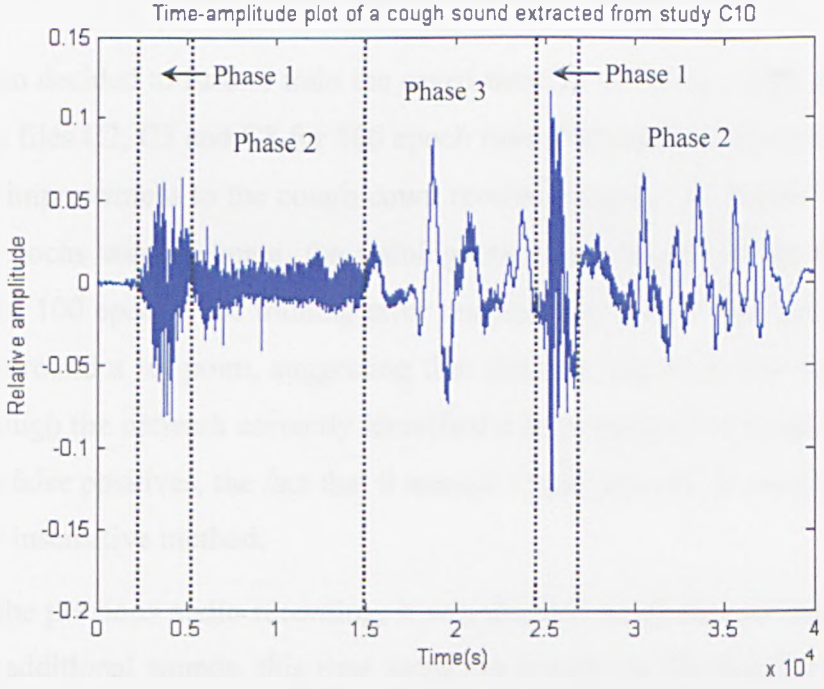


Figure 3.45 Time-amplitude plot of a typical cough sound of the subject in the C10 study; this shows a double cough with an unusual third phase to the first cough sound, and an unusual pattern to the second cough sound

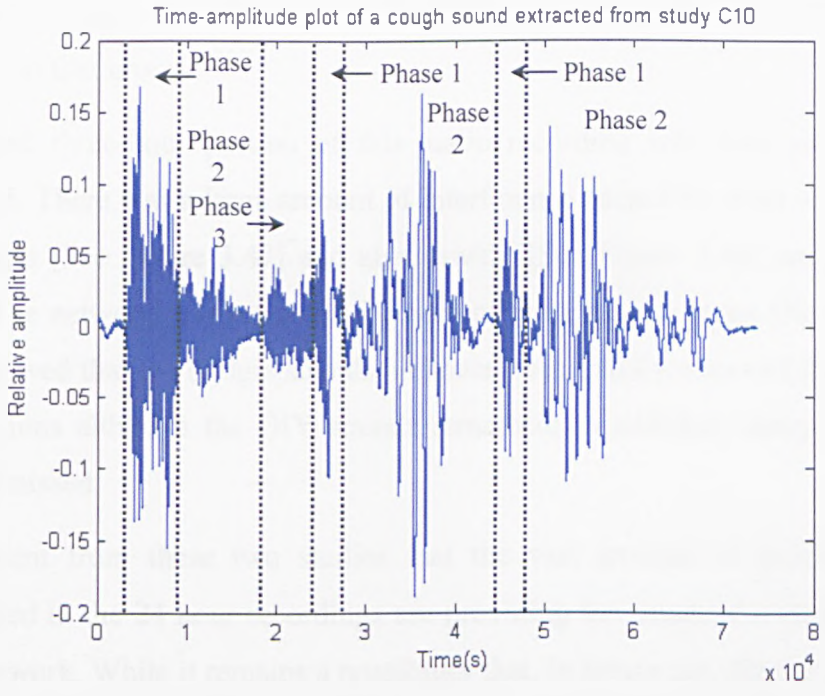


Figure 3.46 Time-amplitude plot of a typical cough sound of the subject in the C10 study; this shows a multiple cough with unusual patterns to the second and third cough sounds

It was then decided to further train the saved network by using cough and non-cough data from files C2, C3 and C8 for 100 epoch runs [Network 5]. However, this did not give any improvement to the cough count results obtained. To ensure the number of training epochs was adequate, the training error was monitored for 400 epochs of training, at 100 epochs, the training error reached a minimum and then proceeded to fluctuate around a set point, suggesting that this was the limit. The final result was that, although the network correctly identified a large number of coughs and suffered from few false positives, the fact that it missed a large amount of coughs rendered it a relatively insensitive method.

As with the previous audio recording, it was decided to attempt to train the network on some additional sounds, this time using the coughs as illustrated in Figures 3.45 and 3.46, that are failing to be correctly identified; this gave Network 6. The result was that many more non-coughs were identified as coughs. It was thought this result was due to the unusual signal pattern of some of the coughs, which introduced an element characteristic of non-coughs into the cough group; having this anomaly in the

the cough group resulted in many non-cough sounds being generalised as also belonging to that group.

The second three-hour portion of this audio recording was then processed using Network 5. There was a large amount of interference caused by what was assumed to be a budgie [See Figure 3.47] and also sawing [See Figure 3.48] and general DIY sounds. The network was retrained for both of these sounds types [Network 7]. The results showed that the budgie sounds had been successfully removed from the cough classifications although the DIY sounds remained; in addition, many more coughs had been missed.

It is evident from these two studies that the vast amount of interfering sounds encountered in the 24 hour recordings are providing too much of a challenge for the neural network. While it remains a possibility that, in future use, the network could be updated on occasional unrecognised coughs, the same cannot be done with incorrectly classified non-coughs; in practice, knowing that a number of classifications within a 24 hour recording are incorrect and require the network to be updated is not feasible.

The current approaches to the cough recognition stage are not ideal and need further consideration and improvement.

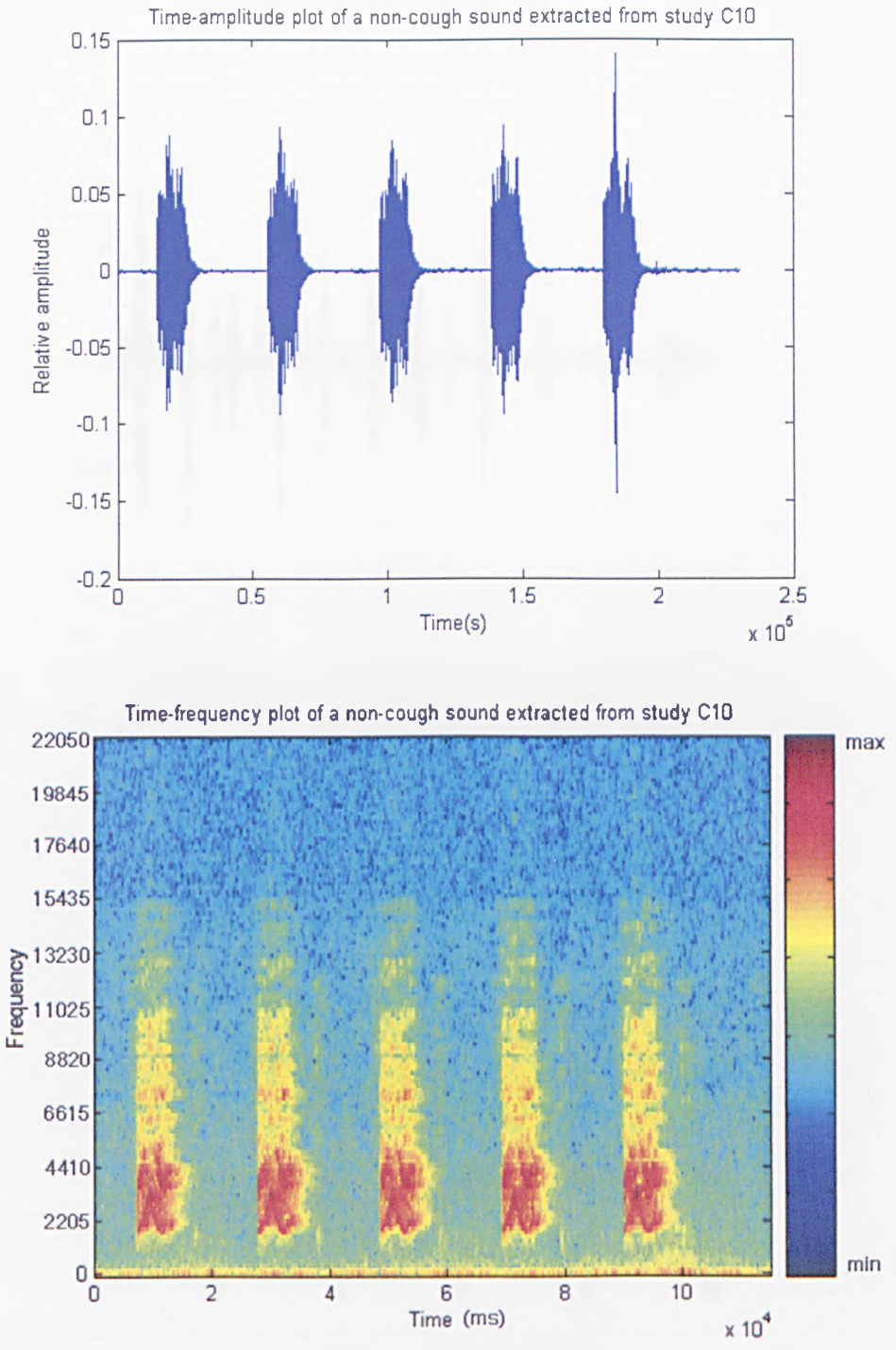


Figure 3.47 Time-amplitude plot (top) and the corresponding spectrogram (bottom) of the sound of a budgie

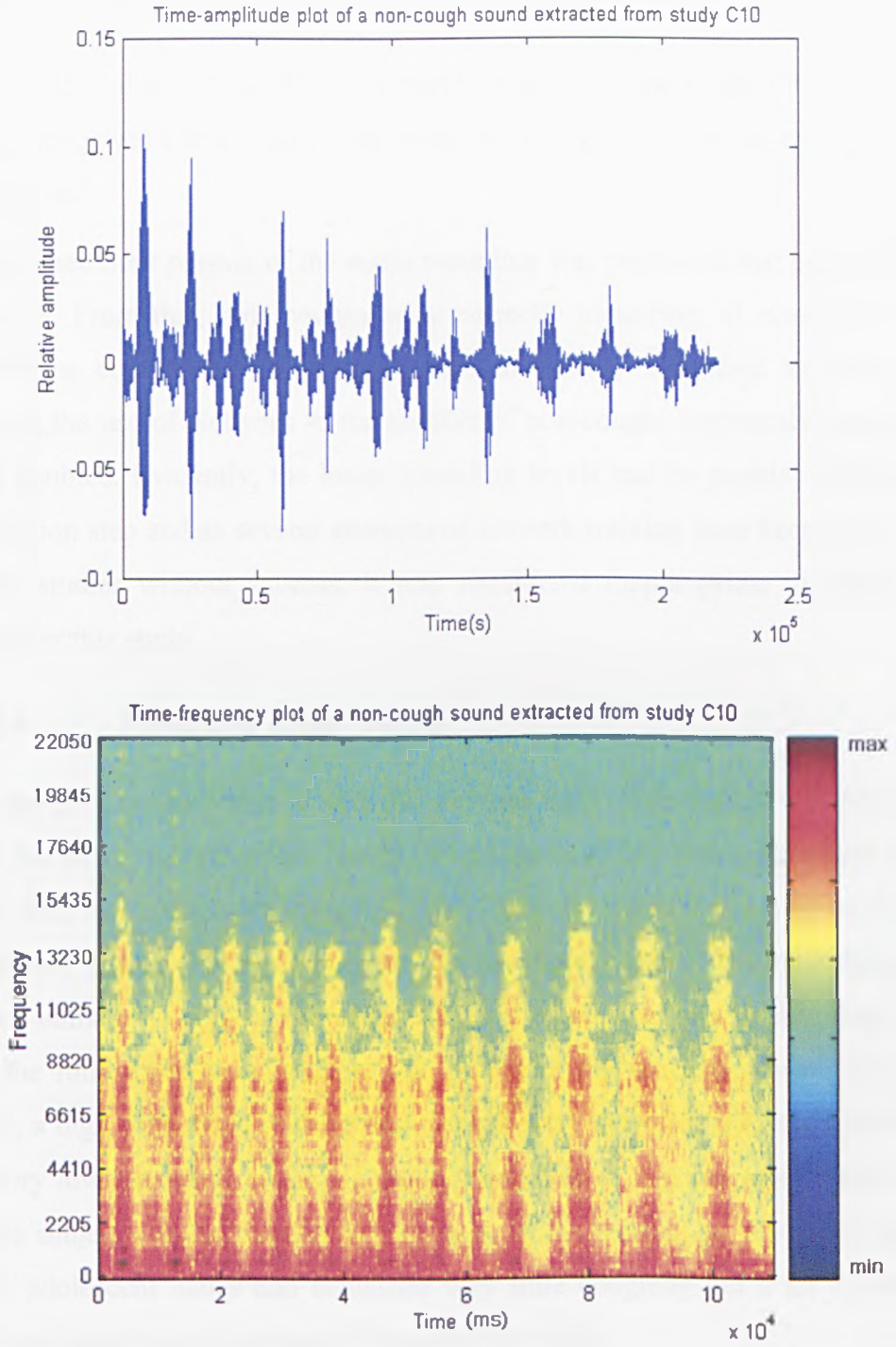


Figure 3.48 Time-amplitude plot (top) and the corresponding spectrogram (bottom) of the sound of sawing

3.3.5.4.3 Cough study C11

Following the large amount of background noise contained in the C9 cough study, this recording was carried out at the lower recording level, six, as changed on the recording unit.

The first three hour portion of the audio recording was processed and passed through Network 3. From this, nine coughs were correctly identified, 51 non-coughs were identified as coughs and 14 coughs were incorrectly identified as non-coughs. Following the use of Network 4, the number of non-coughs incorrectly identified as coughs doubled. Evidently, the lower recording levels had no positive effect on the classification step and as several attempts of network training have been made on the previous studies without success, it was considered inappropriate to repeat these attempts for this study.

3.3.5.4.4 Remaining cough studies

Due to the poor results obtained with the previous three recordings, it was decided not to alter the network settings for further optimisation of the remaining cough studies. Instead, they were processed using Network 3 with no modifications. Study C12 was of a female patient during a stay on the respiratory ward; the recordings were therefore unlikely to contain the wide variety of extraneous sounds that were picked up on the fully ambulatory studies. The results of the processing were reasonably positive, a high proportion of the cough sounds were detected, with the exception of some very low amplitude coughs, although there was a large amount of interference from the subject's voice. Further to this, three of the recordings, C14, C15 and C16 were of adolescent males and contained very little coughing but a lot of shouting, which was subsequently incorrectly classified as coughs.

It was decided at this point that the current method needed significant modification in order to perform adequately with the new recordings.

3.3.6 Additional data pre-treatment

Due to their nature, ANNs have the potential to work for the application of cough recognition. They are still considered to be the most appropriate pattern recognition

technique since the data requiring separation is so variable and requires a high degree of generalisation. Van Hirtum *et al.* have achieved pig cough recognition rates in excess of 90%[111, 114], although they had encountered problems discriminating between coughs and metal "clanging" sounds from machinery[113]. However, the range of sounds detected in the pigs' environment is, as expected, much more limited than the range encountered in 24 hour ambulatory recordings. Equally, the original use of the existing system involved a series of cough studies that were all carried out in the same clinical environment with the same potential background sounds and with a high number of coughs. Thus, the possibility of having already sufficiently trained the network with the majority of the sounds that were to be encountered in the validation recordings was very high such that the network needed only to generalise a small amount to correctly recognise sound events. Despite continuous efforts to successfully apply the use of an ANN to the 24 hour cough recognition, the variability both within and between the audio recordings have rendered this approach ineffective. Several tests were carried out to ensure the neural network had been set up correctly and was capable of classification; however, the test of the networks suitability to the actual data did not prove positive. It is therefore concluded that the problems of classification lie with the data itself; the two datasets in their current state are simply not sufficiently different.

Since the network is not effectively separating the data in its current form, it may be necessary to perform a different pre-treatment step in order to prepare the data into a more suitable format for separation.

The best way to maximise the potential for separation is to remove the information that is common to both groups and to leave only the data that contains the variance; thus enabling a better chance of separation. The approaches used so far have been successful for the shorter duration recordings; though have failed to overcome the problems raised by the vast amount of variation introduced by 24 ambulatory patient studies. Such variation is present both between the non-cough events and also within the cough events. The common factor and the variation therefore both need to be identified.

Focusing initially on the information that can be removed from both data groups; referring back to Section 3.3.3.1.2, analysis of the frequency content of cough and non-cough events, the common factor to all sound events is the low frequencies. Therefore, applying a frequency filter to remove the low frequency information common to both groups, could lead to a better degree of separation.

3.3.6.1 Frequency filtering

The aim of this experiment was to exclude any insignificant low frequency information common to many sounds and adding no valuable information to the data. Frequency filtering has been used in many applications where unwanted frequency information is present, however it has not often been applied to the recognition and counting of cough. The lack of use is definitely not due to it being a novel technology, as in 1967, Loudon demonstrated the use of a frequency specific attenuator and amplitude discriminator in order to reduce the number of sound events actually recorded onto tape; thus reducing the time required to manually count the coughs[31]. Later in 1988, Salmi et al. used high pass filtering to eliminate low frequency noise[35].

3.3.6.1.1 Application of a Butterworth filter

The first approach was the application of a Butterworth filter; a narrow band filter which was set to be high pass to remove low frequency information whilst maintaining a specified band of high frequency information.

The initial working of the frequency filter was achieved by processing each minute of an audio file, one at a time to avoid the memory constraints imposed by working with such large WAV files. The signal remaining after frequency filtering represents the intensities of the specified frequencies.

Listening to the recording showed it to contain 39 loud coughs and 14 low amplitude coughs. The results of applying the Butterworth filter to the cough study C1 showed positive results. The initial run using a frequency band of between 14700 Hz and 15300 Hz (15 kHz filter) gave a result of 49 sound events. Inspection of the 49 sound events following the filtering stage revealed them to contain 31 coughs, including two

that were not picked up by the original sound event detection, six vocal sounds, one inhalation sound and nine other sounds.

The threshold level was changed to $1E^{-4}$ to test if any of the missed coughs would be detected. However, although 97 sound events were identified, most of the coughs missed originally remained undetected, whilst a significant number of non-coughs were introduced into the results.

The next experiment was to change the frequency filter from 15 kHz to 12 kHz to allow more frequencies through and potentially detect more of the cough events. This only picked up 32 sound events, 28 of which were coughs and four were non-coughs. Lowering the filter frequency further to 11 kHz yielded 33 sound events.

Using a high pass filter without selecting a narrow band allowed all sound events containing frequencies above the filter level. A 12 kHz filter with a threshold level of $2E^{-4}$ subsequently gave a result of 240 sound events detected. The reason for this vast increase in detected events is due to the wider range of frequencies being used. This will give rise to a larger signal, resulting in many more sound events possessing the required signal intensity to exceed the threshold level. However, it would be difficult to simply increase the threshold and exclude the unwanted sounds, as it would not be easy to distinguish which events possessed frequencies attributed to cough sounds and which simply contained a relatively uniform amount of each frequency throughout the range.

Reverting back to the narrow-band filters and experimenting with the threshold levels, three filters of 12, 14 and 15 kHz were created as summarised in Table 3-9.

Table 3-9 Summary of filter frequencies, threshold values and the results obtained

FILTER FREQUENCY	THRESHOLD VALUE	SOUND EVENTS DETECTED	COUGHS CLASSIFIED CORRECTLY	NON-COUGHS INCLUDED
12	$7E^{-3}$	64	39	23
14	$7E^{-3}$	51	35	16
15	$7E^{-3}$	48	35	13

The method that detected the most number of coughs was the 12 kHz frequency filter with a $7E^{-3}$ threshold. Only one of the loud coughs was missed, however there were 23 extra non-coughs detected; these included five vocal sounds, three metal sounds, three clicks, one rustle, six other sounds and five that were actually parts of other coughs.

The frequency information had also been used to identify the start and end times for the cough events. However, spectrograms of coughs shown in Section 3.3.3.1.2.1 showed that many coughs only have high frequencies for the first phase of the cough and have lower frequencies for the second and third phase, if present. This would result in only half of the event being classified in as a cough. It was therefore decided to use the amplitude information. This was partly due to the fact that the method used to confirm the occurrence of cough in the recordings is based on sound and audibility. Also the use of an amplitude threshold allowed more control over the definition of the cough event as opposed to frequency information which may not always be higher than the frequency filter passband for some coughs or some parts of coughs.

3.3.6.1.2 Application of additional constraints

Using the 12 kHz filter and the $7E^{-3}$ threshold level, the C1 file was processed and the results analysed.

A common cause of interference was the occurrence of sector boundary errors (SBEs) in the recording. SBEs are portions of signal which have been cut out of alignment with the sector boundary; to combat the resulting gap, the signal processing software fills it with zeros. When the signal is then played back, the sudden change in frequencies from the normal signal to zeros causes a short, high-pitched sound known as a SBE. These were investigated and found to have very short durations compared to a normal cough sound and were probably only due to the signal chopping being carried out to test the system. A minimum duration was implemented to omit these sounds, the minimum being below 2000 samples or 0.045 sec. This also served to cut out some of the speech fricative sounds which were also high frequency but very short.

For cough sounds in excess of one second duration, the *plotcough* function was applied to determine the actual number of coughs present in the event [See Section 3.3.7.2 for details].

There was also a number of high frequency, low amplitude sounds which would probably be excluded by the introduction of an amplitude threshold as discussed previously.

3.3.6.1.3 Application of FFT

It was decided that the previously applied step of identifying valid sound events prior to further analysis was an important step and still required in this case to avoid unnecessary processing of "soundless" regions and also to avoid interference by high frequency but very low-amplitude sounds.

Once this step had been implemented, each sound event was processed in sequence using FFT; frequencies up to 8 kHz and above 15.5 kHz were then discarded. This step replaces the use of the Butterworth filter. A diagram of the frequencies isolated for analysis is shown in Figure 3.49. Illustrations of the intensities of the isolated frequencies for each window of time for examples of cough and non-cough events are shown in Figures 3.50 and 3.51 respectively.

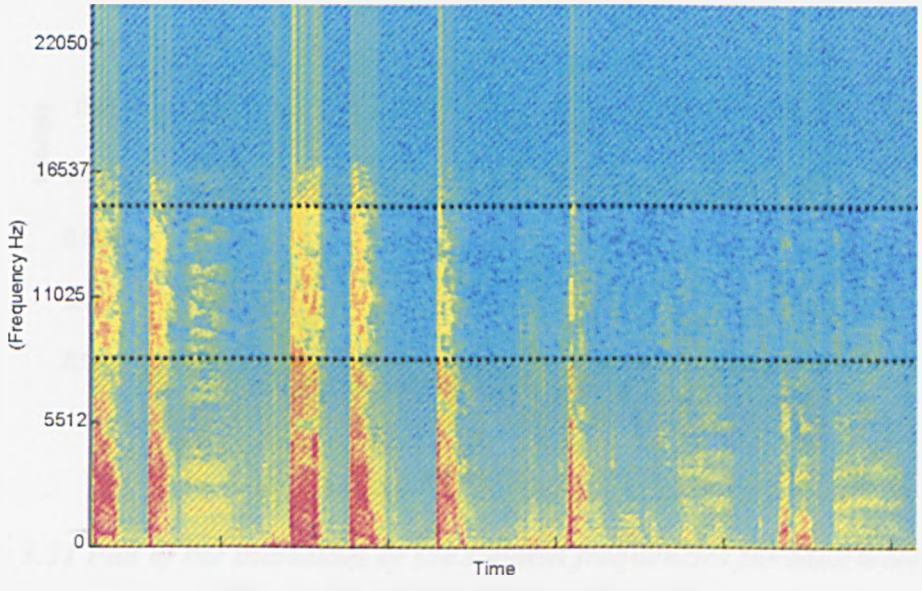


Figure 3.49 A frequency spectrogram illustrating the frequency band isolated for analysis and the remaining frequencies to be discarded (shaded)

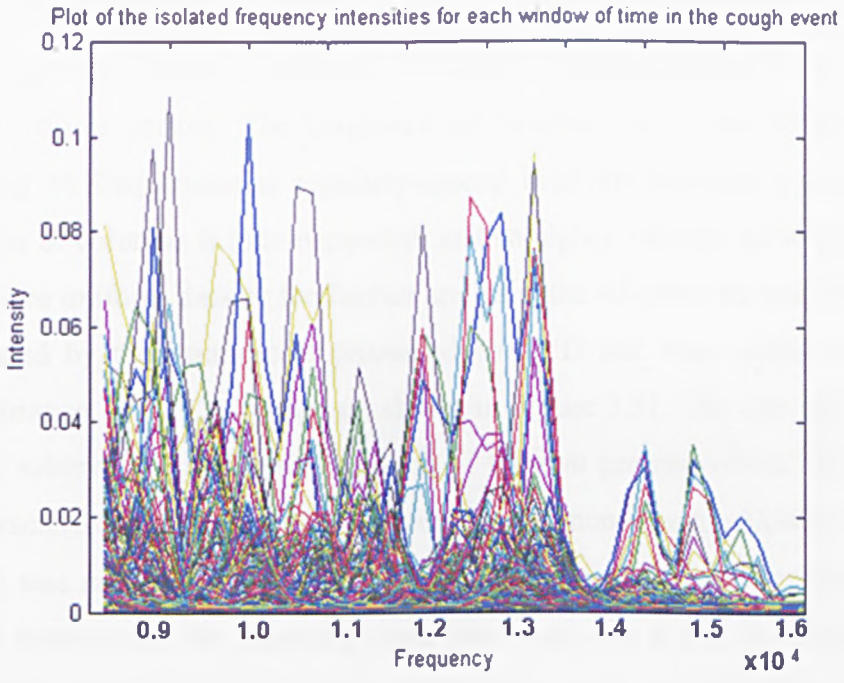


Figure 3.50 Plot of the intensities of the isolated frequencies for each window of time in the sample cough event

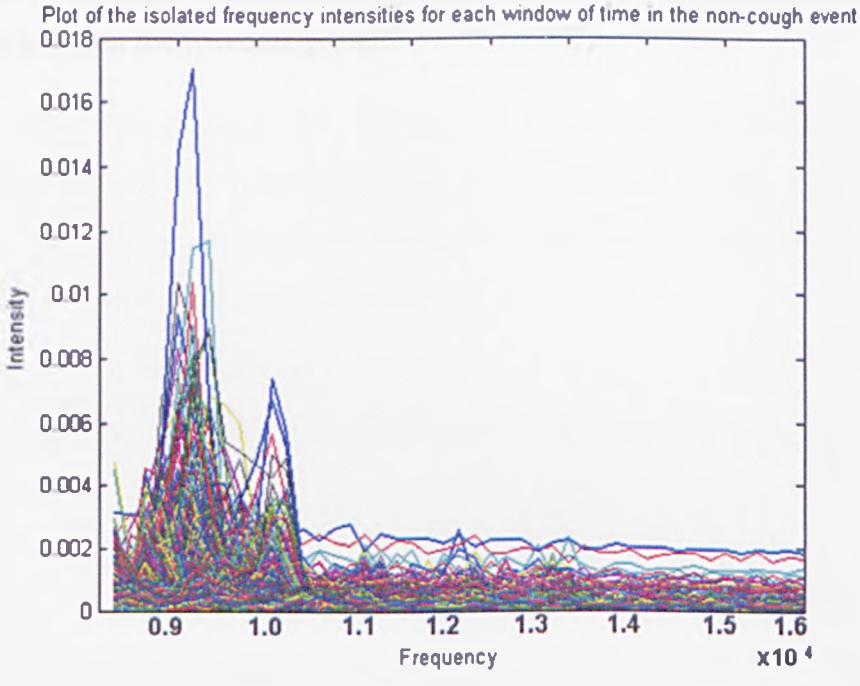


Figure 3.51 Plot of the intensities of the isolated frequencies for each window of time in the sample non-cough event

Playing the sound of the remaining frequencies of a typical cough gives a sound like a very short crackle; there is therefore very little frequency content audible to the human ear in this portion. The frequency information was in the form of 45 rows representing 45 frequencies at regularly-spaced intervals between 8 and 15.5 kHz. The number of columns is time-dependent and so highly variable between events. To create a more uniform dataset for further analysis, the 45 rows for each sound event were squared by their transpose, processed by SVD and then scaled according to relative variance. The PCA scores are shown in Figure 3.52. The aim of PCA was to attempt to achieve a significant separation of the two groups, which would indicate some characteristic difference between coughs and non-coughs. However, although separation was not achieved, the PCA scores did reveal an important characteristic difference between the two data sets; while the cough data points are reasonably well dispersed, the non-cough data points are superimposed on each other, even those from different non-cough events. This suggests that the cough events have a much greater degree of variation in the isolated frequency band than the non-cough events. This feature highlights the potential for classification as this could be the required difference between the two data groups.

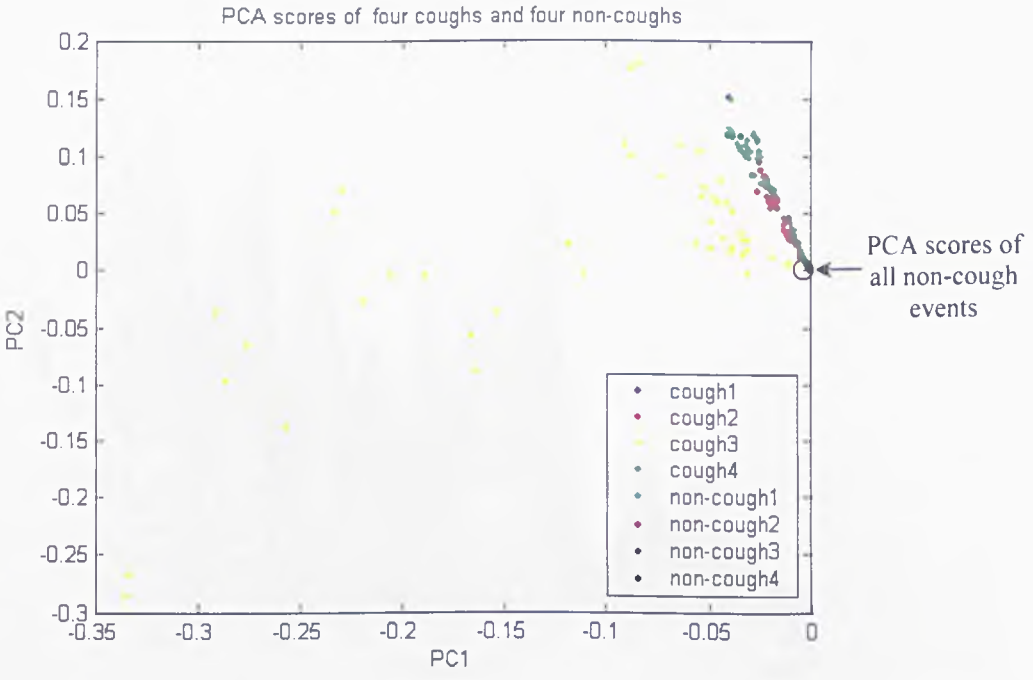


Figure 3.52 PCA scores of the squared FFT data from the isolated 8 – 15.5 kHz frequency band of four coughs and four non-coughs, illustrating the relatively small spread of the non-cough scores

3.3.6.1.4 Covariance

As the results of the previous experiment have showed the cough data to possess a much higher amount of variation than the non-cough data, it was decided to apply covariance analysis to the data and attempt to ascertain whether or not there is a significant difference between the two groups.

The results of the covariance calculations for the cough and non-cough events are shown in Figures 3.53 and 3.54 respectively. For these cases, the cough has a maximum value for covariance of approximately $1.6E^{-4}$, whereas the non-cough has a maximum of approximately $2.2E^{-7}$.

Applying the FFT filtering step and the covariance calculation to all the sound events in the file showed that non-cough events have a maximum covariance of $1E^{-6}$ while cough events have a maximum of $1E^{-3}$.

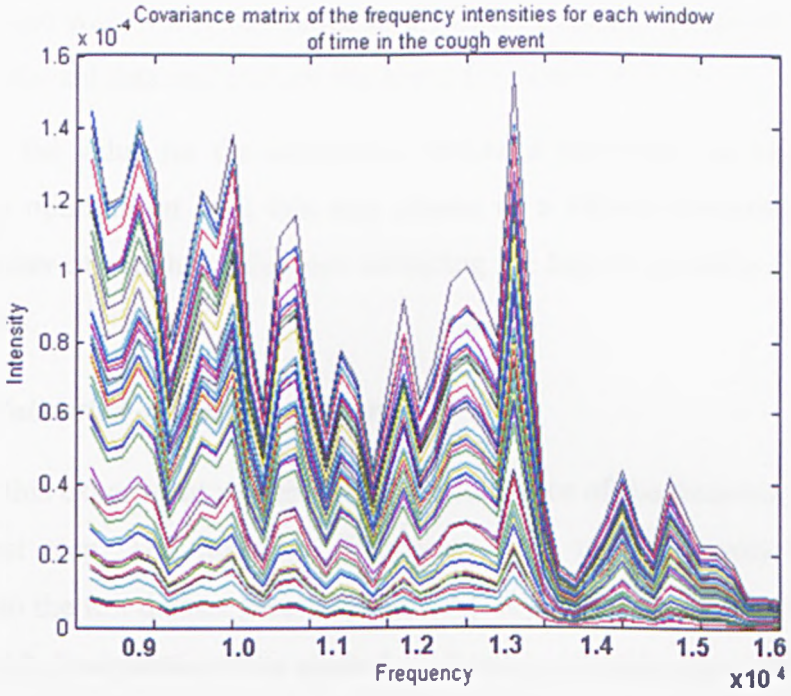


Figure 3.53 Covariance of the 8 – 15.5 kHz FFT data for one cough event

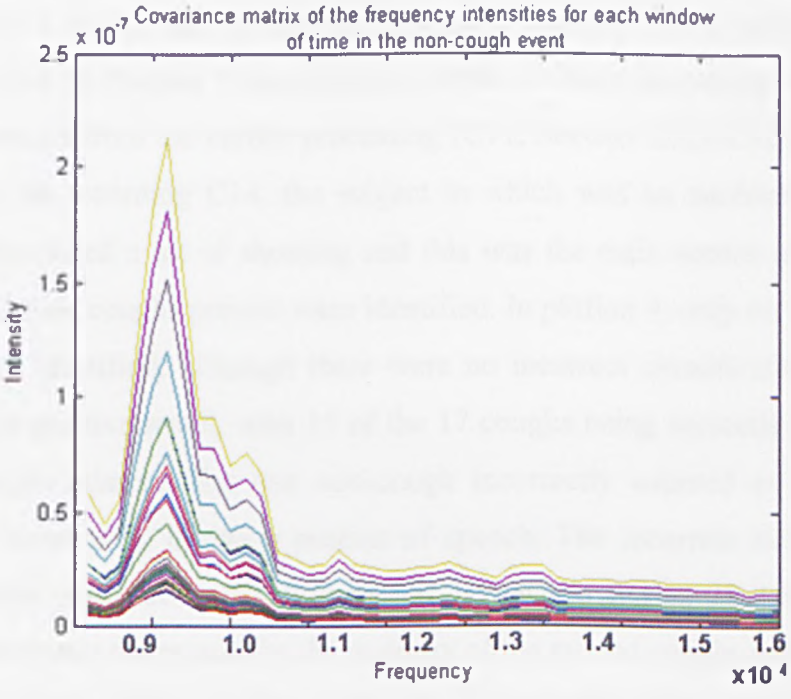


Figure 3.54 Covariance of the 8 – 15.5 kHz FFT data for one non-cough event

It was decided to use this difference and introduce a covariance threshold to the processing stage. Following the application of the frequency filter to known data, the

frequency band was further optimised to 6-15 kHz in order to capture the maximum amount of relevant data and exclude the lower frequency noise.

In addition, the value for the covariance threshold was tested at varying levels at found to be optimum at $1e^{-4}$; this was chosen as a balance between detecting the highest number of coughs whilst also excluding the highest possible number of non-cough sounds.

3.3.6.1.5 Validation of frequency filtering

The aim of this experiment was to test the performance of the frequency filtering data pre-treatment step. The function '*ffi_process*' was used to apply the described techniques to the test dataset [See Appendix A]. A summary of the results are shown in Table 3-10. Each section was studied in detail to identify any weaknesses in the system. For portions 1 and 2, all coughs were counted, with no extra sounds misclassified. Portion 1 was in fact a selection of coughs and non-coughs from the audio file C10 which had proven so difficult to classify in the earlier stage [See Section 3.3.5.4.2]. Portion 2 was also one of the 24 hour recordings which had not had good results from the earlier processing (C11, Section 3.3.5.4.3). Portion 3 was taken from the recording C14, the subject in which was an adolescent male. The recording contained a lot of shouting and this was the main source of interference, although all three coughs present were identified. In portion 4, only six of the thirteen coughs were identified, although there were no incorrect classifications. Portion 5 gave another positive result, with 15 of the 17 coughs being correctly classified and only 2 coughs missed. The one non-cough incorrectly counted as a cough was actually a fricative 's' during a portion of speech. The incorrect classifications in portion 6 were also due to fricative speech, however a further 29 coughs were not detected. It appears the reason for the majority of the missed coughs were due to them being very low amplitude sounds. Portion 7 was taken from another adolescent subject, the majority of the incorrectly labelled 'coughs' were due to shouting, with one being due to loud music. In Portion 8, the two incorrectly labelled 'coughs' were in fact sounds from the subject blowing their nose. Of the 11 missed coughs, 6 were actually due to the fact that the amplitudes were not large enough to be classified as valid sound events.

Table 3-10 Summary of results obtained from the application of the filter to the "testset" file

Portion	Actual Coughs	Coughs Counted	Correct		Incorrect		Missed	
			No.	(%)	No.	(%)	No.	(%)
1	40	40	40	100	0	0	0	0
2	14	14	14	100	0	0	0	0
3	3	16	3	100	13	81	0	0
4	13	6	6	46	0	0	7	54
5	17	16	15	88	1	6	2	12
6	118	94	89	75	5	5	29	25)
7	3	8	1	33	7	88	2	67
8	75	66	64	85	2	3	11	15
TOTAL	**283**	**260**	**232**	**82**	**28**	**11**	**51**	**18**

In summary, 82% of the cough sounds were classed correctly while the remaining 18% were missed and of the coughs counted, 89% were correct. With the exception of sections 4 and 7 which contained few coughs and subsequently gave a low percentage of correct classifications, all other recordings gave highly satisfactory results. The majority of the missed coughs were due to them being of too low amplitude to be detected. The reason for the discrepancy between the actual number and the detected number is actually due to an overestimation in the former. The actual number of coughs was counted simply by counting every single audible cough; in reality, however, many of the smaller coughs would not need to be classified as coughs as they were closer to the category of throat clearings rather than coughs. As there is no set definition of cough and also as it is simply a matter of judgement, it is very difficult to perform an accurate cough count without ambiguity. To support this, portion 6 of the test cough file was listened to once more, this time with an audibility threshold in mind so as to ignore very small "throat clearing" coughs. The result was 89 coughs, exactly the result following filtering. The same was done for portion 8 which revealed a new count of 69 coughs.

The frequency filtering work was intended as an additional pre-treatment stage prior to the use of a neural network. Thus, the most important figure is the amount of coughs that are correctly identified by the method and the amount that have failed to be classified. Any issues with sensitivity introduced at the pre-processing stage will carry through to the classification stage and no matter how strong the latter is, it will not be able to make up for the deficiency. Also important is the ability of the pre-treatment stage to reduce the number of interfering sounds in order to reduce the demands on the neural network. Following the sound identification stage, 1270 sound events were identified, of which 260 were coughs (20%). This pre-treatment step has therefore increased the cough composition to 82%, referring back to the use of contingency tables in Section 3.3.3.2.4, at a system efficiency of 80%, the cough accuracy is approximated at 94.7%. Thus, by increasing the ratio of coughs to non-coughs, the cough detection accuracy has instantly been improved, even without improving the accuracy of the classification system. Results for sensitivity are calculated to be 82%, specificity is 97% and accuracy is 94%. Personal communication with Prof. Alyn Morice (Chair of Respiratory Medicine, Castle Hill Hospital, Cottingham) reveal these figures are considered highly acceptable for clinical applications and are indeed above some of the quoted results for other cough monitors.

3.3.6.1.6 Application of neural network

The next stage was to apply the resultant data from the frequency filtering pre-processing stage to a neural network in an attempt to remove some of the wrongly classified non-cough sounds. All the sound events present in the testset file were identified and the MFCCs were calculated. The data was then divided equally into two datasets for training and validation; resulting in 98 coughs and 14 non-coughs for each group. The results obtained from the application of the neural network correctly recognised all but one of the coughs, although incorrectly classified every non-cough as a cough.

A theory for this result is that MFCCs are no longer an appropriate choice for this data, as these represent the intensities of each frequency present in the signal. Any non-coughs that have remained following the FFT filtering stage show an immediate

similarity with coughs; furthermore, the covariance within the frequency band selected is relatively high. Thus, it would suggest that the cepstral coefficients of these extraneous non-cough sounds would not be hugely different from those of coughs. Since the neural network did not perform particularly well in previous tests, applying it to data which is suspected to be very similar is perhaps not the best approach. Instead it would perhaps be more effective to use characteristics of the data other then those in the frequency domain, for example, RMS data or spectral envelope patterns.

3.3.6.2 Application of DSP pre-processing to 24 hour studies

For further validation, the frequency filtering method was applied to the cough studies that proved difficult to analyse in Section 3.3.5.4. Results showed a significant improvement in the number of coughs detected, although there remained a number of non-coughs incorrectly classified as coughs. For the cough study C10, 84 coughs were correctly classified in the previous method and this increased to 105 using the DSP method. The majority of the coughs missed, which were only a few, were missed at the amplitude checking stage and would therefore benefit from changing the threshold levels. There was a high number of interfering sounds, 71, most of which were vocal sounds and loud banging sounds. The time-amplitude and spectrogram plots are shown in Figures 3.55 and 3.56 for a loud bang and a typical speech sound respectively. Inspection of these plots clearly shows the reason for the wrong classifications; the frequency spectrograms, upon which this recognition method is based, are very similar to those of coughs. However, inspection of the time-amplitude plots shows the potential for separation by shape of the sound event, perhaps by use of RMS plots. Coughs generally have an initial high amplitude portion followed by a decline, whereas speech is characteristically different. Additionally, although loud bangs look very similar to cough sounds, they have a much smoother, almost exponential in appearance, decline from high amplitude to low. Again, this has the potential to be recognised by RMS patterns.

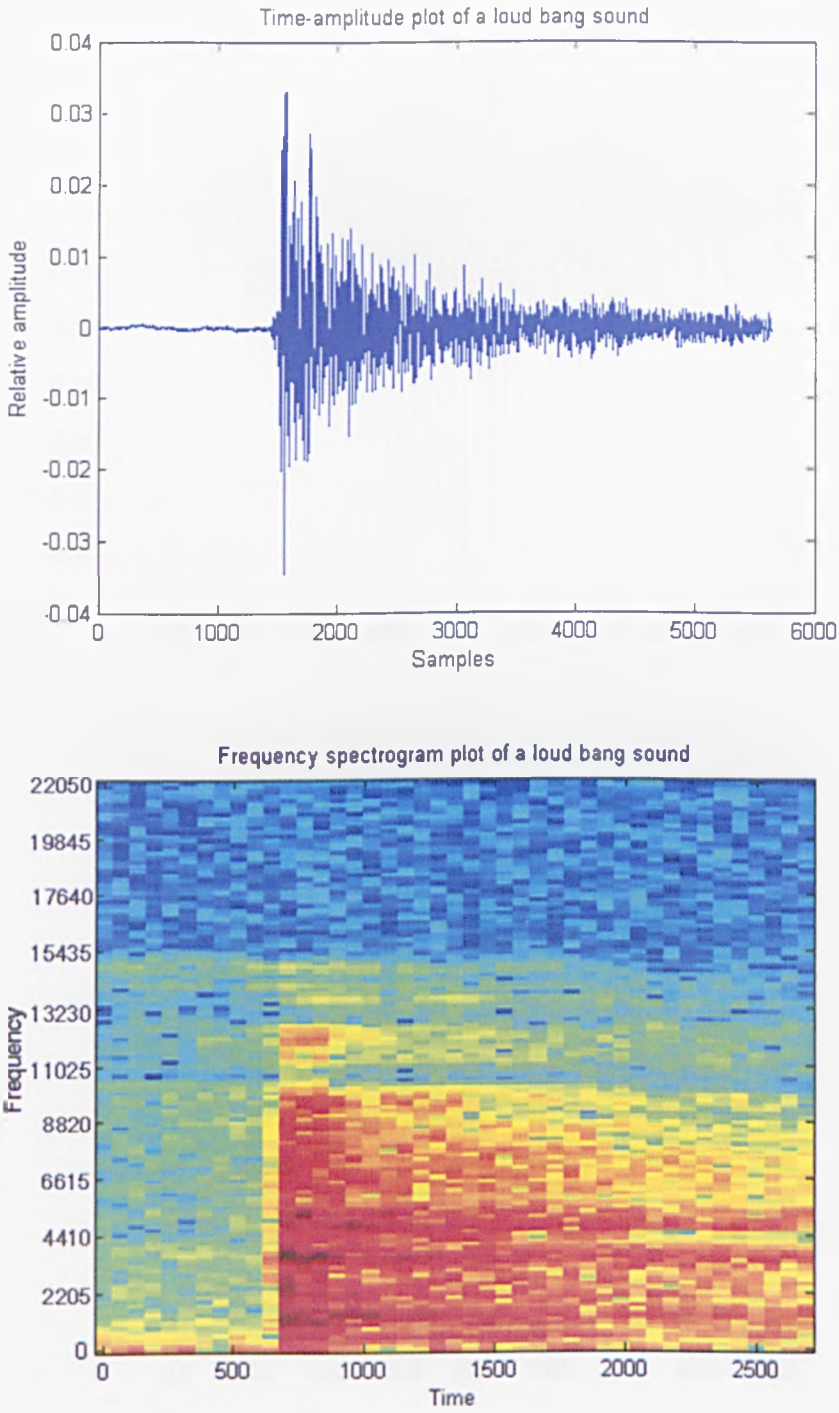


Figure 3.55 Time-amplitude plot (top) and spectrogram (bottom) of a loud bang wrongly classified as a cough due to the similarity in frequency information

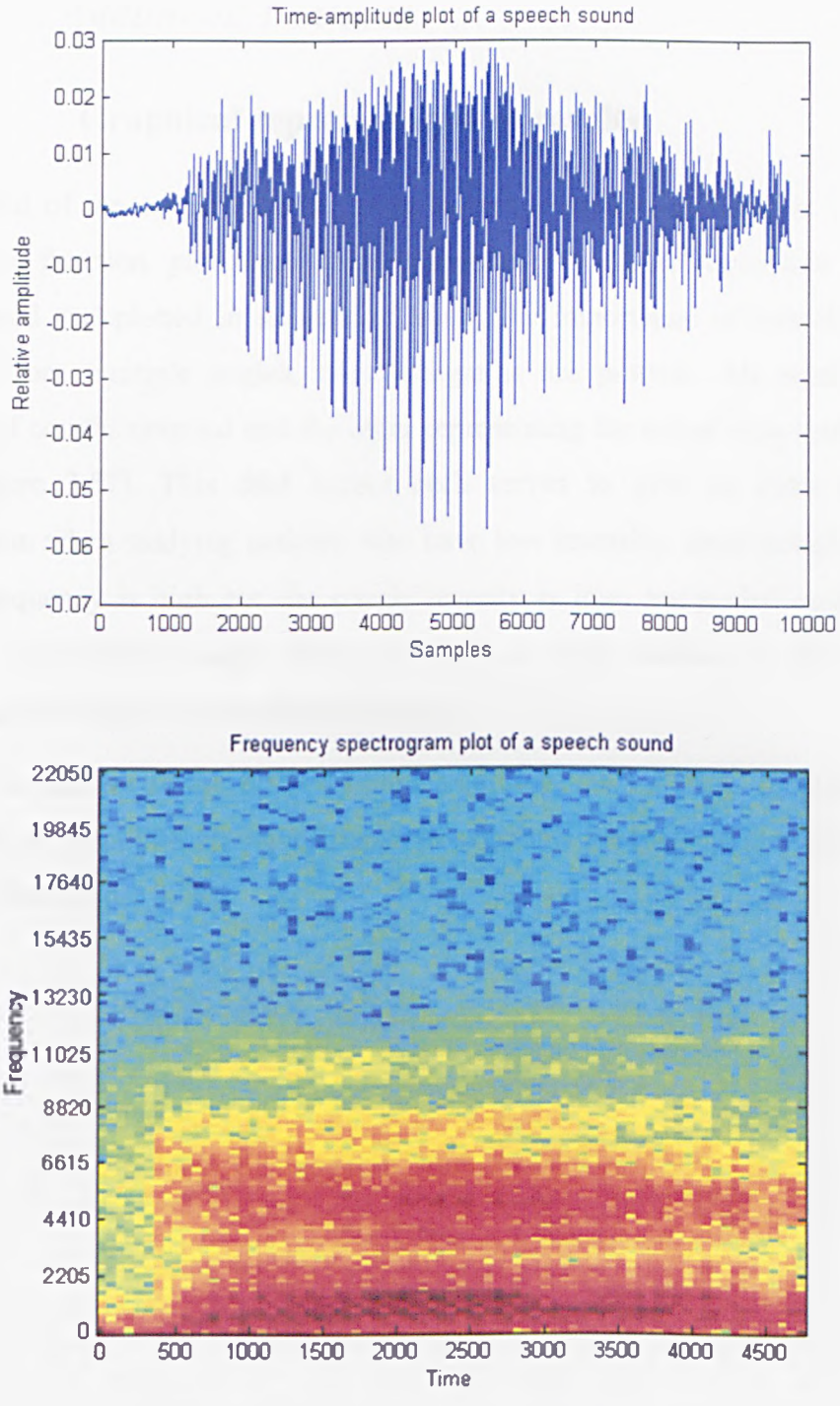


Figure 3.56 Time-amplitude plot (top) and spectrogram (bottom) of a typical speech sound wrongly classified as a cough due to the similarity in frequency information

3.3.7 Additional Features

3.3.7.1 Graphical representation of results

The output of the cough counting stage is represented graphically as a histogram, using the function *plotcough* [See Appendix A]. Each three-hour portion is summarised and plotted in sequence. Due to the importance of considering cough severity from multiple angles, two histograms are plotted; one representing the number of coughs counted and the other representing the actual time spent coughing [See Figure 3.57]. This dual output both serves to give an extra measure of comparison when studying patients who have low intensity, small coughs where the cough frequency is high but the cough severity is low; but it also counteracts the problem of multiple coughs being counted as single events, as the time spent coughing would give a more accurate result.

In order to summarise the entire 24 hour recording, the function *'combineplot'* [See Appendix A] combines all eight three-hour segments for each 24 hour recording into one plot [See Figure 3.58].

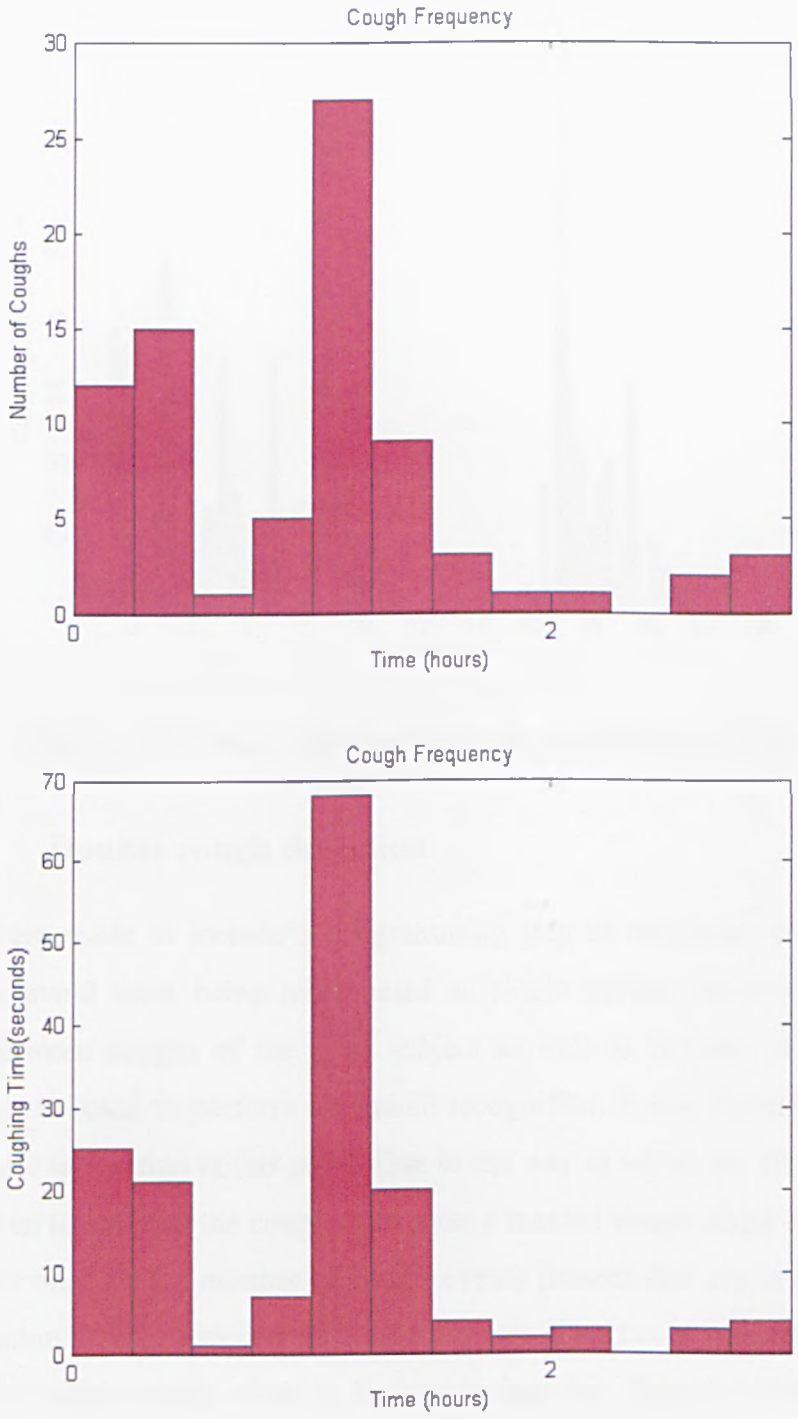


Figure 3.57 The two graphical outputs of the function 'plotcough' representing both the number of coughs in a three-hour segment (above) and the time spent coughing during the same three-hour period (below)

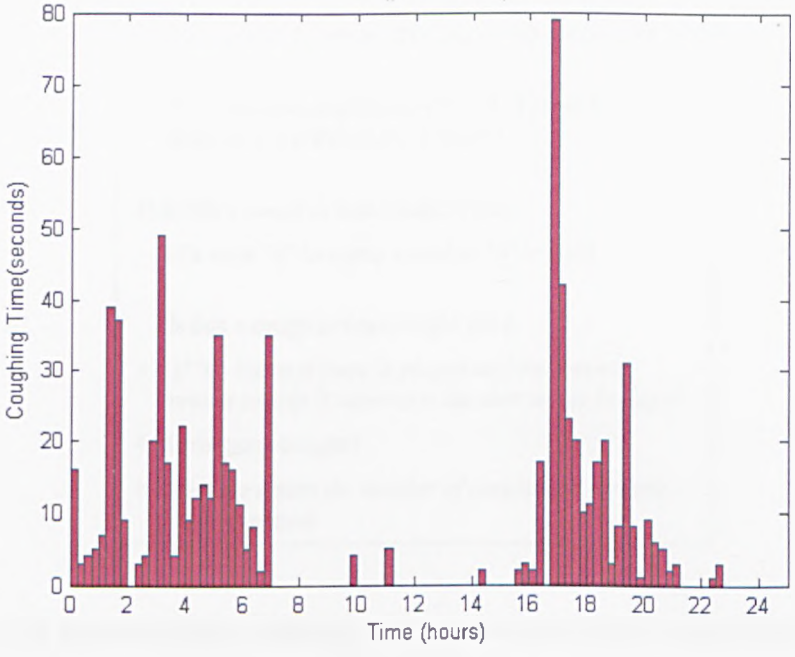


Figure 3.58 Total cough summary for a 24-hour recording

3.3.7.2 Double cough detection

Attempts were made to include a programming step to recognise double or triple coughs and avoid them being miscounted as single events. However, due to the variation between coughs of the same subject as well as between subjects, no set pattern could be used to perform automatic recognition. It was therefore decided to introduce user interaction at this point. Due to the way in which the file is processed, the time taken to listen to the coughs and enter a manual cough count is minimal and is dependent only on the number of cough events present that are in excess of one second duration. The one second threshold was introduced as it was decided that it is unlikely for a single cough event to be longer than this. Figure 3.59 represents the process that occurs during the user-interaction stage.

1) X cough events longer than one second

2) Do you wish to review the long cough events? (y/n)

>> *if 'y' the time-amplitude plot of first event is displayed and the sound is played*

3) Is this a cough or non-cough? (y/n)

(Or enter "p" to replay sound or "q" to quit)

Is this a cough or non-cough? (c/n)

>> *if 'n', the next event is played and the process reverts to step 3, otherwise the user moves to step 4*

4) 'How many coughs?

>> *the user enters the number of coughs and the next event is played*

Figure 3.59 Screen display during user-inspection of the long 'cough' sounds

Following this process, the function recalculates the cough count and plots an updated cough frequency summary. This process was not only found to be useful for correcting single cough events, but also for identifying portions of signal which are actually non-coughs but have fulfilled all the criteria to be classified as coughs. This usually occurs in sound events that remain above the amplitude threshold for a reasonably long duration (in excess of one second) and contain high frequencies. They then pass the frequency threshold, the minimum duration check and also the covariance check simply because they have more variation due to their long duration. Non-coughs that are shorter than one second have usually been shown to possess lower than threshold covariance values and are excluded earlier.

3.3.8 Patient Activity Monitoring

The patient activity marker requires an efficient and accurate processing method to locate the occurrence of the marker and the corresponding activity. As with the cough recognition software, it would be ideal to have an automatic process by which the software can identify the times that the activity marker was used and then compile the necessary information. The activity marker is inserted onto the second channel of the audio recording. The purpose of this section was to design a method for automatic

analysis of the audio recording in order to locate the use of the marker and determine the activity the patient described.

3.3.8.1.1 Spectral analysis

In order to search for the marker automatically, a characteristic signal needs to be used. Spectral analysis of the marker signal was performed to find a feature that would identify the occurrence of the marker with maximum sensitivity and minimum number of false positives. A simple method of signal identification in an audio recording is by use of digital filters to isolate all frequencies with the exception of a narrow band. The narrow frequency band needs to be selected according to the frequency of the identifying marker. The occurrence of a signal within this narrow band can then easily be located. Selection of a frequency that is unlikely to occur, other than when the event marker is used, reduces the problem of false positives. The frequency spectrogram of the marker is shown in Figure 3.60, while a plot of the intensities of each frequency is shown in Figure 3.61. The values for these frequencies are 1206, 3445, 5684, 7580, 10136, 12403 and 14642 Hz. The four higher frequencies clearly contain the least interference. The second channel has the option of being used as a microphone to record background sound such that noise cancellation can be carried out if necessary; it was therefore decided to keep this option open by using a frequency above that of normal speech in order to identify the marker. Normal speech usually stays below 6000 Hz, however, to exclude maximum interference, it was decided that the frequency 14600 Hz was the most appropriate for this filtering method.

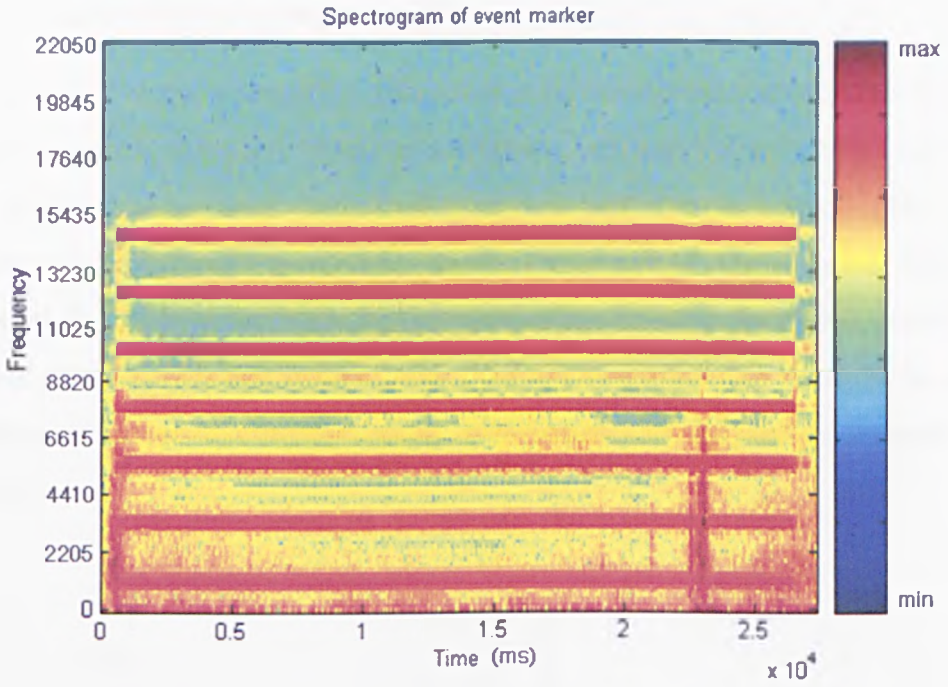


Figure 3.60 Spectrogram of the event marker showing the frequency content of the sound. The dark orange areas represent the most intense frequencies.

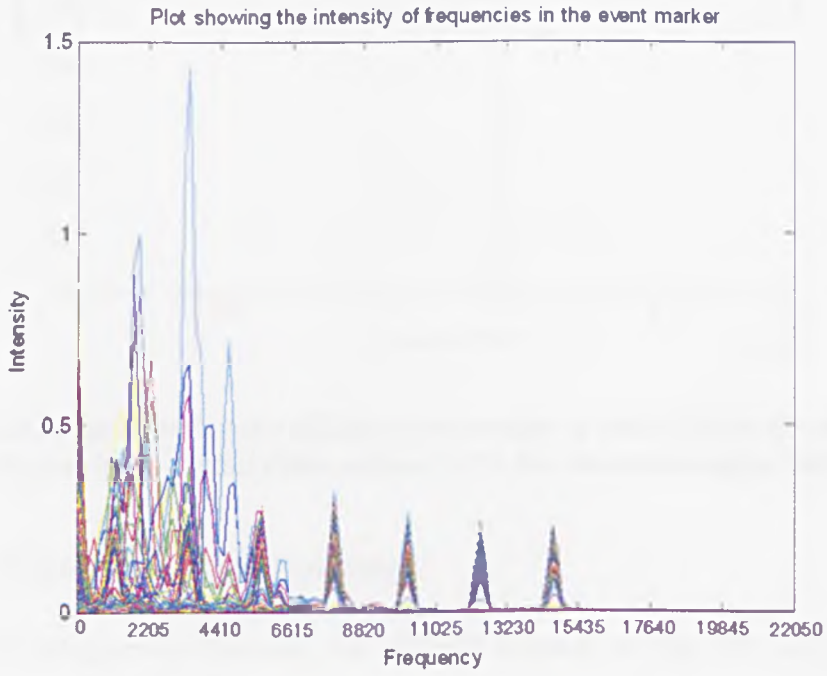


Figure 3.61 Plot showing the intensity of each frequency in the event marker

3.3.8.1.2 Application of a frequency filter

Having decided on the appropriate frequency, a frequency filter was designed to keep frequencies within the band 14300 to 14900 Hz and remove all others [See Figure 3.62]. The remaining signal, only containing this small range of frequencies, should therefore only rise above a threshold upon occurrence of the marker sound. The signal was automatically processed to locate instances when it exceeded a set threshold. The threshold was set such that it was larger than any occurrence of noise in the signal. The times at which the marker sound was located in the second channel were compiled.

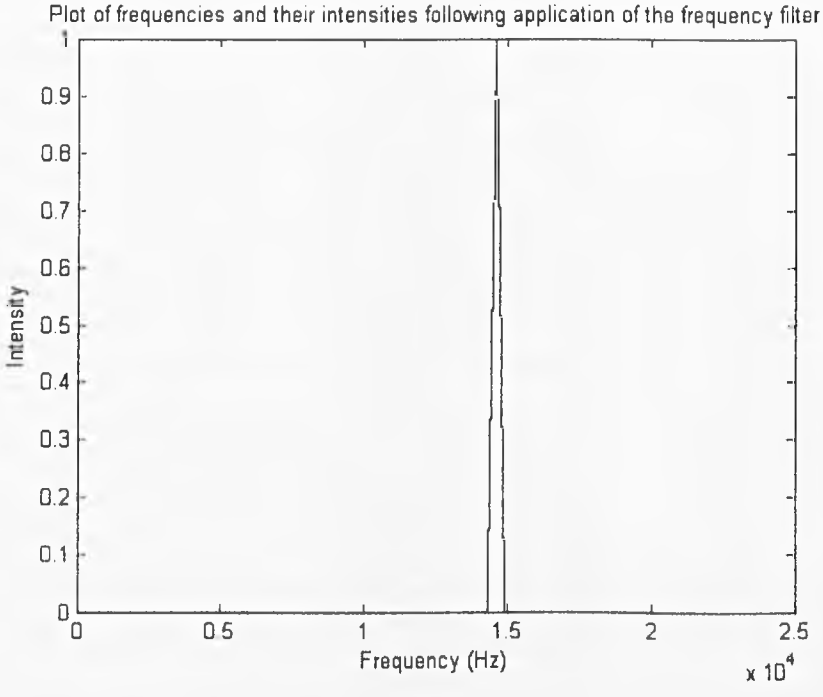


Figure 3.62 Plot showing the effective percentage of each frequency allowed to pass through the digital filter, where 1.0 is the maximum signal allowed.

3.3.8.1.3 Validation of "signal" function

In order to validate this method, the second channel of the file *C10_part1* was analysed according to the described method. This audio recording was a 3 hour portion of a 24 hour recording. The patient had been asked to press the event marker following a bad bout of coughing and to audibly describe the activity being carried out prior to the cough episode. The audio file was listened to and the start times of the

marker were compiled. These were then compared to the results obtained by the software. Table 3-11 shows the comparison of results. All of the uses of the marker detected by the listener were also detected by the program. There was one false positive detected by the program, which was due to the software detecting the same marker twice. However, visual observation of the results by the operator would highlight this during normal use, therefore it was not considered to be a problem.

Table 3-11 Comparison of results of marker start times in audio file C10 as determined by human listening and HACC software

LISTENER	HACC
00:05:47	00:05:47
00:06:40	00:06:40
00:08:40	00:08:40
00:28:07	00:28:07
00:28:22	00:28:22
00:42:25	00:42:25
00:48:49	00:48:49
01:04:31	01:04:31
01:20:16	01:20:16
01:20:59	01:20:59
01:36:42	01:36:42
01:54:59	01:54:59
---	01:55:00
02:48:38	02:48:38

3.3.8.1.4 Compilation of patient activities

Once the event marker locations had been identified on the second channel, it was then necessary to refer to the first channel in order to compile the activities as spoken by the subject. Speech recognition technology would not be practical in such a situation as it would either require training for each subject under recording or it would allow only a limited use of vocabulary to describe activities. For this case,

there would not only be a wide range of activities to allow for, there would also be a number of ways for patients to describe them. Listening to the recordings already made, descriptions have often included unnecessary extra words such as "I was just talking", "I was walking through the hospital" or more simply "stairs". These types of descriptions would require the automatic system to possess semantic knowledge to subsequently derive the correct description. It was therefore decided that the most practical method was to allow the operator to manually listen to the relevant information and enter the appropriate descriptions. The function *activity* uses the marker times compiled previously from the second channel to locate the corresponding time on the first channel, into which the subject would have audibly described the activity. A short section of the recording, immediately following the use of the marker, is then played to the operator, who types in the activity being described. Options are available to repeat play if the description was missed, or to skip the marker without entering any information, for example if there was no description given. The information can then be added on to the 24 cough summary plot to indicate the activities and confirm or refute the theory of activities causing an increase in cough frequency. An example of the activity information compiled is given in Table 3-12. The corresponding signal of the two channels is given in Figure 3.63 illustrating the times when the event marker was used by the patient.

The result of this experiment is a working piece of software which can accurately identify the use of the activity marker and then efficiently compile the necessary descriptive information with minimal user input. This design of the event marker also creates many more applications other than the one described here. The marker can in fact be used for any purpose which requires the ability to identify certain times during a recording. One example is the use of the device to record induced cough models during pharmaceutical efficacy experiments. In this case, the marker can be used to divide up the recording into individual sections, to mark an explanation of each experiment as given by the clinician or to indicate the administering of medication to the patient such that the cough frequency immediately after can be assessed.

Table 3-12 Activity information compiled from the audio recording C10

TIME	ACTIVITY
00:05:47	walking
00:06:40	walking
00:08:40	walking
00:28:07	driving
00:28:22	driving
00:42:25	driving
00:48:49	driving
01:04:31	driving
01:20:16	walking
01:20:59	climbing stairs
01:36:42	sitting
01:54:59	sitting
02:48:38	sitting

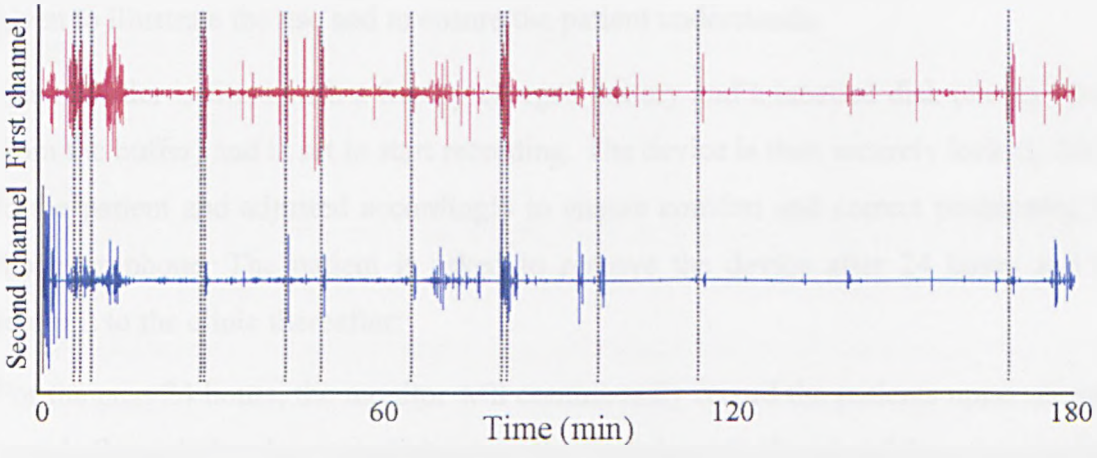


Figure 3.63 Diagram showing the signal measured on both channels of the recording and illustrating the times when the event marker was used

3.4 CASE STUDY

In order to illustrate the methodology developed here, a demonstration of the entire process from data collection to obtaining the results will be presented as a case study.

1. Data collection

A patient is identified as a suitable candidate for a cough study and an appointment is arranged for them to begin the study. The patient is informed that the study is to monitor their cough episodes and to continue their daily activities just as they would normally. They are told that they must wear the device all the time, with the exception of when showering or bathing, in which case the device can be temporarily removed. During the night, the device can either be placed on a bedside table or can be hung over a headboard using the strap, in both cases ensuring that the microphone is placed close to the patient's head for continual recording during sleep.

The patient is instructed on how to use the activity marker such that following a particularly bad attack of coughing, they must press the event marker button and then clearly state the activity they were carrying out prior to the coughing. An example is given to illustrate the use and to ensure the patient understands.

The recorder is fitted with a freshly charged battery and a labelled disk (empty apart from the buffer) and is set to start recording. The device is then securely locked, fitted to the patient and adjusted accordingly to ensure comfort and correct positioning of the microphone. The patient is asked to remove the device after 24 hours and to return it to the clinic thereafter.

For the next 24 hours, the monitor will continuously record the patients upper airway sounds through the chest microphone on the first channel, the use of the event marker on the second channel and will then automatically switch off when the 24 hour study is complete.

2. Data transfer

Once the cough monitor has been returned, the disk is removed and placed in a Hi-MD walkman for transfer via USB to a PC, using the Sony Sonicstage software. The 24 hour recording consists of two tracks, one of 999:59 minutes durations (this is the maximum a file can contain) and one of the remainder, which is approximately 440 minutes. The tracks then need dividing into eight 180 minute portions for processing. Dividing the first track at every 180 minute interval will give five portions plus an additional 100 minutes, the first 80 minutes of the second file must therefore be cut and combined with this to make the sixth 180 minute portion. The remainder of the file is divided in the same way. Each portion must then be converted to WAV format, which is also achieved in Sonicstage and named such that all eight portions are given the same core name with a suffix of "_part1 to 8" accordingly e.g. cough_part1, cough_part2 etc.

3. Data processing

Patient activity marker

In order to process the second channel for the patient activity marker information, this channel must first be isolated. Using Creative Wave Studio software, the signal on the first channel is removed and the whole file is converted into mono format. The signal detection processing can then be applied.

Each minute of the 180 minute file is processed in sequence using the function *signal*. The signal is passed through a Butterworth filter to isolate the frequencies around 15 kHz. The amount of signal remaining following the application of the filter is therefore the amount of 15 kHz signal that was present in the recording. If the amount of signal is above a set threshold, it is assumed to be a positive occurrence of the event marker.

Following compilation of all uses of the event marker, the first channel is then analysed in order to identify the activity information. The software automatically analyses portions of the first channel immediately following the use of the event marker and plays the sound to the operator for a manual entry of descriptions. The

result is a list of the usage of the event marker and the associated activity descriptions.

Cough recognition

In order to perform cough recognition on the first channel, the function *fft_process* is used. This links all the eight portions together and processes each one in sequence. The data is prepared by being divided into manageable chunks for processing and a structural array is set up to store the obtained information. The signal portions are then passed through the digital filter to remove unwanted frequency information and to identify potential cough events. Each event is then further analysed; if it possesses a covariance value above the set threshold and is above a minimum duration, then it is assumed to be a cough event. The information is compiled for later use.

The function *plotcough* then takes the results of the FFT analysis and represents them graphically, initially as three hour portions. The software checks for any events that are longer than one second in duration. The operator has the option of listening to these to firstly ensure that they are coughs and secondly to input the number of coughs in the event. If any changes are made, the plot is updated. Following processing of the entire cough study, the function *combineplot* represents the results for the entire 24 hour period.

4. Analysis of results

The results of the cough count and the activity marker can then be compared by the operator and any correlations or patterns can be noted.

Figure 3.64 is a schematic diagram representing the steps used in the frequency filtering method of cough recognition.

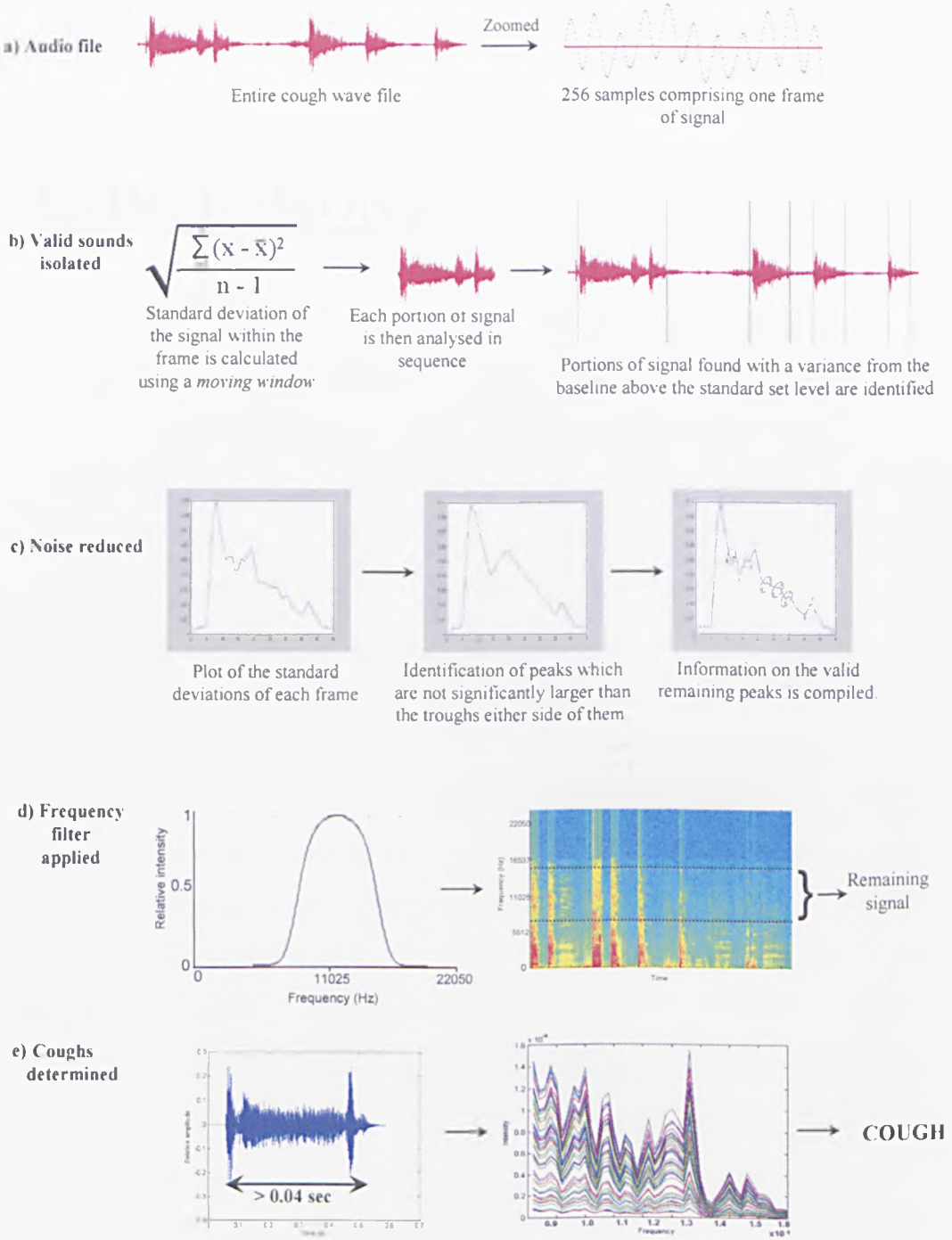


Figure 3.64 Schematic diagram illustrating the frequency filtering approach to cough recognition

CHAPTER 4

CONCLUSIONS

4.1 CONCLUSIONS

An objective, ambulatory, 24 hour patient monitor, along with a method for automatic analysis, has been developed for the purpose of carrying out cough studies using audio recordings. The HACC has been applied to real clinical studies and demonstrated high levels of performance.

4.1.1 HACC hardware

The cough monitor is based around a Sony Hi-MD recorder and dual-channel, capacitor microphone to make audio recordings of patients. The use of audio recordings removes any subjectivity as no patient input of cough frequency or severity is required. Thus, the hardware is objective and provides an unbiased account of cough frequency over a given study period.

The Hi-MD recording media is capable of making up to 34 hours of recordings using the Sony ATRAC method of audio compression. The capability of making 24 hour recordings is necessary in order to study the diurnal changes of cough and to assess cough frequency over a suitably lengthy time period. Modifications to the power source have been necessary in order to make the device portable for 24 hour recordings. Full portability is required in order to study patients in their own environment and during normal activity. The normal cough episode patterns that the patient suffers can then be fully analysed.

In addition, in order to directly and objectively study the correlation of cough episodes with patient activities, an event marker has been incorporated into the device. The event marker consists of a button which, when pressed, inserts a characteristic signal onto the second channel of the microphone; while at the same time the patient can speak into the chest microphone. This allows the patient to highlight information useful to the operator such as the occurrence of certain events which may trigger cough episodes; the effect of these events on cough pattern and frequency can then be studied.

The whole device is enclosed in a secure and convenient case which is non-invasive and comfortable for the patient to wear. Thus, the patients become unaware of the cough monitor and the study can be as close to normal conditions as possible. Unlike some other devices for cough monitoring, the HACC is relatively cheap to build, consisting mainly of commercially available products.

The device has been demonstrated to operate successfully in a number of patient cough studies. Although is has been developed for the purpose of patient cough studies and also for use in cough suppressant efficacy studies, as it is based on audio recordings, its use is not limited to cough. The device can essentially be put to use in many other applications; examples include overnight snore monitoring and the monitoring of infant conditions such as wheezing.

4.1.2 HACC software

The method for automatic analysis of the produced audio recordings has been based around an existing methodology developed prior to this work by Adrie D. Dane. The existing system demonstrated the potential of applying a PNN to the spectral coefficients of sound events in order to perform cough recognition.

24 hour cough recordings were made using the developed HACC hardware on patients with a variety of conditions eliciting cough. The cough recognition system required development for use with the new 24 hour recordings which contain a much wider range of sounds. It was originally decided to base the new system on ideas developed for the existing software. However, the application of this method to the new 24 hour cough studies has proven problematic. The original use of the system involved a series of cough studies; all carried out in the same clinical environment with the same potential background sounds (television, voices of clinic staff etc.) and with a high number of coughs due to the subjects being smokers and chronic cough sufferers. A selection of all sounds within a set number of recordings was then used for the neural network training, which was then applied to the remaining recordings. The possibility of having already sufficiently trained the network with the majority of the sounds that were to be encountered in the validation recordings was very high and thus makes the use of the neural network a successful choice; from the training set,

the network needed only to generalise to a small degree in order to correctly recognise sound events.

However, the new 24 hour cough study data were very different. The number of coughs was significantly lower and the range of background noises was vastly increased. There was a high amount of variation in both datasets, and even with the neural network's ability to generalise, it could not generalise to a sufficient degree as required for this application.

A range of data pre-treatment techniques were employed, in an attempt to make the data more easily separable. MFCC and LPC coefficients were calculated, PCA was used to reduce variation in the dataset, HOSA was used to identify more complex patterns in the signals, PSD plots were calculated to identify any potential for separation and correlation coefficients were used to identify any patterns in the frequency domain. In addition, a range of ANNs and architecture were tested.

A selection of cough and non-cough sounds were studied in the time and frequency domains to identify any characteristics present to allow the classification of similar sounds and their subsequent distinction from dissimilar sounds. It was concluded that the frequency content of the sounds were most likely to contain the relevant information.

Continuing with the idea of working in the frequency domain, focus was then shifted from the characterisation of all frequencies by use of cepstral coefficients to the application of digital filters in order to study a set range of frequencies. Coughs generally contain a wide range of frequencies from low to high, and it was shown that by removing the low frequencies common to many sounds, including speech, the number of interfering background sounds could be significantly reduced. The variation in the frequency content over the duration of a cough was then studied and found to be much higher than in most other sounds. Calculation of covariance was then found to be a suitable method of distinguishing coughs from non-cough sounds.

Initially, the application of the digital filter was intended as a more effective data pre-treatment method before classification by a neural network. However, the resultant incorrectly classified non-coughs were so similar to the cough data due to possessing

very similar frequency information, that the ANN could still not achieve adequate separation. It was concluded that ANNs were not an appropriate method for this application.

The use of the digital filter was therefore developed into a stand-alone, automatic cough recognition system. The system has been demonstrated to correctly classify 82% of all coughs present in audio recordings; the 18% of coughs that were missed were mostly very low in amplitude and would not have been classified as coughs by an experienced cough listener. This figure of 82% is therefore largely underestimated. Only 11% of the events classified coughs were actually non-coughs. Results for sensitivity are calculated to be 82%, specificity is 97% and accuracy is 94%. These are considered to be highly acceptable for clinical applications.

However, for more accurate use during 24 hour studies, and to avoid the problems of encountering more sounds which threaten the performance of the system, it is considered necessary to further investigate methods which could follow up from the data pre-treatment by digital filtering and perform an additional recognition technique to exclude non-cough sounds. It would be recommended to avoid the frequency domain for this as it has been shown that the remaining sounds following digital filtering are too similar in frequency content to be accurately distinguished. A more appropriate technique would be perhaps to investigate the RMS plots of coughs and non-coughs in an attempt to remove interfering non-cough sounds.

The event marker has been demonstrated to successfully highlight patient activities for correlation with cough episodes. A narrow-band digital filter identifies the use of the event marker signal and software then automatically locates the spoken information as given by the subject. This information is then compiled by the operator to avoid problems with speech recognition software, a process which takes a few minutes to complete. Activities such as eating, driving and talking on the telephone which precipitate cough episodes in patients with GORD, for instance, can then be correlated and aid in diagnosis. Whilst processing the cough studies C5 – C7, the chronic cough patient, who was believed to suffer from GORD, exhibited a three-fold increase in coughs following a meal; a strong indication of a correct diagnosis.

4.1.3 Overall conclusions

The HACC device that has been developed was demonstrated to successfully make 24 hour recordings and then to automatically analyse them effectively and efficiently.

Technologies for objective, portable cough monitoring already exist, although each has its disadvantages [See Section 1.2.2.2]. Perhaps the most impressive-sounding cough monitor is the Lifeshirt™ developed by Coyle *et al.* and produced by Vivometrics Inc.[43] It boasts high levels of sensitivity and specificity but fails to reproduce these quoted figures in real applications. It is relatively invasive and constrains the patient to wearing the device, without the potential to remove even for bathing, for the entire study. It is also highly expensive. Hiew *et al.* reported an automatic approach to cough monitoring but without the ability to record for 24 hours[44]. The validation used only sleeping patients and in a study by the same group four years later, the automatic processing step had been exchanged for manual counting. Another group, Matos *et al.* developed an automatic cough recognition system based on HMMs, although the sensitivity and specificity figures were not impressive[46]. One of the most recently developed systems, by Paul et al. lacked both automated analysis and 24 hour recordings[47]. There is therefore a clear requirement for the development of an objective, portable, 24 hour cough monitor along with a system for automated analysis.

4.1.4 Reflection

Two of the HACC devices have been sold to Merck (Philadelphia, USA) for use in cough suppressant efficacy studies. The recorders are used during induced-cough challenges in order to make a permanent record of results for Food and Drug Administration (FDA) checks and also to allow manual cough counting for independent analysis at a later date. Although the application does not require the use of the automatic analysis software, it shows a viable clinical use for the hardware.

In addition, direct comparison studies of the entire HACC system with the Lifeshirt™ are to be carried out by Schering-Plough (New Jersey, USA). As HACC is less-invasive, more convenient, cheaper and potentially more accurate than the

Lifeshirt™, positive results from these studies could result in Schering-Plough using the HACC for their drug efficacy clinical trials. The current method of using the Lifeshirt™ and having results analysed and compiled by Vivometrics Inc. is considered to be too "black-box" for FDA controlled studies. Along with the employment of HACC for clinical trials would come FDA approval of the whole system for such clinical applications.

The device is not limited to drug efficacy studies. The Hull Cough Clinic, based at Castle Hill Hospital, Cottingham, is the world's leading centre for investigation and diagnosis of cough. It is one of only a few specialist cough centres and, as a result, patients are referred there from all parts of the U.K. They have a treatment success rate of over 90%; with over 70% of patients being treated by anti-reflux therapy for GORD related cough. It seems the reason for such a high number of referred GORD sufferers is due to frequent misdiagnosis by the patients' GPs. Unfamiliarity with GORD and its symptoms leads to reluctance to diagnose it and results in ineffective treatment. Many patients with GORD are instead diagnosed as having asthma and waste time and money trying various inhaled medication. In addition, chronic cough is associated with severe debilitating effects, and the time lag between the initial GPs appointment and the correct diagnosis following referral to the cough clinic can be a very traumatic wait. The ideal solution, in order to alleviate the number of patient referrals, the time taken for a diagnosis, and the costs of incorrect treatments, is a diagnostic package for use by GPs. Thus, the cough monitor could be distributed accordingly and software could then process and analyse the results; the characteristic patterns of GORD related cough would then be highlighted and therefore aid in patient diagnosis. This kind of patient monitoring to assist diagnosis and treatment is already employed by some practices in the management of diabetes; patients undergo three-day continuous blood-glucose level monitoring by use of a glucose sensor and recorder, and then receive advice regarding the use of insulin dependent upon the results.

Overall, this work has been successful with the outcome being a usable device for acoustic monitoring and a system for the automatic analysis of audio recordings. The

desirability of the system is well demonstrated by the interest of pharmaceutical companies in using the device for their own clinical trials.

It was considered whether it would be advantageous to produce a system that uses an ANN and trains on a series of voluntary coughs at the start of each recording. The main problem with this would be that the same wide range of interference would still be present and would cause the same problems with recognition. In addition, literature has reported that there is a significant acoustic difference between voluntary and spontaneous coughs that can be detected by ANNs. Finally, the intended design of the system was to be automatic and the process of network training would require a degree of operator experience. However, due to the wide range of cough sounds that occur, it could be useful to incorporate a feature for automatic updating of the network if an unusual cough is encountered. As discussed, this would be unfeasible for a non-cough sound, but it would not take a long time to locate the cough sound and perform an automatic network update. This could perhaps be the only way to approach the problem of being able to classify such a variety of cough sounds into one group.

A relatively long time was spent trying to achieve recognition using ANNs without ultimate success. Previous work had indicated that ANNs had potential to achieve a high level of performance and therefore merited thorough investigation. The work carried out to test the ANN's abilities to separate the real data suggested that it would perhaps be advantageous to train the network on a range of information including both frequency data and magnitude data such as RMS plots. This suggestion was also made following the application of the digital filtering data pre-treatment as a method of discriminating between coughs and those sounds that contained very similar frequency content. Unfortunately both these conclusions were drawn toward the end of the work and time proved to be the limiting factor; it would therefore be a point for future study.

4.1.5 Personal development

This work for me was intended as an opportunity to develop my skills and abilities and to allow me to enter a new field of work. Having completed a degree in

analytical chemistry, the move to programming and informatics was indeed a challenge, although one which I was keen to accept. A large part of the work has involved familiarising myself with Matlab programming language and working through the existing methodology in order to understand how it worked; a process which was largely self-taught and not an easy task. As a result, the work has not only allowed me to develop skills in Matlab and DSP but also several invaluable transferable skills. I have had the opportunity to travel to Philadelphia to coordinate a training course for Merck for the use of the HACC device in their clinical trials. The devices were initially only on loan to Merck but it was reassuring to learn that they later purchased the devices as a result of their satisfaction. The work has been very interesting and feedback from presentations and discussions of the work has been very positive; possibly due to the subject being one to which most people can relate.

4.2 FUTURE WORK

The application of a further classification step following the digital filtering to reduce the 11% of incorrectly identified non-coughs would be advantageous. Research into the characteristic differences between the coughs and the interfering sounds would need to be carried out in order to find potential techniques for recognition. However, following the work already carried out, it seems an automatic method to study the general patterns to cough such as RMS plots would be an ideal starting point.

The discontinuation of the Sony Hi-MD MZ-RH10 and replacement by an upgraded model has called for further development of the device, which therefore allows any necessary improvements to be made. Future devices, currently in production, possibly would benefit from reverting back to the line-in method of recording, as this was shown to possess advantages over the noisier microphone-input.

The automated analysis software requires development into a stand-alone piece of software, independent of Matlab, and with a user-friendly front-end for ease of use, in order to be able to complete the original aim of creating a diagnostic package for use by GPs in treating patients with GORD. The integration of the activity marker results and the 24 hour cough summary would be a useful and easily interpretable output.

Additionally, the system could be designed to perform automatic correlation of trigger factors and cough episodes in order to provide an instant diagnosis.

REFERENCES

1. Dalmasso, F.; Prota, R., Snoring: analysis, measurement, clinical implications and applications. *European Respiratory Journal* **1996,** 9, 146-159.

2. Siuru, B., Applying acoustic monitoring to medical diagnostics. *Sensors* March, 1997.

3. Irwin, R. S.; Boulet, L. P.; Cloutier, M. M.; Fuller, R.; Gold, P. M.; Hoffstein, V.; Ing, A. J.; McCool, F. D.; O'Byrne, P.; Poe, R. H.; Prakash, U. B. S.; Pratter, M. R.; Rubin, B. K., Managing cough as a defense mechanism and as a symptom - A consensus panel report of the American College of Chest Physicians. *Chest* **1998,** 114, (2), 133S-181S.

4. Fuller, R. W.; Jackson, D. M., Physiology and treatment of cough. *Thorax* **1990,** 45, (6), 425-430.

5. McGarvey, L. P. A., Idiopathic chronic cough: a real disease or a failure of diagnosis? *Cough* **2005,** 1, 9.

6. Ida, H., The nonnarcotic antitussive drug dimemorfan: A review. *Clinical Therapeutics* **1997,** 19, (2), 215-231.

7. Chung, K. F.; Chang, A. B., Therapy for cough: Active agents. *Pulmonary Pharmacology & Therapeutics* **2002,** 15, (3), 335-338.

8. Dicpinigaitis, P. V., Potential new cough therapies. *Pulmonary Pharmacology & Therapeutics* **2004,** 17, (6), 459-462.

9. McLeod, R. L.; Bolser, D. C.; Jia, Y. L.; Parra, L. E.; Mutter, J. C.; Wang, X.; Tulshian, D. B.; Egan, R. W.; Hey, J. A., Antitussive effect of nociceptin/orphanin FQ in experimental cough models. *Pulmonary Pharmacology & Therapeutics* **2002,** 15, (3), 213-216.

10. Dicpinigaitis, P. V.; Gayle, Y. E., Effect of guaifenesin on cough reflex sensitivity. *Chest* **2003,** 124, (6), 2178-2181.

11. Grattan, T. J.; Marshall, A. E.; Higgins, K. S.; Morice, A. H., The effect of inhaled and oral dextromethorphan on citric-acid induced cough in man. *British Journal of Clinical Pharmacology* **1995,** 39, (3), 261-263.

12. Hoglund, N. J.; Michaelsson, M., A method for determining the cough threshold with some preliminary experiments on the effect of codeine. *Acta Physiologica Scandinavica* **1950,** 21, 168-173.

13. Schroeder, K.; Fahey, T., Systematic review of randomised controlled trials of over the counter cough medicines for acute cough in adults. *British Medical Journal* **2002,** 324.

14. French, C. L.; Irwin, R. S.; Curley, F. J.; Krikorian, C. J., Impact of chronic cough on quality of life. *Archives of Internal Medicine* **1998,** 158, (15), 1657-1661.

15. Dicpinigaitis, P. V., Angiotensin-converting enzyme inhibitor-induced cough - ACCP evidence-based clinical practice guidelines. *Chest* **2006,** 129, (1), 169S-173S.

16. Corrao, W. M.; Braman, S. S.; Irwin, R. S., Chronic cough as the sole presenting manifestation of bronchial asthma. *New England Journal of Medicine* **1979,** 300, (12), 633-637.

17. Morice, A., The diagnosis and management of chronic cough. *European Respiratory Journal* **2004,** 24, 481-492.

18. Chung, K. F., Assessment and measurement of cough: The value of new tools. *Pulmonary Pharmacology & Therapeutics* **2002,** 15, (3), 267-272.

19. Chang, A. B.; Newman, R. G.; Carlin, J. B.; Phelan, P. D.; Robertson, C. F., Subjective scoring of cough in children: Parent-completed vs child-completed diary cards vs an objective method. *European Respiratory Journal* **1998,** 11, (2), 462-466.

20. Birring, S. S.; Prudon, B.; Carr, A. J.; Singh, S. J.; Morgan, M. D. L.; Pavord, I. D., Development of a symptom specific health status measure for patients with chronic cough: Leicester Cough Questionnaire (LCQ). *Thorax* **2003,** 58, (4), 339-343.

21. French, C. T.; Irwin, R. S.; Fletcher, K. E.; Adams, T. M., Evaluation of a cough-specific quality-of-life questionnaire. *Chest* **2002,** 121, (4), 1123-1131.

22. Archer, L. N. J.; Simpson, H., Night cough counts and diary card scores in asthma. *Archives of Disease in Childhood* **1985,** 60, (5), 473-474.

23. Falconer, A.; Oldman, C.; Helms, P., Poor agreement between reported and recorded nocturnal cough in asthma. *Pediatric Pulmonology* **1993,** 15, (4), 209-211.

24. Kastelik, J. A., Acid inhalation cough challenge. In *Acute and Chronic Cough*, Redington, A. E.; Morice, A., Eds. Taylor & Francis Group: 2005; Vol. 205, pp 177-194.

25. Dicpinigaitis, P. V., Capsaicin inhalation cough challenge. In *Acute and Chronic Cough*, Redington, A. E.; Morice, A., Eds. Taylor & Francis Group: 2005; Vol. 205, pp 161-175.

26. Fontana, G. A.; Lavorini, F.; Pistolesi, M., Water aerosols and cough. In *Acute and Chronic Cough*, Redington, A. E.; Morice, A., Eds. Taylor & Francis Group: 2005; Vol. 205, pp 195-214.

27. Sevelius, H.; Colmore, J. P., Objective assessment of antitussive agents in patients with chronic cough. *The Journal of New Drugs* **1966,** 6, (4), 216-223.

28. Woolf, C. R.; Rosenberg, A., Objective assessment of cough suppressants under clinical conditions using a tape recorder system. *Thorax* **1964,** 19, 125-130.

29. Gravenstein, J. S.; Devloo, R. A.; Beecher, H. K., Effect of antitussive agents on experimental and pathological cough in man. *Journal of Applied Physiology* **1954,** 7, (2), 119-139.

30. Keats, A. S.; Telford, J., Subjective effects of nalorphine in hospitalized patients. *Journal of Pharmacology and Experimental Therapeutics* **1957,** 119, (3), 370-377.

31. Loudon, R. G.; Romans, W. E., Cough-monitoring equipment. *Medical Research Engineering* **1967,** 6, 25-27.

32. Loudon, R. G.; Brown, L. C., Cough frequency in patients with respiratory disease. *American Review of Respiratory Disease* **1967,** 96, (6), 1137-1143.

33. Thomas, J.; Heurich, A. E.; Shepherd, D. A.; Sanzari, N. P., System for clinical assessment of antitussive activity of caramiphen. *Current Therapeutic Research-Clinical and Experimental* **1974,** 16, (10), 1082-1090.

34. Matthys, H.; Bleicher, B.; Bleicher, U., Dextromethorphan and codeine - objective assessment of antitussive activity in patients with chronic cough. *Journal of International Medical Research* **1983,** 11, (2), 92-100.

35. Salmi, T.; Sovijarvi, A. R. A.; Brander, P.; Piirila, P., Long-term recording and automatic analysis of cough using filtered acoustic signals and movements on static charge sensitive bed. *Chest* **1988,** 94, (5), 970-975.

36. Chang, A. B.; Newman, R. G.; Phelan, P. D.; Robertson, C. F., A new use for an old Holter monitor: An ambulatory cough meter. *European Respiratory Journal* **1997,** 10, (7), 1637-1639.

37. Eccles, R., Codeine, cough and upper respiratory infection. *Pulmonary Pharmacology & Therapeutics* **1996,** 9, (5-6), 293-297.

38. Hsu, J. Y.; Stone, R. A.; Logan-Sinclair, R. B.; Worsdell, M.; Busst, C. M.; Chung, K. F., Coughing frequency in patients with persistent cough - assessment using a 24-hour ambulatory recorder. *European Respiratory Journal* **1994,** 7, (7), 1246-1253.

39. Munyard, P.; Busst, C.; Logan-Sinclair, R.; Bush, A., A new device for ambulatory cough recording. *Pediatric Pulmonology* **1994,** 18, (3), 178-186.

40. Subburaj, S.; Parvez, L.; Rajagopalan, T. G., Methods of recording and analysing cough sounds. *Pulmonary Pharmacology & Therapeutics* **1996,** 9, (5-6), 269-279.

41. Pavesi, L.; Subburaj, S.; Porter-Shaw, K., Application and validation of a computerized cough acquisition system for objective monitoring of acute cough - a meta-analysis. *Chest* **2001,** 120, (4), 1121-1128.

42. Rietveld, S.; Rijssenbeek-Nouwens, L. H. M., Diagnostics of spontaneous cough in childhood asthma - results of continuous tracheal sound recording in the homes of children. *Chest* **1998,** 113, (1), 50-54.

43. Coyle, M. A.; Keenan, D. B.; Henderson, L. S.; Watkins, M. L.; Haumann, B. K.; Mayleben, D. W.; Wilson, M. G., Evaluation of an ambulatory system for the quantification of cough frequency in patients with chronic obstructive pulmonary disease. *Cough* **2005,** 1, (3), 7.

44. Hiew, Y. H.; Smith, J. A.; Earis, J. E.; Cheetham, B. M. G.; Woodcock, A. A., DSP algorithm for cough identification and counting. *IEEE International Conference on Acoustics, Speech and Signal Processing* **2002,** 4, 3888-3891.

45. Smith, J. A.; Owen, E. C.; Jones, A. M.; Dodd, M. E.; Webb, A. K.; Woodcock, A., Objective measurement of cough during pulmonary exacerbations in adults with cystic fibrosis. *Thorax* **2006,** 61, (5), 425-429.

46. Matos, S.; Birring, S. S.; Pavord, I. D.; Evans, D. H., Detection of cough signals in continuous audio recordings using hidden Markov models. *IEEE Transactions on Biomedical Engineering* **2006,** 53, (6), 1078-1083.

47. Paul, I. M.; Wai, K.; Jewell, S. J.; Shaffer, M. L.; Varadan, V. V., Evaluation of a new self-contained, ambulatory, objective cough monitor. *Cough* **2006,** 2, (7).

48. Barry, S. J.; Dane, A. D.; Morice, A. H.; Walmsley, A. D., The automatic recognition and counting of cough. *Cough* **2006,** 2, (8).

49. Korpas, J.; Sadlonova, J.; Vrabec, M., Analysis of the cough sound: An overview. *Pulmonary Pharmacology & Therapeutics* **1996,** 9, (5-6), 261-268.

50. Piirila, P.; Sovijarvi, A. R. A., Objective Assessment of Cough. *European Respiratory Journal* **1995,** 8, (11), 1949-1956.

51. Yanagihara, N.; von Leden, H.; Werner-Kukuk, E., The physical parameters of cough: The larynx in a normal single cough. *Acta Otolaryngologica* **1966,** 61, 495-605.

52. Smith, J. A.; Ashurst, L.; Jack, S.; Woodcock, A. A.; Earis, J. E., The description of cough sounds by healthcare professionals. *Cough* **2006,** 2, (1).

53. Piirila, P.; Sovijarvi, A. R. A., Differences in acoustic and dynamic characteristics of spontaneous cough in pulmonary diseases. *Chest* **1989,** 96, (1), 46-53.

54. Toop, L. J.; Dawson, K. P.; Thorpe, C. W., A portable system for the spectral analysis of cough sounds in asthma. *Journal of Asthma* **1990,** 27, (6), 393-397.

55. Thorpe, C. W.; Fright, W. R.; Toop, L. J.; Dawson, K. P., A microcomputer-based interactive cough sound analysis system. *Computer Methods and Programs in Biomedicine* **1991,** 36, (1), 33-43.

56. Thorpe, C. W.; Toop, L. J.; Dawson, K. P., Towards a quantitative description of asthmatic cough sounds. *European Respiratory Journal* **1992,** 5, (6), 685-692.

57. Doherty, M. J.; Wang, L. J.; Donague, S.; Pearson, M. G.; Downs, P.; Stoneman, S. A. T.; Earis, J. E., The acoustic properties of capsaicin-induced cough in healthy subjects. *European Respiratory Journal* **1997,** 10, 202-207.

58. Olia, P. M.; Sestini, P.; Vagliasindi, M., Acoustic parameters of voluntary cough in healthy non-smoking subjects. *Respirology* **2000,** 5, 271-275.

59. Van Hirtum, A.; Berckmans, D., Automated recognition of spontaneous versus voluntary cough. *Medical Engineering & Physics* **2002,** 24, (7-8), 541-545.

60. Murata, A.; Ohota, N.; Shibuya, A.; Ono, H.; Kudoh, S., New non-invasive automatic cough counting program based on 6 types of classified cough sounds. *Internal Medicine* **2006,** 45, (6), 391-397.

61. Murata, A.; Taniguchi, Y.; Hashimoto, Y.; Kaneko, Y.; Takasaki, Y.; Kudoh, S., Discrimination of productive and non-productive cough by sound analysis. *Internal Medicine* **1998,** 37, (9), 732-735.

62. Rabiner, L.; Juang, B. H., Fundamentals of speech recognition. In *Fundamentals of Speech Recognition*, Prentice Hall: New Jersey, 1993; pp 1-10.

63. Davis, K. H.; Biddulph, R.; Balashek, S., Automatic recognition of spoken digits. *Journal of the Acoustical Society of America* **1952,** 24, (6), 637-642.

64. Olson, H. F.; Belar, H., Phonetic typewriter. *Journal of the Acoustical Society of America* **1956,** 28, (6), 1072-1081.

65. Velichko, V. M.; Zagoruyko, N. G., Automatic recognition of 200 words. *International Journal of Man-Machine Studies* **1970,** 2, (3), 223-234.

66. Sakoe, H.; Chiba, S., Dynamic-programming algorithm optimization for spoken word recognition. *Ieee Transactions on Acoustics Speech and Signal Processing* **1978,** 26, (1), 43-49.

67. Itakura, F., Line spectrum representation of linear predictor coefficients of speech signals. *Journal of the Acoustical Society of America* **1975,** 57, S35-S35.

68. Rabiner, L.; Juang, B. H., The speech signal: Production, perception and acoustic-phonetic characterisation. In *Fundamentals of Speech Recognition*, Prentice-Hall: New Jersey, 1993; pp 11-68.

69. Cooley, J. W.; Tukey, J. W., An algorithm for the machine calculation of complex Fourier Series. *Mathematics of Computation* **1965,** 19, 297-301.

70. Bogert, B. P.; Healy, M. J. R.; Tukey, J. W., The quefrency alanysis of time series for echoes: Cepstrum, pseudo-autocovariance, cross-cepstrum and saphe cracking. In *Proceedings of the Symposium on Time Series Analysis*, Rosenblatt, M., Ed. Wiley: New York, 1963; pp 209-243.

71. Stevens, S.; Volkmann, J., The relation of pitch to frequency: a revised scale. *The American Journal of Psychology* **1940,** 53, (3), 329-353.

72. Picone, J., Continuous speeech recognition using Hidden Markov Models. *IEEE Acoustic Speech and Signal Processing* **1990,** 7, (3), 26-41.

73. Richardson, M.; Bilmes, J.; Diorio, C., Hidden-articulator Markov models for speech recogntion. *Speech Communication* **2000,** 41, 511-529.

74. Massart, D. L.; Vandeginste, B. G. M.; Buydens, L. M. C.; De Jong, S.; Lewi, P. J.; Smeyers-Verbeke, J., *Handbook of Chemometrics and Qualimetrics: Part A*. Elsevier: Amsterdam, 1997.

75. Savitsky, A.; Golay, M. J. E., Smoothing and differentiation of data by simplified least squares procedures. *Analytical Chemistry* **1964,** 36, (8), 1627-1639.

76. Malinowski, E. R., *Factor Analysis in Chemistry*. Second ed.; John Wiley & Sons, Inc.: 1991.

77. McCulloch, M.; Pitts, W., A logical calculus of the ideas immanent in nervous activity. *Bulletin of Mathematical Biophysics* **1943,** 5, 115-133.

78. Hebb, D. O., *The Organization of Behaviour*. Wiley: New York, 1949.

79. Haykin, S., *Neural Networks, A Comprehensive Foundation*. Second ed.; Prentice-Hall Inc.: New Jersey, 1999; p 842.

80. Gabor, D.; Wilby, W. P. L.; Woodcock, R., A universal, non-linear filter, predictor and simulator which optimizes itself by a learning process. *Proceedings of the Institution of Electrical Engineers, London* **1960,** 108, 422-435.

81. Rochester, N.; Holland, J. H.; Haibt, L. H.; Duda, W. L., Tests on a cell assembly theory of the action of the brain, using a large digital computer. *IRE Transactions on Information Theory* **1956,** IT-2, 80-93.

82. Rosenblatt, F., The perceptron: A probabilistic model for information storage and organisation in the brain. *Psychological Review* **1958,** 65, 386-408.

83. Widrow, B.; Hoff Jr., M. E. In *Adaptive switching circuits*, IRE Wescon, New York, 1960; New York, 1960; pp 96-104.

84. Minsky, M.; Papert, S., *Perceptrons*. MIT Press: Cambridge, 1969.

85. Zupan, J.; Gasteiger, J., Neural networks: A new method for solving chemica problems or just a passing phase? *Analytica Chimica Acta* **1991,** 248, 1-30.

86. Von der Malsburg, C., Self-organisation of orientation sensitive cells in the striate cortex. *Kybernetik* **1973,** 14, 85-100.

87. Willshaw, D. J.; Von der Malsburg, C., How patterned neural connections can be set up by self-organization. *Proceedings of the Royal Society of London* **1975,** 194, 431-445.

88. Werbos, P. J. Beyond Regression: New Tools for Prediction and Analysis in Behavioural Sciences. Harvard University, 1974.

89. Hopfield, J. J., Neural networks and physical systems with emergent collective computational abilities. *Proceedings of the National Academy of Sciences of the United States of America* **1982,** 79, 2554-2558.

90. Kohonen, T., Self-organized formation of topologically correct feature maps. *Biological Cybernetics* **1982,** 43, (1), 59-69.

91. Ackley, D. H.; Hinton, G. E.; Sejnowski, T. J., A learning algorithm for Boltzmann machines. *Cognitive Science* **1985,** 9, 147-169.

92. Rumelhart, D. E.; Hinton, G. E.; Williams, R. J., Learning representations by back-propagating errors. *Nature* **1986,** 323, (6088), 533-536.

93. Broomhead, D. S.; Lowe, D., Multivariable functional interpolation and adaptive networks. *Complex Systems* **1988,** 2, 321-355.

94. Shao, Y. P.; A., W.; Oppenheim, C., Knowledge-based systems in retail banking: A survey of current practice. *Intelligent Systems in Accounting, Finance and Management* **1997,** 6, 73-81.

95. Brause, R.; Langsdorf, T.; Hepp, M. In *Neural data mining for credit card fraud detection*, Proceedings of the International Conference on Tools with Artificial Intelligence, 1999, pp 103-106.

96. Barson, P.; Field, S.; Davey, N.; McAskie, G.; Frank, R., The detection of fraud in mobile phone networks. *Neural Network World* **1996,** 6, (4), 477-484.

97. Kaefer, F.; Heilman, C. M.; Ramenofsky, S. D., A neural network application to consumer classification to improve the timing of direct marketing activities. *Computers and Operations Research* **2005,** 32, (10), 2595-2615.

98. Wu, A.; Hsieh, W. W.; Tang, B., Neural network forecasts of the tropical Pacific sea surface temperatures. *Neural Networks* **2006,** 19, 145-154.

99. Deisingh, A. K.; Stone, D. C.; Thompson, M., Applications of electronic noses and tongues in food analysis. *International Journal of Food Science and Technology* **2004,** 39, (6), 587-604.

100. Rajagopalan, R.; Rajagopalan, P., Applications of neural network in manufacturing. *Proceedings of the 29th Annual Hawaii International Conference on System Sciences* **1996**, 447-458.

101. Lisboa, P. J. G., A review of evidence of health benefit from artificial neural networks in medical intervention. *Neural Networks* **2002,** 15, (1), 11-39.

102. Cenci, M.; Nagar, C.; Vecchione, A., PAPNET-assisted primary screening of conventional cervical smears. *Anticancer Research* **2000,** 20, (5C), 3887-3889.

103. Brake, G. M. T.; Karssemeijer, N.; Hendriks, J. H. C. L., An automatic method to discriminate malignant masses from normal tissue in digital mammograms. *Physics in Medicine and Biology* **2000,** 45, (10), 2843-2857.

104. Lee, S. K.; Lo, C. S.; Wang, C. M.; Chung, P. C.; Chang, C. I.; Yang, C. W.; Hsu, P. C., A computer-aided design mammography screening system for detection and classification of microcalcifications. *International Journal of Medical Informatics* **2000,** 60, (1), 29-57.

105. Andrisevic, N.; Ejaz, K.; Rios-Gutierrez, F.; Alba-Flores, R.; Nordehn, G.; Burns, S., Detection of heart murmurs using wavelet analysis and artificial neural networks. *Journal of Biomechanical Engineering* **2005,** 127, (6), 899-904.

106. Wu, W.; Walczak, B.; Massart, D. L.; Heuerding, S.; Erni, F.; Last, I. R.; Prebble, K. A., Artficial neural networks in classification of NIR spectral data: Design of the training set. *Chemometrics and Intelligent Laboratory Systems* **1995,** 33, 35-46.

107. Zupan, J.; Gasteiger, J., *Neural Networks for Chemists, An Introduction.* VCH-Verlag: Weinheim, 1993; p 300.

108. Huang, L. L.; Shimizu, A., A multi-expert approach for robust face detection. *Pattern Recognition* **2006,** 39, 1695-1703.

109. Morris, S. A.; Wu, Z.; Yen, G., A SOM mapping technique for visualizing documents in a database. In *International Joint Conference on Neural Networks*, Washington D.C., USA, 2001.

110. Egmont-Peterson, M.; de Riddler, D.; Handels, H., Image processing with neural networks - a review. *Pattern Recognition* **2002,** 35, 2279-2301.

111. Chedad, A.; Moshou, D.; Aerts, J. M.; Van Hirtum, A.; Ramon, H.; Berckmans, D., Recognition system for pig cough based on probabilistic neural networks. *Journal of Agricultural and Engineering Research* **2001,** 79, (4), 449-457.

112. Moshou, D.; Chedad, A.; Van Hirtum, A.; De Baerdemaeker, J.; Berckmans, D.; Ramon, H., An intelligent alarm for early detection of swine epidemics based on neural networks. *Transactions of the American Society of Agricultural Engineers* **2001,** 44, (1), 167-174.

113. Moshou, D.; Chedad, A.; Van Hirtum, A.; De Baerdemaeker, J.; Berckmans, D.; Ramon, H., Neural recognition system for swine cough. *Mathematics and Computers in Simulation* **2001,** 56, (4-5), 475-487.

114. Van Hirtum, A.; Berckmans, D., Fuzzy approach for improved recognition of citric acid induced piglet coughing from continuous registration. *Journal of Sound and Vibration* **2003,** 266, (3), 677-686.

115. Specht, D. F. In *Probabilistic neural networks for classification, mapping, or associative memory*, 1988; 1988; pp 525-532.

116. Specht, D. F., Probabilistic neural networks. *Neural Networks* **1990,** 3, (1), 109-118.

117. MacQueen, J., Some methods for classification and analysis of multivariate observations. In *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability.*, Le Cam, L. A.; Neyman, J., Eds. University of California Press: Berkeley, California, 1967; Vol. 1, pp 281-297.

118. Sony, *Portable MD recorder Operation Instructions*. Sony Electronics Inc.: 2005.

119. Widdicombe, J. G., Symposium on the cough reflex. *Clinical Respiratory Physiology* **1987,** 23, (Suppl. 10), 73-74.

Appendix A

Program Code

1 Program code

1.1 Data pre-treatment,

1.1.1 'Dataprep'

```
function [Xtrain,Ytrain,Xval,Yval,calibindex,validindex] = dataprep(fname)
% Preparation of data for training network
% Developed by Samantha Barry
global COUGH_

[pname,fname]=fileparts(fname);
if isempty(pname)
  pname=COUGH_.path;
end
fname=fullfile(pname,fname);

fnamephn=(fname(1:end-4));
extension='.mat';
data='_phn';
fnamephn=[fname, data, extension];
if exist (fnamephn)
   load(fnamephn);
   disp('phn loaded')
end

load(fsname([fname '_peaks']),'p');
k=cough(fname);
p=k.p;

% take 1/3 of data for validation
nevents=size(p,2);
reduce=3;

y=zeros(1,(size(p,2)));
for i=1:nevents
   y(1,i)=p(i).cough;
end
% cough and non-cough events are not equal in frequency - need to split up to
% enable an equal distribution of both events in training and validation sets

% isolating coughs
coughs=find(y);
ycough=y(:,coughs);
xcough=p(:,coughs);
ncough=length(coughs);

% isolating non-coughs
noncoughs=find(y==0);
ynoncough=y(:,noncoughs);
xnoncough=p(:,noncoughs);
noncough=length(noncoughs);

% remove chosen validation data to leave training data - coughs!
xval=xcough(:,1:reduce:ncough);
yval=ycough(:,1:reduce:ncough);
```

```
xtrain=xcough;
ytrain=ycough;
xtrain(:,1:reduce:ncough)=[];
ytrain(:,1:reduce:ncough)=[];
% do same with indices
valind=coughs(:,1:reduce:ncough);
trainind=coughs;
trainind(:,1:reduce:ncough)=[];

% remove chosen validation data to leave training data - non-coughs
xvalnon=xnoncough(:,1:reduce:noncough);
yvalnon=ynoncough(:,1:reduce:noncough);
xtrainnon=xnoncough;
ytrainnon=ynoncough;
xtrainnon(:,1:reduce:noncough)=[];
ytrainnon(:,1:reduce:noncough)=[];
% do same with indices
valindnon=noncoughs(:,1:reduce:noncough);
trainindnon=noncoughs;
trainindnon(:,1:reduce:noncough)=[];

% performing the features function
disp('extracting features...')

%whichones=COUGH_.feat.whichones;
std_tresh=COUGH_.features;
%if isa(whichones,'char')
 %whichones={whichones};
 whichones={'melcepst'};
   %whichones={'melcepst' 'lpcauto'};

 %end
nfeatures=length(whichones);
options=[];
o=getoptions(options);

% perform signal processing on training data
% choose either just cough data (indices=trainind)
% or just non-cough data (indices=trainindnon)
Xtrain=[];
Ytrain=[];

% decrease size of noncough data to be almost equal to cough data
a=size(trainind,2);
b=size(trainindnon,2);
   if b>a
     div=(round(b/a)/2);
     traincut=trainindnon;
     traincut=traincut(1:div:b);

     %traincut=traincut';
   else traincut=trainindnon;
   end

no_rowstr=[];
calibindex=([trainind'; traincut'])';
indices=([trainind'; traincut'])';

% calculate features for calibration data
for i=indices;
```

```
   if (get(k,'cough',i)==-1)
     continue
   end

  sig=get(k,'peaks_plus',i);
  onlythese=stdsel(sig,std_tresh,'peaks_plus');

  [f]=getfeatures(sig,get(k,'fs'),whichones,nfeatures,onlythese,o);

  norows=size(f);
  rowsind=[norows i];
  no_rowstr=[no_rowstr; rowsind];

  % Select option...(delete as apt)
%1
[u,s,v]=svd(f);
loads=(v(:,1:15));

newtrindex=zeros(size(f,1),1);
nsize=size(no_rowstr,1);
m=1;
for n=1:nsize
   if no_rowstr(n,1)==1
      newtrindex(m,1)=no_rowstr(n,3);
      m=m+1;
   elseif no_rowstr(n,1)>1
      dupl=no_rowstr(n,1);
      newtrindex(m:(m+dupl)-1,1)=no_rowstr(n,3);
      m=m+dupl;
   end
end
%2
  data = f'*f;
  [u,s,v]=svd(data);
  s=sqrt(s);
  %[u,s,v]=svd(f);
  %loads=[(v(1,1:15).*s(1,1)) (v(2,1:15).*s(2,2)) (v(3,1:15).*s(3,3))];
  loads=[(v(1,1:15).*s(1,1)) (v(2,1:15).*s(2,2)) (v(3,1:15).*s(3,3)) (v(4,1:15).*s(4,4))];
  %loads=[(v(1,1:15).*s(1,1)) (v(2,1:15).*s(2,2)) (v(3,1:15).*s(3,3)) (v(4,1:15).*s(4,4))];
  %loads=[(v(1,1:15).*s(1,1)) (v(2,1:15).*s(2,2)) (v(3,1:15).*s(3,3)) (v(4,1:15).*s(4,4))
%(v(5,1:15).*s(5,5))];
  %loads=[(v(1,1:15).*s(1,1)) (v(2,1:15).*s(2,2)) (v(3,1:15).*s(3,3)) (v(4,1:15).*s(4,4))
%(v(5,1:15).*s(5,5)) (v(6,1:15).*s(6,6))];
   %if size(f,1)>2
   %   load=[(v(1,1:15).*s(1,1)) (v(2,1:15).*s(2,2)) (v(3,1:15).*s(3,3))];
   %elseif size(f,1)==2
   %   load=[(v(1,1:15).*s(1,1)) (v(2,1:15).*s(2,2)) (v(2,1:15).*s(2,2))];
   %else load=[(v(1,1:15).*s(1,1)) (v(1,1:15).*s(1,1)) (v(1,1:15).*s(1,1))];
   % end
  %load=(v(1,1:20).*s(1,1));

Xtrain=[Xtrain; loads];
Ytrain=[Ytrain; get(k,'cough',i)*ones(size(f,1),1)];

   if p(i).cough==1
      type='cough';
   else
      type='noncough';
   end
```

```
end

% perform signal processing on validation data
% choose either just cough data (indices=valind)
% or just non-cough data (indices=valindnon)

Xval=[];
Yval=[];
%valindex=[];

% increase size of cough data to be almost equal to non-cough data
a=size(valind,2);
b=size(valindnon,2);

    if b>a
      div=(round(b/a)/2);
      valcut=valindnon';
      valcut=valcut(1:div:b);
      valcut=valcut';
    else valcut=valindnon;
    end

no_rowsva=[];
validindex=([valind'; valcut'])';
indices=([valind'; valcut'])';
%indices=valind;
% calculate features for validation data
for i=indices;

    if (get(k,'cough',i)==-1)
     continue
   end

  sig=get(k,'peaks_plus',i);
  onlythese=stdsel(sig,std_tresh,'peaks_plus');

  [f,g]=getfeatures(sig,get(k,'fs'),whichones,nfeatures,onlythese,o);
  norows=size(f);
  rowsind=[norows i];
  no_rowsva=[no_rowsva; rowsind];

  % Select option...(delete as apt)
%1
[u,s,v]=svd(f);
loads=(v(:,1:15));

newvalindex=zeros(size(f,1),1);
nsize=size(no_rowstr,1);
m=1;
for n=1:nsize
   if no_rowstr(n,1)==1
      newvalindex(m,1)=no_rowstr(n,3);
      m=m+1;
   elseif no_rowstr(n,1)>1
      dupl=no_rowstr(n,1);
      newvalindex(m:(m+dupl)-1,1)=no_rowstr(n,3);
      m=m+dupl;
   end
end
%2
```

```
  data = f'*f;
  [u,s,v]=svd(data);
  s=sqrt(s);
  %loads=[(v(1,1:15).*s(1,1)) (v(2,1:15).*s(2,2)) (v(3,1:15).*s(3,3))];
  loads=[(v(1,1:15).*s(1,1)) (v(2,1:15).*s(2,2)) (v(3,1:15).*s(3,3)) (v(4,1:15).*s(4,4))];
  %loads=[(v(1,1:10).*s(1,1)) (v(2,1:10).*s(2,2)) (v(3,1:10).*s(3,3)) (v(4,1:10).*s(4,4))];

  %loads=[(v(1,1:15).*s(1,1)) (v(2,1:15).*s(2,2)) (v(3,1:15).*s(3,3)) (v(4,1:15).*s(4,4))
%(v(5,1:15).*s(5,5))];
  %loads=[(v(1,1:15).*s(1,1)) (v(2,1:15).*s(2,2)) (v(3,1:15).*s(3,3)) (v(4,1:15).*s(4,4))
%(v(5,1:15).*s(5,5)) (v(6,1:15).*s(6,6))];
    %if size(f,1)>2
    %   load=[(v(1,1:15).*s(1,1)) (v(2,1:15).*s(2,2)) (v(3,1:15).*s(3,3))];
    %elseif size(f,1)==2
    %   load=[(v(1,1:15).*s(1,1)) (v(2,1:15).*s(2,2)) (v(2,1:15).*s(2,2))];
    %else load=[(v(1,1:15).*s(1,1)) (v(1,1:15).*s(1,1)) (v(1,1:15).*s(1,1))];
    % end
  %load=(v(1,1:20).*s(1,1));

  Xval=[Xval; loads];
 Yval=[Yval; get(k,'cough',i)*ones(size(f,1),1)];
end

function options = getoptions(opts)
% GETOPTIONS -
%  Usage:    options = getoptions(opts)
%  Created:  Mon Jul 23 15:17:56 2001
%  Update:   Fri Nov 09 14:04:00 2001
%  Author:   Adrie Dane
global COUGH_

options=COUGH_.feat.options;

for i=1:2:length(opts),
  options=setfield(options,opts{i},opts{i+1});
end
```

1.1.2 Correlation coefficients

```
f=getfeatures(sig,get(k,'fs'),whichones,nfeatures,onlythese,o);
  correl=corrcoef(f);
  correlrow=[];
  for r=1:20
     correlrow=[correlrow correl(r,1:20)];
  end

Xtrain=[Xtrain; correlrow];
    Ytrain=[Ytrain; get(k,'cough',i)];


%To divide the matrix of correlation coefficients:

f=getfeatures(sig,get(k,'fs'),whichones,nfeatures,onlythese,o);
  f(:,16:42)=[];
  correl=corrcoef(f);
  correlrow=[];
  line=[];
```

```
 norows=size(correl,1);
 for r=1:norows
    line=correl(r,1:(norows-r));
    correlrow=[correlrow line];
 end

Xtrain=[Xtrain; correlrow];
Ytrain=[Ytrain; get(k,'cough',i)];
```

1.1.3 HOSA

```
Fs=44100;
sizey=size(y,1);
x=(y-mean(y))/std(y);
figno(1) = gcf;
subplot(211), plot((1:sizey)*1000/Fs, x), grid on
xlabel('time in ms')
title('cough() data')
subplot(212), hist(x,20),
info=[];
 fprintf('\n Data and histogram plotted in figure window\n');
 disp(' The marginal distribution does not appear to be symmetric')
 fprintf('\n ------  Summary stats \n');
 fprintf(' Mean                %g\n',mean(y));
 fprintf(' Variance             %g\n',std(y).^2 );
 fprintf(' Skewness (normalized)   %g\n', mean(x.^3) );
 fprintf(' Kurtosis (normalized)   %g\n\n', mean(x.^4) - 3);

 avg=mean(y);
 var=std(y).^2;
 skew=mean(x.^3);
 kurt= mean(x.^4) - 3;
 info=[avg; var;skew;kurt];
 figure
    figno(length(figno)+1) = gcf;
 [spx, f,t] = specgram (x, 512, Fs, hamming(256), 240);
 contour(t*1000,f,abs(spx),8), grid on
 xlabel('time in ms'), ylabel('frequency in Hz'),
 set(gcf,'Name','speech spectrogram')
 set(gca,'ylim',[0 6000]);

 figure, [pxx,a2_1,a2_2] = harmest(x,25,0,'biased',512,2);
    figno(length(figno)+1) = gcf;
    set(gcf,'Name','speech: power spectra')

    figure, [pxx,a4_1,a4_2] = harmest(x,25,0,'biased',512,4);
    figno(length(figno)+1) = gcf;
    set(gcf,'Name','speech: cumulant spectra')

    r2_1=roots(a2_1);
    r2_2=roots(a2_2);
    r4_1=roots(a4_1);
    r4_2=roots(a4_2);

    [sg, sl]=glstat(x,.51,256);

    info=[info; sg; sl];
    [Bs,w]=bispecd( x, 256, 0, 100, 0);
```

```
    figure,
    contour(w,w,abs(Bs),4), grid on
title('Bispectrum estimated via the direct (FFT) method')
xlabel('f1'), ylabel('f2')
set(gcf,'Name','sound: bispectrum')
figno(length(figno)+1) = gcf;

figure,
d=diag(Bs);
plot(w,abs(d)), grid on
figno(length(figno)+1) = gcf;
set(gcf,'Name','sound: diagonal slice of bispectrum')

[loc,val]=pickpeak(d,3,5);
disp( w(loc)' )

figure,
ar=qpctor(x,25,10,256,100,0,'biased');
set(gcf,'Name','sound: QPC detection')
figno(length(figno)+1) = gcf;

figure
kfft = 2048;
Xf = fft(x,kfft) ;
Xf = Xf(1:kfft/2);
if (exist('lf') ~= 1) lf = 5; end
for k=1:4
 L=fix(kfft/(2*k));
 S = Xf(1:L).^k  .* conj( Xf(1:k:k*L) ) ;
 S = abs ( filter(ones(lf,1), 1, S) );
 eval(['S' int2str(k) ' = S;']);
 eval(['subplot(22', int2str(k), ')'])
 w = [1:(L-1)] / kfft * Fs;
 semilogy(w, S(2:L) ),    %avoid DC
 a= axis;  axis([0, round(40/k)*100, a(3), a(4) ]) ;
 ylabel(int2str(k))
 txt1 = [ 'CX(f) .* X(' int2str(k) 'f)' ];
 eval(['title (txt1)'])
 grid on
end
figno(length(figno)+1) = gcf;
```

1.2 Neural Networks

1.2.1 Creation and training of neural networks

1.2.1.1 Learning vector quantisation network, 'lvqnet'

```
function [varargout] = lvqnet(fname)
% Network train function
% Developed by Samantha Barry
global COUGH_

[pname,fname]=fileparts(fname);
if isempty(pname)
  pname=COUGH_.path;
end
```

```
fname=fullfile(pname,fname);

fnamephn=(fname(1:end-4));
extension='.mat';
data='_phn';
fnamephn=[fname, data, extension];
if exist (fnamephn)
   load(fnamephn);
   disp('phn loaded')
end

if exist(fsname([fname '_XY.mat']),'file')
   load(fsname([fname '_XY.mat']));
   disp('XY loaded')
end

if exist(fsname([fname '_peaks.mat']),'file')
   load(fsname([fname '_peaks.mat']));
   disp('peaks loaded')
end

[Xcalib,Ycalib,Xvalid,Yvalid,calibindex,validindex]=dataprep(fname);

% create a two-column matrix to describe coughs and non-coughs
nrows=size(Ycalib,1);
 matrix=zeros(nrows,2);
 for i=1:nrows
    if Ycalib(i,:)==1  %cough
       matrix(i,1)=1;
    elseif Ycalib(i,:)==0  %non-cough
       matrix(i,2)=1;
    end
  end
   Ycalibmat=matrix;
   %column 1 = cough, column 2 - non-cough

extension='.mat';
data='_info';
fnameinfo=[fname, data, extension];

%Set to 0 if creating new network or 1 to re-train existing
cont=1;

  disp('Loading Linear Vector Quantisation neural network...')
 % if network doesn`t exist, create

    if cont==0
         save (fnameinfo,'Xcalib','Ycalib','Xvalid','Yvalid','Ycalibmat');
       clear net
    fprintf('No network\n Creating new neural network\n')
    net=network;
    net=newlvq(minmax(Xcalib'),6,[.5 .5]);
    net.trainFcn='trainr';

    save coughnet net;
    cont=1;
    run=0;
    save fnameinfo run
    save (fnameinfo,'cont','run','Xcalib','Ycalib','Xvalid','Yvalid','Ycalibmat');
    [Y,E,perf]=sim(net,Xcalib');
```

```
  else disp('Training existing neural network')
     while cont==1
load fnameinfo run;
       run=[run+1];
        fprintf(['Run %d of network training\n'],run)

        save fnameinfo run

        load('coughnet');
        %[net,tr,Yout,Er]=adapt(net,Xcalib',Ycalib',[],[]);
        net.trainFcn='trainr';
        net.trainParam.epochs=100;

        [net,tr,Y,err]=train(net,Xcalib',Ycalibmat');

        err=mse(err);
        fprintf(['Training error: %3.4f\n'],err)

   % validate network
  disp(['Validating network'])
  [Ypredic]=sim(net,Xvalid');
  pred=Ypredic';
  % condense the event information back down to one column of zeros for non-coughs and
  % ones for coughs

     %column 1 = cough, column 2 - non-cough
     nrows=size(pred,1);
       Ypred=zeros(nrows,1);
         for j=1:nrows
             if pred(j,1)==1  %cough
              Ypred(j,1)=1;
             elseif pred(j,2)==1  %non-cough
              Ypred(j,1)=0;
             end
         end

  %compile indexed list of predictions and actual coughs to compare
sizeval=size(Yvalid,1);
result=[Yvalid Ypred];
correct=zeros(sizeval,1);
for i=1:sizeval
if result(i,1)==result(i,2)
correct(i,1)=1;
else
end
end
numcorr=find(correct);
num_correct=size(numcorr,1);
num=find(Yvalid);
totalcough=size(num,1);
totalnoncough=sizeval-totalcough;

  %compare actual coughs with predicted coughs to get a COUGH validation
  %accuracy
  truecough=[];
  for i=1:(size(result,1))
  true=find(result(i,1)==1)&(result(i,1)==result(i,2));
  if true==1
  truecough=[truecough; true];
```

```
  end
  end
  cough_correct=size(truecough,1);
  totalall=nrows;

  %compare actual non-coughs with predicted non-coughs to get a NONCOUGH validation
  %accuracy
  truenoncough=[];
  for i=1:(size(result,1))
  truenon=find(result(i,1)==0)&(result(i,1)==result(i,2));
  if truenon==1
  truenoncough=[truenoncough; truenon];
  end
  end
  noncough_correct=size(truenoncough,1);

  fprintf('Cough validation accuracy: %d out of %d classified correctly\n',
cough_correct,totalcough)
  perc_correct=(100/totalcough)*cough_correct;
  perc_incorrectc=(100/totalcough)*(totalcough-cough_correct);
  fprintf('%2.2f %% classified correctly\n',perc_correct)
  load fnameinfo
  save(fnameinfo,'run','cont','Xcalib','Ycalib','Xvalid','Yvalid','Ycalibmat');

fprintf('Non-Cough validation accuracy: %d out of %d classified correctly\n',
noncough_correct,totalnoncough)
  perc_correct=(100/totalnoncough)*noncough_correct;
  perc_incorrect=(100/totalnoncough)*(totalnoncough-noncough_correct);
  fprintf('%2.2f %% classified correctly\n',perc_correct)

  fprintf('Validation accuracy: %d out of %d classified correctly\n', num_correct, totalall)
  perc_correct=(100/totalall)*num_correct;
  perc_incorrect=(100/totalall)*(totalall-num_correct);
  fprintf('%2.2f %% classified correctly\n',perc_correct)

  if perc_correct==100
    disp('Validation error maximum reached... training stopped and network saved...')
    cont=[];
    save(fnameinfo,'run','cont','Xcalib','Ycalib','Xvalid','Yvalid','Ycalibmat');
    save coughnet net
    break
end
  % compare current validation error to previous error
  if ~exist('error','var')
     error=perc_incorrectc;
     cont=1;
     save(fnameinfo,'run','cont','error','Xcalib','Ycalib','Xvalid','Yvalid','Ycalibmat');
     save coughnet net
elseif perc_incorrectc<error
    error=perc_incorrectc;
    save coughnet net
  disp('Validation error smaller than previous run, training continues...')
  cont=1;
  save(fnameinfo,'error','run','cont','Xcalib','Ycalib','Xvalid','Yvalid','Ycalibmat');
  save coughnet net
elseif perc_incorrectc==error
    error=perc_incorrectc;
    save coughnet net
  comm = input('Validation error equal to previous run, press "y" for training to continue or "n" to
stop...\n','s');
```

```
 if comm == 'y'
    disp('Training continues...')
 cont=1;
 save(fnameinfo,'error','run','cont','Xcalib','Ycalib','Xvalid','Yvalid','Ycalibmat');
 save coughnet net
 elseif comm == 'n'
    disp('Network saved, training stopped...')
 cont=[];
 save(fnameinfo,'error','run','cont','Xcalib','Ycalib','Xvalid','Yvalid','Ycalibmat');
 end
elseif perc_incorrectc>error
   %do not save...revert back to last saved network
   disp('Validation error increased... training stopped and previous network saved...')
   cont=[];
    save(fnameinfo,'error','run','cont','Xcalib','Ycalib','Xvalid','Yvalid','Ycalibmat');

end
end
end
 disp('neural network training complete')
```

1.2.1.2 Feedforward neural network, 'ffnet'

```
function [varargout] = ffnet(fname)
% Network train function
% Developed by Samantha Barry
global COUGH_

[pname,fname]=fileparts(fname);
if isempty(pname)
 pname=COUGH_.path;
end
fname=fullfile(pname,fname);

fnamephn=(fname(1:end-4));
extension='.mat';
data='_phn';
fnamephn=[fname, data, extension];
if exist (fnamephn)
   load(fnamephn);
   disp('phn loaded')
end

if exist(fsname([fname '_XY.mat']),'file')
   load(fsname([fname '_XY.mat']));
   disp('XY loaded')
end

if exist(fsname([fname '_peaks.mat']),'file')
   load(fsname([fname '_peaks.mat']));
   disp('peaks loaded')
end

[Xcalib,Ycalib,Xvalid,Yvalid,calibindex,validindex]=dataprep(fname);

% create a two-column matrix to describe coughs and non-coughs
nrows=size(Ycalib,1);
 matrix=zeros(nrows,2);
```

```
  for i=1:nrows
     if Ycalib(i,:)==1  %cough
        matrix(i,1)=1;
     elseif Ycalib(i,:)==0  %non-cough
        matrix(i,2)=1;
     end
   end
    Ycalibmat=matrix;
    %column 1 = cough, column 2 - non-cough

    % change 0 (non-cough) to values of -1
    for n=1:nrows
       if Ycalib(n,1)==0
          Ycalib(n,1)=-1;
       end
    end
extension='.mat';
data='_info';
fnameinfo=[fname, data, extension];

%Set to 0 if creating new network or 1 to re-train existing
cont=1;

disp('Loading Feed Forward neural network...')
if cont==0
     save (fnameinfo,'Xcalib','Ycalib','Xvalid','Yvalid','Ycalibmat');
     clear net
     fprintf('No network\n Creating new neural network\n')

%create network
S1=6;
S2=4;
[R,Q]=size(Xcalib');
[S3,Q]=size(Ycalib');
P=Xcalib';
net=newff(minmax(P),[S1,S2,S3],{'tansig','logsig','tansig'},'traingd','learngdm');
%net=init(net);

save coughnet net
     run=0;
     save fnameinfo run
     save (fnameinfo,'cont','run','Xcalib','Ycalib','Xvalid','Yvalid','Ycalibmat');

else disp('Training existing neural network')
     while cont==1
       load fnameinfo run;
       run=[run+1];
       fprintf(['Run %d of network training\n'],run)

       save fnameinfo run

       load('coughnet');

%train network without noise
S1=6;
S2=4;
[R,Q]=size(Xcalib');
[S3,Q]=size(Ycalib');
P=Xcalib';
T=Ycalib';
```

```
net.performFcn='msereg';
net.performParam.ratio=0.5;
net.trainParam.goal=1e-5;
net.trainParam.show=100;
net.trainParam.epochs=1000;
net.trainParam.mc=0.95;
%learnParam.lr=0.05;

[net,tr,outs,e]=train(net,P,T);

  err=mse(e);
  fprintf(['Training error: %3.4f\n'],err)

%simulate network
disp(['Validating network'])

result=zeros((size(Xvalid,1)),1);
for s=1:(size(Xvalid,1))
   test=Xvalid(s,:);
   out=sim(net,test');
   if (1-out)>1
      answer=-1;
   else
      answer=1;
   end

   if answer==1
      result(s,1)=1;
   else
      result(s,1)=0;
   end
end

% condense the event information back down to one column of zeros for non-coughs and
% ones for coughs

%column 1 = cough, column 2 - non-cough
%nrows=size(result,1);
       %Ypred=zeros(nrows,1);
         %for j=1:nrows
         %    if result(j,1)==1  %cough
         %     Ypred(j,1)=1;
         %    elseif result(j,1)==0  %non-cough
         %     Ypred(j,1)=0;
         %    end
         % end
 Ypred=result;
%compile indexed list of predictions and actual coughs to compare
sizeval=size(Yvalid,1);
output=[Yvalid Ypred];
correct=zeros(sizeval,1);
for i=1:sizeval
if output(i,1)==output(i,2)
correct(i,1)=1;
else
end
end
numcorr=find(correct);
num_correct=size(numcorr,1);
num=find(Yvalid);
```

```
totalcough=size(num,1);
totalnoncough=sizeval-totalcough;

%compare actual coughs with predicted coughs to get a COUGH validation
 %accuracy
 truecough=[];
 for i=1:(size(output,1))
 true=find(output(i,1)==1)&(output(i,1)==output(i,2));
 if true==1
 truecough=[truecough; true];
 end
 end
 cough_correct=size(truecough,1);
 totalall=nrows;

 %compare actual non-coughs with predicted non-coughs to get a COUGH validation
 %accuracy
 truenoncough=[];
 for i=1:(size(output,1))
 truenon=find(output(i,1)==0)&(output(i,1)==output(i,2));
 if truenon==1
 truenoncough=[truenoncough; truenon];
 end
 end
 noncough_correct=size(truenoncough,1);

 fprintf('Cough validation accuracy: %d out of %d classified correctly\n',
cough_correct,totalcough)
 perc_correct=(100/totalcough)*cough_correct;
 perc_incorrectc=(100/totalcough)*(totalcough-cough_correct);
 fprintf('%2.2f %% classified correctly\n',perc_correct)
 load fnameinfo
 save(fnameinfo,'run','cont','Xcalib','Ycalib','Xvalid','Yvalid','Ycalibmat');

fprintf('Non-Cough validation accuracy: %d out of %d classified correctly\n',
noncough_correct,totalnoncough)
 perc_correct=(100/totalnoncough)*noncough_correct;
 perc_incorrect=(100/totalnoncough)*(totalnoncough-noncough_correct);
 fprintf('%2.2f %% classified correctly\n',perc_correct)

 fprintf('Validation accuracy: %d out of %d classified correctly\n', num_correct, totalall)
 perc_correct=(100/totalall)*num_correct;
 perc_incorrect=(100/totalall)*(totalall-num_correct);
 fprintf('%2.2f %% classified correctly\n',perc_correct)

 if perc_correct==100
  disp('Validation error maximum reached... training stopped and network saved...')
  cont=[];
  save(fnameinfo,'run','cont','Xcalib','Ycalib','Xvalid','Yvalid','Ycalibmat');
  save coughnet net
  break
end
 % compare current validation error to previous error
 if ~exist('error','var')
   error=perc_incorrectc;
   cont=1;
   save(fnameinfo,'run','cont','error','Xcalib','Ycalib','Xvalid','Yvalid','Ycalibmat');
   save coughnet net
elseif perc_incorrectc<error
  error=perc_incorrectc;
```

```
   save coughnet net
  disp('Validation error smaller than previous run, training continues...')
  cont=1;
  save(fnameinfo,'error','run','cont','Xcalib','Ycalib','Xvalid','Yvalid','Ycalibmat');
  save coughnet net
  %elseif perc_incorrectc==error
    %error=perc_incorrectc;
    %save coughnet net
 % comm = input('Validation error equal to previous run, press "y" for training to continue or "n"
to stop...\n','s');
 % if comm == 'y'
 %     disp('Training continues...')
 % cont=1;
 % save(fnameinfo,'error','run','cont','Xcalib','Ycalib','Xvalid','Yvalid','Ycalibmat');
 % save coughnet net
 % elseif comm == 'n'
%      disp('Network saved, training stopped...')
%  cont=[];
%  save(fnameinfo,'error','run','cont','Xcalib','Ycalib','Xvalid','Yvalid','Ycalibmat');
%  end
elseif perc_incorrect>error
    %do not save...revert back to last saved network
    disp('Validation error increased... training stopped and previous network saved...')
    cont=[];
     save(fnameinfo,'error','run','cont','Xcalib','Ycalib','Xvalid','Yvalid','Ycalibmat');

end
end
end
  disp('neural network training complete')
```

1.2.1.3 Probabilistic neural network, 'pnnet'

```
function [varargout] = pnnet(fname)
% Network train function
% Developed by Samantha Barry
global COUGH_


[pname,fname]=fileparts(fname);
if isempty(pname)
  pname=COUGH_.path;
end
fname=fullfile(pname,fname);

fnamephn=(fname(1:end-4));
extension='.mat';
data='_phn';
fnamephn=[fname, data, extension];
if exist (fnamephn)
    load(fnamephn);
    disp('phn loaded')
end

if exist(fsname([fname '_XY.mat']),'file')
    load(fsname([fname '_XY.mat']));
    disp('XY loaded')
end
```

```
if exist(fsname([fname '_peaks.mat']),'file')
   load(fsname([fname '_peaks.mat']));
   disp('peaks loaded')
end

[Xcalib,Ycalib,Xvalid,Yvalid,Xall,Yall,calibindex,validindex,allindex]=dataprep(fname);

%Xcalib(:,16:42)=[];
% create a vector of ones and twos describe coughs and non-coughs
% respectively
for i=1:size(Ycalib,1)
if Ycalib(i,1)==1
Ycalib(i,1)=2;
elseif Ycalib(i,1)==0
Ycalib(i,1)=1;
end
end

Ycalibmat=ind2vec(Ycalib);

extension='.mat';
data='_info';
fnameinfo=[fname, data, extension];

  disp('Loading probabilistic neural network...')
 % if network doesn`t exist, create

       save (fnameinfo,'Xcalib','Ycalib','Xvalid','Yvalid','Ycalibmat');
       clear net
    fprintf('No network\n Creating new neural network\n')
    net=network;
    net=newpnn(Xcalib',Ycalibmat,0.1);

    save coughnet net;
    cont=1;
    run=0;
    save fnameinfo run
    save (fnameinfo,'cont','run','Xcalib','Ycalib','Xvalid','Yvalid','Ycalibmat');
    [Y,E,perf]=sim(net,Xcalib');
    %plot(Xcalib',Ycalib','b-',Xcalib',Y,'r-+'),title('Initial outputs versus targets');

disp('Training existing neural network')
    while cont==1
load fnameinfo run;
       run=[run+1];
        fprintf(['Run %d of network training\n'],run)

%save(fnameinfo,'run','cont','Xcalib','Ycalib','Xvalid','Yvalid','Ycalibmat','traindex','valindex');
        save fnameinfo run

        load('coughnet');
        %[net,tr,Yout,Er]=adapt(net,Xcalib',Ycalib',[],[]);

        net.trainParam.epochs=100;

        [net,tr,Y,err]=train(net,Xcalib',Ycalibmat');
        %save coughnet net

        err=mse(err);
```

```
    fprintf(['Training error: %3.4f\n'],err)

% validate network
disp(['Validating network'])
[Ypredic]=sim(net,Xvalid');
pred=Ypredic';
% condense the event information back down to one column of zeros for non-coughs and
% ones for coughs

  %column 1 = cough, column 2 - non-cough
  nrows=size(pred,1);
    Ypred=zeros(nrows,1);
      for j=1:nrows
          if pred(j,1)==1  %cough
           Ypred(j,1)=1;
          elseif pred(j,2)==1  %non-cough
           Ypred(j,1)=0;
          end
      end

 %compile indexed list of predictions and actual coughs to compare
sizeval=size(Yvalid,1);
result=[Yvalid Ypred];
correct=zeros(sizeval,1);
for i=1:sizeval
if result(i,1)==result(i,2)
correct(i,1)=1;
else
end
end
numcorr=find(correct);
num_correct=size(numcorr,1);
num=find(Yvalid);
totalcough=size(num,1);
totalnoncough=sizeval-totalcough;


%compare actual coughs with predicted coughs to get a COUGH validation
%accuracy
truecough=[];
for i=1:(size(result,1))
true=find(result(i,1)==1)&(result(i,1)==result(i,2));
if true==1
truecough=[truecough; true];
end
end
cough_correct=size(truecough,1);
totalall=nrows;

%compare actual non-coughs with predicted non-coughs to get a COUGH validation
%accuracy
truenoncough=[];
for i=1:(size(result,1))
truenon=find(result(i,1)==0)&(result(i,1)==result(i,2));
if truenon==1
truenoncough=[truenoncough; truenon];
end
end
noncough_correct=size(truenoncough,1);
```

```
  fprintf('Cough validation accuracy: %d out of %d classified correctly\n',
cough_correct,totalcough)
  perc_correct=(100/totalcough)*cough_correct;
  perc_incorrect=(100/totalcough)*(totalcough-cough_correct);
  fprintf('%2.2f %% classified correctly\n',perc_correct)
  load fnameinfo
  save(fnameinfo,'run','cont','Xcalib','Ycalib','Xvalid','Yvalid','Ycalibmat');

fprintf('Non-Cough validation accuracy: %d out of %d classified correctly\n',
noncough_correct,totalnoncough)
  perc_correct=(100/totalnoncough)*noncough_correct;
  perc_incorrect=(100/totalnoncough)*(totalnoncough-noncough_correct);
  fprintf('%2.2f %% classified correctly\n',perc_correct)

  fprintf('Validation accuracy: %d out of %d classified correctly\n', num_correct, totalall)
  perc_correct=(100/totalall)*num_correct;
  perc_incorrect=(100/totalall)*(totalall-num_correct);
  fprintf('%2.2f %% classified correctly\n',perc_correct)

  if perc_correct==100
    disp('Validation error maximum reached... training stopped and network saved...')
    cont=[];
    save(fnameinfo,'run','cont','Xcalib','Ycalib','Xvalid','Yvalid','Ycalibmat');
    save coughnet net
    break
end
  % compare current validation error to previous error
  if ~exist('error','var')
     error=perc_incorrect;
     cont=1;
     save(fnameinfo,'run','cont','error','Xcalib','Ycalib','Xvalid','Yvalid','Ycalibmat');
     save coughnet net
elseif perc_incorrect<error
    error=perc_incorrect;
    save coughnet net
  disp('Validation error smaller than previous run, training continues...')
  cont=1;
  save(fnameinfo,'error','run','cont','Xcalib','Ycalib','Xvalid','Yvalid','Ycalibmat');
  save coughnet net
elseif perc_incorrect==error
    error=perc_incorrect;
    save coughnet net
  comm = input('Validation error equal to previous run, press "y" for training to continue or "n" to
stop...\n','s');
  if comm == 'y'
      disp('Training continues...')
  cont=1;
  save(fnameinfo,'error','run','cont','Xcalib','Ycalib','Xvalid','Yvalid','Ycalibmat');
  save coughnet net
  elseif comm == 'n'
      disp('Network saved, training stopped...')
  cont=[];
  save(fnameinfo,'error','run','cont','Xcalib','Ycalib','Xvalid','Yvalid','Ycalibmat');
  end
elseif perc_incorrect>error
    %do not save...revert back to last saved network
    disp('Validation error increased... training stopped and previous network saved...')
    cont=[];
     save(fnameinfo,'error','run','cont','Xcalib','Ycalib','Xvalid','Yvalid','Ycalibmat');
```

```
end
end
 disp('neural network training complete')
```

1.2.2 Implementation of trained neural network

1.2.2.1 'Netprocess'

```
function varargout = netprocess(varargin)
% Initiation of neural network processing
% Created by Samantha Barry

global COUGH_
cdefault;

nargs=length(varargin);
if (nargs<1)
cd(COUGH_.path);
      [filename,pathname]=uigetfile({'*.wav','All WAVE-Files (*.wav)'},...
                                    'Select WAVE File');
 % If "Cancel" is selected then return
 if isequal([filename,pathname],[0,0])
  return
 else
%   cd(pathname);
  filename=fullfile(pathname,filename);
end
elseif (nargs==1)
  filename=fullfile(COUGH_.path,varargin{1});
end

  fname=filename;
  COUGH_.feat.options.nclasses=2;
  reduce=1;
  lastchunk=0;
  overcount=0;
  save(fsname([fname '_fc']),'lastchunk','overcount');

  %load(fname)
  for mn=1:10 % max number of files since 8 portions of 3 hours in a 24 hour recording
  str=num2str(mn);
  file=(fname);
  extension='.wav';
  under='_';
  filename=[file, under, str];
  more=[];
  if exist(fsname([filename]))
     next=mn+1;
     next=num2str(next);
     nextfile=[file, under, next];
     stop=0;
     if exist('nextfile','file')
        more=1;
     else
        more=0;
     end
  elseif exist(fsname([file]))
     stop=0;
```

```
      more=0;
    else
      stop=1;
    end
    if stop~=1
     k=cough(filename);
     fnamexy=[fsname(file) '_XY'];
     save(fsname([filename '_fc']),'more','lastchunk','overcount');


    [x,y,p]=features(k,filename,'halfheight');
    save(fsname([fname '_peaks']),'p');
    save(fsname([fname '_XY.mat']),'x','y');
    else
    return
    end
end
```

1.2.2.2 'Features'

```
function [X,Y,p] = features(k,fname,std_tresh,whichones,options)
% FEATURES
global COUGH_

[pname,fname]=fileparts(fname);
if isempty(pname)
  pname=COUGH_.path;
end
fname=fullfile(pname,fname);

load(fsname([fname '_fc']));
load(fsname([fname '_fc']),'more','lastchunk','overcount');

options={'indices' i};
o=getoptions(options);

if nargin<4 | isempty(whichones)
  %whichones=COUGH_.feat.whichones;
    whichones={'melcepst'};
end

if nargin<3
  std_tresh=COUGH_.features;
end

if isa(whichones,'char')
  %whichones={whichones};
    whichones={'melcepst'};
end

p=k.p;
%% Addition to split up the wave file into manageable chunks  05/Oct/05



totnumber=[];
mins=15;
chops=mins*11025*60; % convert ms to indices
```

```
pends=[];
pstarts=[];
plengths=[];
max=size(p,2);
lastend=1;
saveend=0;
for w=1:max
   starts=p(w).start;
   pstarts=[pstarts; starts];
   ends=p(w).end;
   pends=[pends; ends];
   length=p(w).length;
   plengths=[plengths; length];
end

a=auin(fname);

if chops>(p(end).end);
   filechops=[1 a.length];
else filechops=[1:chops:a.length a.length];
end

filechops=filechops';
numch=size(filechops,1);

% correct for lastchunk overlap by making first chunk smaller
if lastchunk~=0
correct=ones(size(filechops,1),1);
correct=(correct.*lastchunk);
corrected=filechops(2:end)-correct(2:end);
filechops=[1; corrected];
marker=1;
else marker=0;
end

alength=a.length;
save(fsname([fname '_fc']),'alength','more','lastchunk','overcount','plengths');



lastpeak=0;
%create array of chop information
if numch>1
   for ch=2:numch
      [peak,no,indvalue]=findnearest(filechops(ch),pstarts,-1);

   fc(ch-1).firststart=p(lastpeak+1).start;
   fc(ch-1).firstend=p(lastpeak+1).end;
   fc(ch-1).laststart=p(peak).start;
   fc(ch-1).lastend=p(peak).end;

   fc(ch-1).peak=(peak);

   lastpeak=peak;
   end


elseif numch==1
   fc(1).start=p(1).start;
   fc(1).end=p(end).end;
```

```
   fc(1).peak=size(pstarts,1);
   ch=ch+1;
end

endchunk(1)=0;
for z=1:(size(fc,2));
   endchunk(z+1)=fc(z).peak;
end

  for j=1:(endchunk(end))
     p(j).nnpred=0;
     p(j).fraction=0;
  end

nfeatures=length(whichones);

X=[];
Y=[];

Ypred=k;

extension='.mat';
data='_pca';
fnamepca=[fname, data, extension];
 load fnamepca newload
indices=[1:get(k,'npeaks')];
nind=length(indices);

d=1;
while d<size(endchunk,2)
for i=(endchunk(d)+1):endchunk(d+1)

[X,Y]=features(Ypred,[],[],{'indices' i});

Xpca=X*newload;
load ('coughnet');
    [pred]=sim(net,Xpca');
    pred=pred';
    %column 1 = cough, column 2 - non-cough
    nrows=size(pred,1);
      Ypredict=zeros(nrows,1);
        for m=1:nrows
            if pred(m,1)==1  %cough
             Ypredict(m,1)=1;
            elseif pred(m,2)==1  %non-cough
             Ypredict(m,1)=0;
            end
         end

    p(i).fraction=mean(Ypredict);
    p(i).nnpred=p(i).fraction>0.5;

    np=size(p(i).p,1);

  for j=1:np,
   %p(i).p(j,4)=mean(Ypredict(p(i).p(j,1):p(i).p(j,2)));
   p(i).p(j,4)=p(i).fraction;
   p(i).p(j,6)=p(i).p(j,2)-p(i).p(j,1)+1;
  end
  if p(i).p(:,4)>0.5;
```

```
    p(i).p(:,5)=1;
  else p(i).p(:,5)=0;
  end

    if p(i).nnpred==1
    p(i).cough=1;
    else p(i).cough=0;
    end

Ypred=set(Ypred,'p',p);
save([fsname(get(Ypred,'file')) '_peaks'],'p');

% check coughs over 1 sec long are not double coughs
a=auin(fname);

cs=zeros(1,(size(p,2)));
        for d=1:size(p,2)
        cs(1,d)=p(d).cough;
        end
     coughs=find(cs);
     cp=p(coughs);

     coughtime=ones(size(cp,2),1);
       for h=1:size(cp,2)
       coughtime(h)=cp(h).length;
       end

    long=find(coughtime>11025);

    if isempty(long)
      %continue
    else lstart=p(long).start;
        lend=p(long).end;
        sound(k,lend-lstart,lstart,'{}');


%save(fsname([fname '_peaks']),'p');
%save(fname,'p');


end
ny=zeros(1,(size(p,2)));
for r=lastend:endchunk(d+1)
  ny(1,r)=p(r).cough;
  lastend=endchunk(d+1);
end

  number=find(ny);
  numberc=size(number,2);
  totnumber=[totnumber; numberc];

% add on any overcount to first, smaller chunk
if marker==0
  fc(d).count=size(number,2);
elseif marker==1
  fc(d).count=size(number,2)+overcount;
end
marker=0; %revert to normal after first run
```

```
 fprintf('Number of coughs identified: %d \n',numberc)

 d=d+1;
end

save(fsname([fname '_fc']),'fc','alength','lastchunk','overcount','more');

sumcough=sum(totnumber);
fprintf('Total number of coughs identified: %d \n',sumcough)



load(fsname([fname '_fc']),'more');
if more==1
lastchunk=(fc(end).end-fc(end).start);
  if lastchunk<chops
   incomp=1;  % incomplete portion
   overcount=fc(end).count;
   save(fsname([fname '_fc']),'fc','alength','lastchunk','overcount');
   fc=fc(1:(end-1));
end
else
   fc=fc;
end
end

function options = getoptions(opts)
% GETOPTIONS -
global COUGH_

options=COUGH_.feat.options;

for i=1:2:length(opts),
 options=setfield(options,opts{i},opts{i+1});
end
```

1.3 FFT Implementation

1.3.1 File preparation, 'fft_process'

```
function varargout = fft_process(varargin)
% Initiation of fft processing
% Created by Samantha Barry

global COUGH_
cdefault;

nargs=length(varargin);
if (nargs<1)
cd(COUGH_.path);
      [filename,pathname]=uigetfile({'*.wav','All WAVE-Files (*.wav)'},...
                                    'Select WAVE File');
 % If "Cancel" is selected then return
 if isequal([filename,pathname],[0,0])
  return
 else
```

```
%    cd(pathname);
    filename=fullfile(pathname,filename);
end
elseif (nargs==1)
    filename=fullfile(COUGH_.path,varargin{1});
end

path='C:\MATLAB6p5\Adrie\sound_new\44100\';
    file=filename(end-5:end);
    fname=[path,file];

    reduce=1;
    lastchunk=0;
    overcount=0;
    save(fsname([fname '_fc']),'lastchunk','overcount');

    %load(fname)
    for mn=1:8 % max number of files since 8 portions of 3 hours in a 24 hour recording
    str=num2str(mn);
    extension='.wav';
    under='_part';
    peaks='_peaks.mat';
    filename=[fname, under, str, peaks];
    filewav=[fname,under,str,extension];
    fileshort=[fname,under,str];
    more=[];
    if exist([filename])
     coughwav=filewav(end-15:end-4);
     k=cough(coughwav);
     fnamexy=[fsname(file) '_XY'];
     save(fsname([filename '_fc']),'more','lastchunk','overcount');

    fprintf('Section %d\n',mn)
    [x,y,p]=fft_features(k,fileshort);
    %save(fsname([fname '_peaks']),'p');
    save(fsname([fname '_XY.mat']),'x','y');
    else
    return
    end
end
```

1.3.2 Data preparation, 'fft_features'

```
function [X,Y,p] = fft_features(k,fname)
% Preparation for processing
% Created by Samantha Barry

global COUGH_

[pname,fname]=fileparts(fname);
if isempty(pname)
  pname=COUGH_.path;
end
fname=fullfile(pname,fname);

if exist(fsname([fname '_fc']))
load(fsname([fname '_fc']));
end
```

```
p=k.p;
%% Addition to split up the wave file into manageable chunks  05/Oct/05

totnumber=[];
mins=15;
chops=mins*11025*60; % convert ms to indices
pends=[];
pstarts=[];
plengths=[];
max=size(p,2);
lastend=1;
saveend=0;
for w=1:max
   starts=p(w).start;
   pstarts=[pstarts; starts];
   ends=p(w).end;
   pends=[pends; ends];
   length=p(w).length;
   plengths=[plengths; length];
end

a=auin(fname);

if chops>(p(end).end);
   filechops=[1 a.length];
else filechops=[1:chops:a.length a.length];
end

filechops=filechops';
numch=size(filechops,1);


alength=a.length;
save(fsname([fname '_fc']),'alength');


lastpeak=0;
%create array of chop information
if numch>1
   for ch=2:numch
      [peak,no,indvalue]=findnearest(filechops(ch),pends,-1);
      if isempty(peak)
         peak=0;
      end
   % check that in the last chunk, there are still peaks left to be
   % identified...
      if lastpeak==peak % no peaks in the section
      fc(ch-1).firststart=0;
   fc(ch-1).firstend=0;
   fc(ch-1).laststart=0;
   fc(ch-1).lastend=0;
   fc(ch-1).peak=0;
   fc(ch-1).count=0;
   fc(ch-1).length=0;
      else
   fc(ch-1).firststart=p(lastpeak+1).start;
   fc(ch-1).firstend=p(lastpeak+1).end;
   fc(ch-1).laststart=p(peak).start;
   fc(ch-1).lastend=p(peak).end;
```

```
    fc(ch-1).peak=(peak);
    fc(ch-1).count=(peak-lastpeak);

       lastpeak=peak;
    end
    end

elseif numch==1
    fc(1).start=p(1).start;
    fc(1).end=p(end).end;
    fc(1).peak=size(pstarts,1);
    ch=ch+1;
end

endchunk=zeros(size(fc,2),1);

for z=1:(size(fc,2));
    endchunk(z+1,1)=fc(z).peak;
end

X=[];
Y=[];
coughlength=[];

d=1;
while d<size(endchunk,1)
    inds=(endchunk(d)+1):endchunk(d+1);
    %inds={inds};

    %need peak indices in the current chunk to pass to coughid
    filt=fft_coughid(fname,[inds]);
    numcoughs=size(filt,2);

    if ~isempty(filt(1).ind)
    for m=1:numcoughs
       pos=filt(m).ind;
       p(pos).cough=1;
       clength=filt(m).peaksize;
       coughlength=[coughlength; clength];
    end
    end

  save([fsname(fname) '_peaks'],'p');


       if isempty(coughlength)
          fc(ch-1).length=0;
       else
       cumval=cumsum(coughlength);
       fc(d).length=cumval(end);
       end

ny=zeros(1,(size(p,2)));
for r=lastend:endchunk(d+1)
    ny(1,r)=p(r).cough;
    lastend=endchunk(d+1);
end

    number=find(ny);
    numberc=size(number,2);
```

```
   totnumber=[totnumber; numberc];

   fc(d).count=numberc;

  fprintf('Number of coughs identified: %d \n',numberc)

  d=d+1;
end

save(fsname([fname '_fc']),'fc','alength');

sumcough=sum(totnumber);
fprintf('Total number of coughs identified: %d \n',sumcough)

end

plotcough(fname);
```

1.3.3 Identification of cough, 'fft_coughid'

```
function c = fft_coughid(fname,inds)
% Function passes signal through adaptive filters to identify cough
% frequencies
% Created by Samantha Barry

global COUGH_

shortfile=fname(end-11:end);
shortpath=fname(1:end-12);
fileloc=[shortpath,'44100\',shortfile];

ext='.wav';
file=[fname ext];

file2=[fileloc '_peaks.mat'];
load (file2)

fs=44100;
minute=60*fs;
nq=fs/2;

c.start=[];
c.end=[];
c.sigsize=[];
c.cough=[];
c.peaksize=[];
c.ind=[];
signal=[];


% take the p values of only the indices given
num=size(inds,2);
for n=1:num
   current=inds(n);
   peak(n)=p(current);
end
p=peak;

siz = wavread(file,'size');
```

```
fs=44100;
minute=60*fs;
numpeak=size(p,2);
thresh=1e-4;
covmax=[];
loopsum=[];
eventcounter=1;

for event=1:numpeak;

first=p(event).start;
last=p(event).end;
plength=last-first;
[sig fs]=wavread(file,[first last]);

if plength>1500
[B,F,T]=specgram(sig(:,1),256,fs,hamming(256),128);
cutb=B(36:88,:);
absb=abs(cutb);
sqrd=absb*absb';

eventcov=cov(sqrd);
eventmax=max(eventcov,[],2);
finalmax=max(eventmax);
covmax=[covmax; finalmax];
clear eventcov

  if finalmax>thresh
   loopsum=[loopsum event];
   c(eventcounter).start=first;
   c(eventcounter).end=last;
   c(eventcounter).peaksize=size(sig,1);
   c(eventcounter).cough=1;
   c(eventcounter).ind=inds(event);
   eventcounter=eventcounter+1;
  else
  end

end
end

%events that are too small to be coughs
lengths=[];
    for i=1:size(c,2)
      lengths=[lengths; c(i).peaksize];
    end
    short=find(lengths<2000);
    for s=short
       c(s)=[];
    end
```

1.4 Plots

1.4.1 Plot of cough summary, 'plotcough'

```
function varargout = plotcough(fname)
% PLOTCOUGH
```

```
% Created by Samantha Barry
global COUGH_
global t y k numcough next

   %if exist(fsname([fname '.mat']),'file')
   load(fsname([fname '_fc']),'fc','alength');
   load(fsname([fname '_peaks']),'p');

        % taking just the cough events
        cs=zeros(1,(size(p,2)));
        for d=1:size(p,2)
        cs(1,d)=p(d).cough;
        end
        coughs=find(cs);
        cp=p(coughs);

        cpends=[];
        cplengths=[];
        max=size(cp,2);
  % compiling a vector of cough lengths
         for w=1:max
         length=cp(w).length;
         cplengths=[cplengths; length];
         end
         pends=[];

  % compiling a vector of cough ends
         for w=1:max
         ends=cp(w).end;
         cpends=[cpends; ends];
         end
  % finding long coughs
        long=[];
        for lp=1:size(p,2)
          if p(lp).cough==1 & p(lp).length>44100
            length=p(lp).length;
            detail=[lp length];
            long=[long; detail];
          end
        end
      numlong=size(long,1);
      extra=zeros(size(long,1),1);
      long=[long extra];
  % to plot a graph of cough numbers

  % calculating the 15min portions
  totlength=size(fc,2);
  totnumber=ones(totlength,1);
for t=1:totlength
   totnumber(t)=fc(t).count;
end
mins=15;
minslong=alength/44100/60;
numbint=minslong/mins;
whole=floor(numbint);
incomp=mod(minslong,mins);
min=[];
if whole~=0
   min=zeros(whole,1);
   for w=1:whole
```

```
      min(w,1)=w*15;
   end
   lastnum=min(end)+incomp;
else
   lastnum=incomp;
end

ind=mins*60*44100;
timeind=ones(whole+1,1);
for j=1:(size(timeind,1))-1
   timeind(j)=ind*j;
end
timeind(end)=alength;

% preparing the timeind for extra info
extra=zeros(size(timeind,1),3);
timeind=[timeind extra];
% timeind = 4 columns:- [15min interval time (indices) : peakend nearest to end time :
% length of cough time in each section (indices) : cough time (s)]

sizefc=size(fc,1);
sizetime=size(timeind,1);
  for g=1:sizetime
     timeind(g,2)=fc(g).lastend;
  end


for h=1:totlength
   length=fc(h).length;
   if ~isempty(length);
   timeind(h,3)=length;
   end
end

for n=1:size(timeind,1)
timeind(n,4)=timeind(n,3)/11025;
end
numbermin=ones(size(timeind,1),1);
numbermin=numbermin.*timeind(:,4);

intervals=[min; lastnum];
intervals=intervals/60;
tot=[];
tot2=[];

      % calculating the plot of number of coughs
      for t=1:size(intervals,1)
         m=totnumber(t);
         c=ones(m,1);
         intmins=intervals(t);
         mult=c.*intmins;
         tot=[tot; mult];
      end

      %calculating the plot of length of time coughing
      for t2=1:size(intervals,1)
         m2=ceil(numbermin(t2));
         c=ones(m2,1);
         intmins=intervals(t2);
         mult=c.*intmins;
```

```
        tot2=[tot2; mult];
    end

bin=(mins/2)/60;
bins=[bin:(mins/60):intervals(end)];

% plot number of coughs
figure, hist(tot,bins);
axis=([0 24 0 Inf]);
set(gca,'xtick',[0:2:24]);
title('Cough Frequency')
xlabel('Time (hours)')
ylabel('Number of Coughs')

% plot cough seconds
figure, hist(tot2,bins);
axis=([0 24 0 Inf]);
set(gca,'xtick',[0:2:24]);
title('Cough Frequency')
xlabel('Time (hours)')
ylabel('Coughing Time(seconds)')

save (fsname([fname '_plot']), 'tot','tot2', 'bins')

% solution to double cough problem

if isempty(long)
  %disp('No cough events longer than one second')
else
  fprintf('%d cough events longer than one second\n', numlong)
display=input('Do you wish to review the long cough events? (y/n)\n','s');
   if display=='y'
      review=1;
   else
      review=0;
   end

   if review==1

      next=1;
      numcough=1;
for r=1:size(long,1)

        pind=long(r,1);
        pstart=p(pind).start;
        pfinish=(p(pind).end);
        plength=(p(pind).length);

%%% up to this point, long coughs correctly identified.
% plotting the sound
wavname=[fname(1:end-18), fname(end-11:end), '.wav'];
au=auin([wavname]);
Fs=get(au,'fs');
file=au.filename;
[sig fs]=wavread(file,[pstart pfinish]);
h=figure,plot(sig(:,1))
soundsc(sig,44100);

choice=input('Is this a cough or non-cough? (c/n)\n (Or enter "p" to replay sound or "q" to
quit)','s');
```

```
if choice=='c'
   id=1;
elseif choice=='n'
   id=0;
   coughcount=0;
elseif choice=='p'
   soundsc(sig,44100);
   choice=input('Is this a cough or non-cough? (c/n)\n','s');
   if choice=='c'
   id=1;
   elseif choice=='n'
   id=0;
   coughcount=0;
   end
elseif choice=='q'
   return
end
   if id==1
      soundsc(sig,44100);
      howmany=input('How many coughs? \n','s');
      coughcount=str2num(howmany);
   end

close(h)
long(r,3)=coughcount;

end
for longp=1:size(long,1)
   index=long(longp,1);
   if long(longp,3)==0
      p(index).cough=0;
   else
   end
end
save(fsname([fname '_peaks']),'p');

 % locate the portion the long cough is in
  lastpeaks=ones(totlength,1);
 for f=1:totlength
 lastpeaks(f,1)=fc(f).peak;
 end
 lastpeaks=[0; lastpeaks];
 extra=zeros(size(lastpeaks,1),1);
 lastpeaks=[lastpeaks extra];

 newcount=0;
 for t=1:totlength
    hilimit=lastpeaks(t+1);
    lolimit=lastpeaks(t);
    locport=find((long(:,1)<=hilimit) & (long(:,1)>lolimit));
    for s=1:size(locport,1)
       g=locport(s,1);
       newcount=newcount+long(g,3);
    end
     lastpeaks(t,2)=newcount;
     newcount=0;
 end

 % add the new coughs to the existing count
 realend=size(lastpeaks,1)-1;
```

```
 modcount=totnumber(:,1)+lastpeaks(1:realend,2);

 % replot cough number graph
 for t=1:size(intervals,1)
        m=modcount(t);
        c=ones(m,1);
        intmins=intervals(t);
        mult=c.*intmins;
        tot=[tot; mult];
      end
 figure, hist(tot,bins);
axis=([0 24 0 Inf]);
set(gca,'xtick',[0:2:24]);
title('Cough Frequency')
xlabel('Time (hours)')
ylabel('Number of Coughs')


else
end
end
```

1.4.2 Combination of three-hour plots to make 24 hour summary, 'combineplot'

```
function [varargout] = combineplot(fname)
% Combine 24 hour plots function
% Developed by Samantha Barry
global COUGH_

[pname,fname]=fileparts(fname);
if isempty(pname)
  pathname=COUGH_.path;
  ext='\44100';
  pname=[pathname,ext];
end
fname=fullfile(pname,fname);
coughcount=[];
coughfreq=[];


core=fname(1:end-1); % remove the final number

for mn=1:8 % max number of files since 8 portions of 3 hours in a 24 hour recording
   str=num2str(mn);
   extension='.mat';
   suff='_plot';
   filename=[core, str, suff, extension];

   load (filename) % giving 'tot', 'tot2' and 'bins'
   %need to offset the values
    if mn>1
      offset=(mn-1)*3;
      off=ones(size(tot,1),1);
      off2=ones(size(tot2,1),1);
      offsetval=offset.*off;
      offsetval2=offset.*off2;
      tot=tot+offsetval;
```

```
        tot2=tot2+offsetval2;
    end
    coughcount=[coughcount;tot];
    coughfreq=[coughfreq;tot2];

end

mins=15;
minslong=1440;
numbint=minslong/mins;
min=zeros(numbint,1);

for w=1:numbint
      min(w,1)=w*15;
end

intervals=[min];
intervals=intervals/60;
bin=(mins/2)/60;
bins=[bin:(mins/60):intervals(end)];

% plot number of coughs
figure, hist(coughcount,bins);
axis=([0 24 0 Inf]);
set(gca,'xtick',[0:2:24]);
title('Cough Frequency')
xlabel('Time (hours)')
ylabel('Number of Coughs')

% plot cough seconds
figure, hist(coughfreq,bins);
axis=([0 24 0 Inf]);
set(gca,'xtick',[0:2:24]);
title('Cough Frequency')
xlabel('Time (hours)')
ylabel('Coughing Time(seconds)')
```

1.5 Patient activity monitoring

1.5.1 Event marker locater, 'signal'

```
function [varargout] = signal(fname)
% Signal detection
% Developed by Samantha Barry
global COUGH_

[pname,fname]=fileparts(fname);
if isempty(pname)
  pname=COUGH_.path;
end
fname=fullfile(pname,fname);

length=180; % minutes in 3 hour wave file
timeind=length*60*44100; % length of file in samples
minute=60*44100; % length of a minute in samples
siz = wavread(fname,'size');
```

```
sf=44100;
nq=sf/2;
band1=14300;
band2=14900;
width=50;
bottom=band1-width;
top=band2-width;
Wp=[band1 band2]/nq;
Ws=[14000 15200]/nq;
Rp=3;
Rs=40;
[n,Wn]=buttord(Wp,Ws,Rp,Rs);
[b,a]=butter(n,Wn);

last=0;
beep=[];
peakstart=[];
for m=1:length  % start the loop
 first=last+1;
 last=minute*m;
 if last > siz(1,1)
    last=siz(1,1);
 end
 [sig fs]=wavread(fname,[first last]);
 yf=abs(filter(b,a,sig));
thr=1e-3;
          eventcounter=1;
          pointer=1;
   if m==180
      resid=siz(1,1)-(179*minute);
      minute=resid;
   end
          while pointer < minute
                    if yf(pointer) < thr
                    pointer=pointer+1;
                    else
                    peakstart(eventcounter)=pointer+(minute*(m-1));

                    disp(pointer/fs);
                    pointer=pointer+fs*2;
                    eventcounter=eventcounter+1;
        end
     end

beep=[beep peakstart];
peakstart=[];
end
```

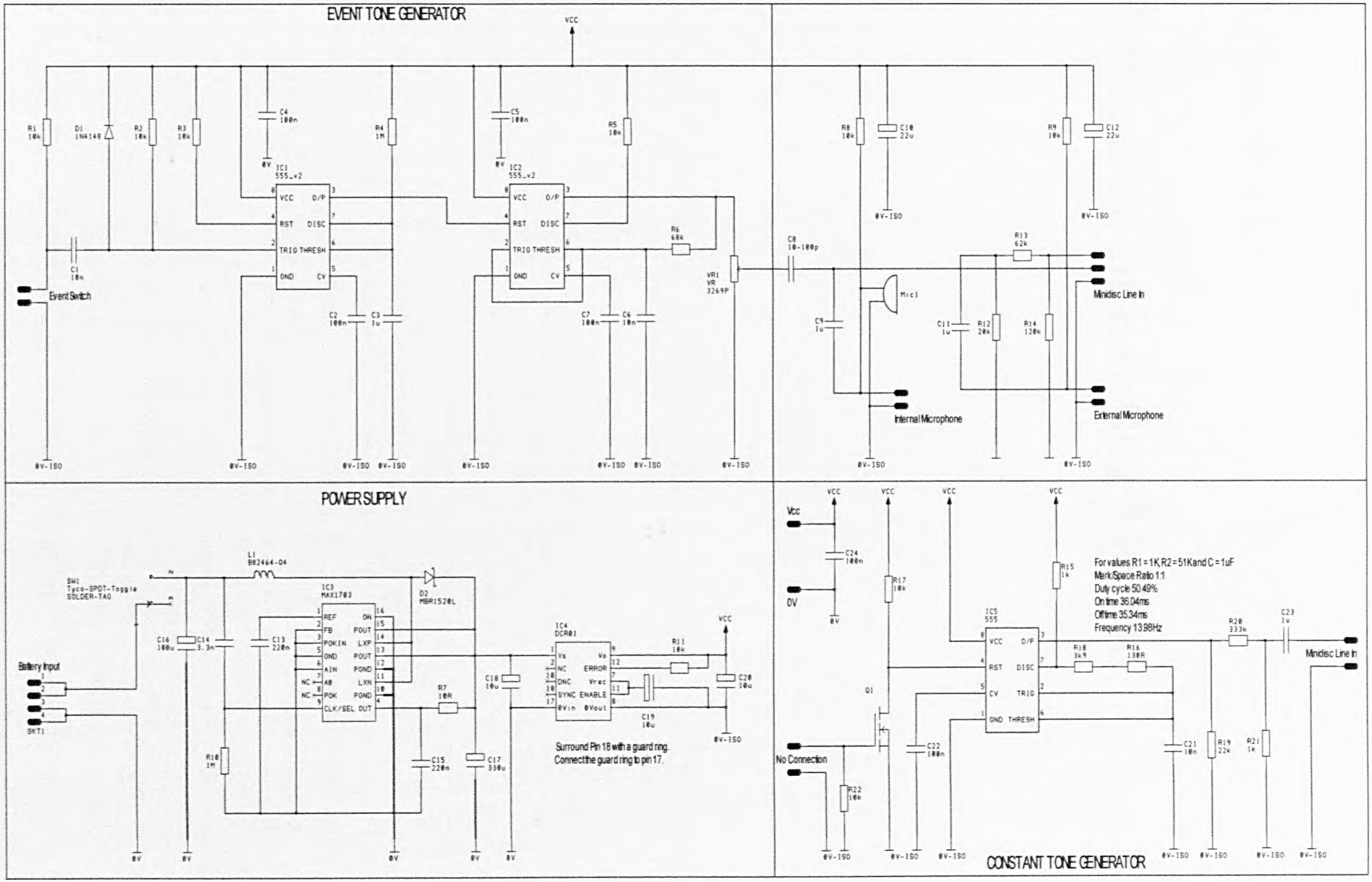
1.5.2 Activity list compiler, 'activity'

```
function [varargout] = activity(fname)
% Compile list of activities function
% Developed by Samantha Barry
global COUGH_

[pname,fname]=fileparts(fname);
if isempty(pname)
 pname=COUGH_.path;
end
```

```
fname=fullfile(pname,fname);

%load ('Time_ind')
fid=fopen('activity.txt','wt');

for t=1:size(Time_ind)
% take a marker...
 mark=Time_ind(t);

% add two seconds to find the end time
 play=88200;
 mend=mark+play;

 %load the sound
 [y,fs,au.mode,au.fp]=readwav(fname,'f',play,mark);
 % take just the left channel and play
 sig=y(:,1);
 time=timestr(mark,fs);
 soundsc(sig,fs);
 activity=input('Please enter activity (or "p" to play again)...','s');

 while activity=='p'
    soundsc(sig,fs);
    disp('replay')
    activity=input('Please enter activity (or "p" to play again)...','s');
 end
 fprintf(fid, 'Sample index at start of marker is %d,\n Corresponding time is %s,\n Activity
undertaken is %s,\n\n',mark,time,activity);

 disp('Press any key for next activity...')
 pause
end
disp('End of activities')
fclose(fid)
```

Appendix B

Circuit Diagram


EVENT TONE GENERATOR
VCC
R1 10k
D1 1N4148
R2 10k
R3 10k
C4 100n
R4 1M
C5 100n
R5 10k
IC1 555_v2
IC2 555_v2
VCC O/P RST DISC TRIG THRESH GND CV
R6 68k
VR1 VR 3269P
C1 10n
Event Switch
C2 100n
C3 1u
C7 100n
C6 10n
0V-ISO
R8 10k
C10 22u
R9 10k
C12 22u
C8 10-100p
R13 62k
Mic1
C9 1u
C11 1u
R12 20k
R14 120k
Minidisc Line In
Internal Microphone
External Microphone
POWER SUPPLY
L1 B82464-04
SW1 Tyco-SPDT-Toggle SOLDER-TAG
Battery Input
SKT1
IC3 MAX1703
REF FB POKIN GND AIN AO NC POK CLK/SEL ON POUT LXP POUT PGND LXN PGND OUT
C16 100u
C14 3.3n
C13 220n
D2 MBR1520L
R7 10R
C18 10u
IC4 DCR01
Vs NC DNC SYNC Vin ENABLE Vout Vo ERROR Vrec
R11 10k
C20 10u
C19 10u
R10 1M
C15 220n
C17 330u
Surround Pin 18 with a guard ring.
Connect the guard ring to pin 17.
Vcc
0V
C24 100n
R17 10k
R15 1k
For values R1 = 1K, R2 = 51K and C = 1uF
Mark/Space Ratio 1:1
Duty cycle 50.49%
On time 35.04ms
Offtime 35.34ms
Frequency 13.98Hz
IC5 555
VCC O/P RST DISC CV TRIG GND THRESH
R18 3k9
R16 130R
R20 333k
C23 1u
Minidisc Line In
Q1
No Connection
R22 10k
C22 100n
C21 10n
R19 22k
R21 1k
CONSTANT TONE GENERATOR

Appendix C

Publication of Existing Method

Cough

BioMed Central

Methodology

Open Access

The automatic recognition and counting of cough

Samantha J Barry[1], Adrie D Dane[1], Alyn H Morice*[2] and Anthony D Walmsley[1]

Address: [1]Department of Chemistry, Faculty of Science and the Environment, University of Hull, Cottingham Road, Hull, HU6 7RX, UK and [2]Department of Academic Medicine, University of Hull, Cottingham Road, Hull, HU6 7RX, UK

Email: Samantha J Barry - s.j.barry@chem.hull.ac.uk; Adrie D Dane - adriedane@danmetrics.com; Alyn H Morice* - a.h.morice@hull.ac.uk; Anthony D Walmsley - a.d.walmsley@hull.ac.uk

* Corresponding author

Published: 28 September 2006

Cough 2006, 2:8 doi:10.1186/1745-9974-2-8

Received: 02 March 2006
Accepted: 28 September 2006

This article is available from: http://www.coughjournal.com/content/2/1/8

Abstract

Background: Cough recordings have been undertaken for many years but the analysis of cough frequency and the temporal relation to trigger factors have proven problematic. Because cough is episodic, data collection over many hours is required, along with real-time aural analysis which is equally time-consuming.

A method has been developed for the automatic recognition and counting of coughs in sound recordings.

Methods: The Hull Automatic Cough Counter (HACC) is a program developed for the analysis of digital audio recordings. HACC uses digital signal processing (DSP) to calculate characteristic spectral coefficients of sound events, which are then classified into cough and non-cough events by the use of a probabilistic neural network (PNN). Parameters such as the total number of coughs and cough frequency as a function of time can be calculated from the results of the audio processing.

Thirty three smoking subjects, 20 male and 13 female aged between 20 and 54 with a chronic troublesome cough were studied in the hour after rising using audio recordings.

Results: Using the graphical user interface (GUI), counting the number of coughs identified by HACC in an hour long recording, took an average of 1 minute 35 seconds, a 97.5% reduction in counting time. HACC achieved a sensitivity of 80% and a specificity of 96%. Reproducibility of repeated HACC analysis is 100%.

Conclusion: An automated system for the analysis of sound files containing coughs and other non-cough events has been developed, with a high robustness and good degree of accuracy towards the number of actual coughs in the audio recording.

Background

Cough is the commonest symptom for which patients seek medical advice [1]. Population studies reported prevalence of cough to vary between 3% and 40% [2-4]. As cough affects us all, its management has massive health economic consequences with the use of over-the-counter cough remedies in the UK being estimated at 75 million sales per annum [5]. Cough is conventionally considered

to consist of an initial deep inspiration followed by expiration against a closed glottis that then opens [6-8]. As a result a characteristic phonation is formed, which is composed of two distinct components termed first and second cough sounds [6,7].

Whilst the recognition of a single cough event is relatively easy, the assessment of cough frequency over a long period of time remains difficult both for clinical and research purposes. Part of the problem is the paroxysmal nature of cough necessitating recording over a prolonged time period in order to generate an accurate estimate of cough frequency. Subjective recording or scoring of cough is unreliable as individual perception of cough differs from mild irritation to marked impairment of quality of life [9,10]. In addition, subjective assessment of cough frequency during the night-time has been shown to be unreliable [11,12]. The simple recording of cough sound using a microphone and cassette recorder allows for counting of the cough events, however, analysis is very time consuming even with the application of sound activated recording or methods for removing silence [7,8,13,14]. Similarly, the use of cough recorders that incorporate electromyogram (EMG) [15,16] or modified Holter monitor [17,18] require manual reading of the recorded tapes by a trained investigator. Automatic cough recognition from ambulatory multi-channel physiological recordings have been reported [19]. Here we describe a method for automatic recognition and counting of coughs solely from sound recordings which reduces the processing time and removes the need for trained listeners.

Materials and methods

The method, Hull Automatic Cough Counter (HACC) operates in three steps.

Firstly, the signal is analysed to identify periods of sound within the recordings; these sound events are then extracted and any periods of silence are omitted from further analysis. Secondly, digital signal processing (DSP) is applied to calculate the characteristic feature vectors which represent each sound event. The techniques used are linear predictive coding (LPC) and a bank-of-filters front-end processor. The resultant coefficients are reduced by principal component analysis (PCA); this step highlights the components of the data that contain the most variance, such that only these components are used for further analysis. Thirdly, the sound events are then classified into cough and non-cough events by use of a probabilistic neural network (PNN) [20]. The PNN is trained to recognise the feature vectors of reference coughs and non-coughs and classify future sound events appropriately.

Parameters such as the total number of coughs and cough frequency as a function of time can be calculated from the results of the audio processing. Currently, the determination of the number of coughs inside each cough event is carried out by a human listener.

Subjects and sound recording

Thirty three smoking subjects, 20 male and 13 female aged between 20 and 54 with a chronic troublesome cough were studied in the hour after rising. The smoking histories of the subjects ranged between 5 and 100 pack years with a mean of 21.4. As part of a previously published controlled trial [21] a cigarette was administered 20 minutes after the start of recording. All the subjects were studied in the outpatients clinic with the subjects ambulatory and television and conversation freely permitted.

Sound was recorded at a sampling frequency of 48 kHz using a Sony ECM-TIS Lapel microphone connected to a Sony TCD-D8 Walkman DAT-recorder. For each of the subjects, this recording was converted into 44.1 kHz 16 bit mono Microsoft wave format. To minimise data storage the sound recordings are initially analysed at a sampling frequency f_s of 11.025 kHz by using only every fourth point.

Software and hardware

All software was developed under Matlab® version 6.1 [22]. The following Matlab toolboxes were used: PLS_Toolbox version 2.1.1 [23], Signal processing toolbox version 5.1 [24], Neural network toolbox version 4.0.1 [25] and Voicebox (a free toolbox for speech recognition) [26]. The programs were executed under Windows 2000 on a 1.4 GHz Pentium 4 PC with 256 megabytes of RAM.

Determination and classification of sound events

Figure 1 shows a schematic representation of the HACC operation. Table 1 defines the variables and symbols used in the analysis.

The first step is the isolation of sound events, as shown in Figure 1 (a to h).

The audio recording is initially converted into a 44.1 kHz 16 bit mono Microsoft digital wave file. For this process, the sound recordings are analysed at a sampling frequency of 11.025 kHz. The signal is then analysed using the moving windowed signal standard deviation σ_{signal}, i.e. the standard deviation as a function of time. The moving window works along the entire length of the audio signal, taking each frame as the centre of a new window. This windowed standard deviation is similar to the more commonly used root mean square signal however, it corrects for deviations of the mean from zero. Portions of the signal containing no sound events will show a reasonably constant background signal (baseline) with small devia-

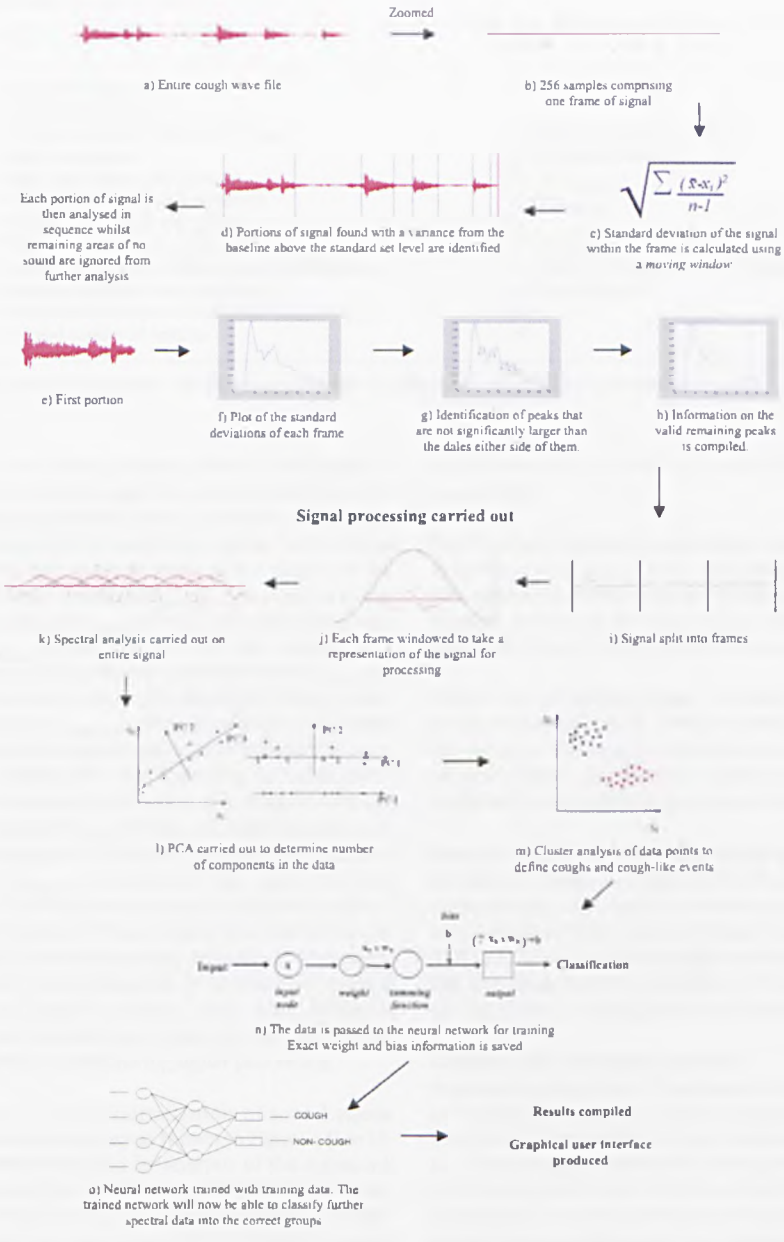


Figure 1
Pattern Recognition Approach to cough/non-cough classification.

Table 1: Symbols used and their settings.

Symbol	Meaning	Value
f_s	Sampling Frequency	11025 Hz
t	Time in milliseconds	
σ_{signal}	Windowed standard deviation of signal	Calculated as a function of time
$\Delta t_{background}$	Background interval	11026 points (1000 ms)
$thresh_{peak}$	High (event detection) threshold	10 ($\times\sigma_{background}$)
$thresh_{limits}$	Low (event start and end) threshold	2 ($\times\sigma_{background}$)
$\sigma_{background}$	Standard deviation of background	
N_{train}	Number of reference patterns	150 (75 cough/75 non-cough)
$n_{b\text{-}o\text{-}f}$	Number of mel bank-of-filters cepstral coefficients	42 (14+14 1st derivatives +14 2nd derivatives)
n_{LPC}	Number of LPC cepstral coefficients	14 (no derivatives)
$N_{cepstral}$	Total number of cepstral coefficients ($n_{B\text{-}O\text{-}F} + n_{LPC}$)	56
N_{PCA}	Reduced number of features	45

Settings are based on established values and preliminary experiments. Symbols only used locally are explained in the text.

tion relating to the inherent noise present in the signal. A sound event will cause the signal to rise above the baseline with a magnitude proportional to the validity of the signal. The moving window technique ensures the standard deviation of the background signal is not fixed for the duration of the signal; instead $\sigma_{background}$ at time t is calculated as the minimum σ_{signal} between the start of the window, $t - \Delta t_{background}$ and the end of the window, $t + \Delta t_{background}$. Sound events are thus detected when σ_{signal} for a particular window exceeds the threshold value, $thresh_{peak}$, multiplied by $\sigma_{background}$ for that window. Although this procedure means that sound sensitivity varies to a certain extent, it allows for peak detection in noisy backgrounds. The start and end values of a sound event are defined as the nearest σ_{signal} before and after the peak maximum which are below the defined low level calculated by $thresh_{limits} \times \sigma_{background}$. Portions of the signal that are below this low level are removed and excluded from further analysis (Figure 2). The amount of noise within the section of signal is then reduced by smoothing. The standard deviations for each frame in the section are plotted and treated as a series of peaks. Peaks with variations lower than the noise-level are removed. The remaining frames of signal are compiled for signal processing.

The second step is the characterisation of sound events using a signal processing step as shown in Figure 1 (i to k). The sound events identified by analysis of the signal are then characterised. Each window undergoes a parameter measurement step in which a set of parameters is determined and combined into a test pattern (termed a feature vector). Because windowing is used, multiple test patterns are created for a single sound event. These test patterns are compared with a set of N_{train} reference patterns for which the cough/non-cough classification is known. Depending on whether the test patterns are more similar to the cough or the non-cough reference patterns the corresponding sound event is classified as a cough or non-cough event respectively.

The third step is pattern comparison and decision-making as shown in Figure 1 (l to o). For this HACC uses a PNN. This network provides a general solution to pattern classification problems by following a Bayesian classifiers approach. The PNN stores the reference patterns.

Instead of classifying single patterns, HACC classifies complete sound events. The p_k values for all test patterns belonging to the sound event are summed yielding a sum of probabilities $\sum p_k$ for each class k. The sound event is classified as a member of the class with the largest $\sum p_k$.

Manual cough recognition and counting

In order to create and test the HACC program, reference measurements are required. For this purpose a graphical user interface (GUI) was developed (see Figure 3). This GUI lets the user scroll through a recording while displaying the corresponding waveform. The displayed sound can be played and coughs can be identified.

Creation of the reference patterns

Sound recordings from 23 subjects are used to create a set of 75 cough patterns and 75 non-cough patterns. The first step is to identify suitable cough and non cough events in all 23 recordings. Suitability is determined by the clarity of the sound, and by its ability to add relevant variation to the dataset. Non cough events are sounds present in the audio recording which are not coughs. These events are combined into a cough pattern matrix $\mathbf{X}_{cough}$ (10324 cough patterns) and a non-cough pattern matrix $\mathbf{X}_{non\text{-}cough}$ (254367 non-cough patterns). The length of the feature vectors in these matrices is reduced by performing a principal component analysis (PCA) [27]. The combined $\mathbf{X}_{cough}$, $\mathbf{X}_{non\ cough}$ matrix is first auto-scaled (scaling of the

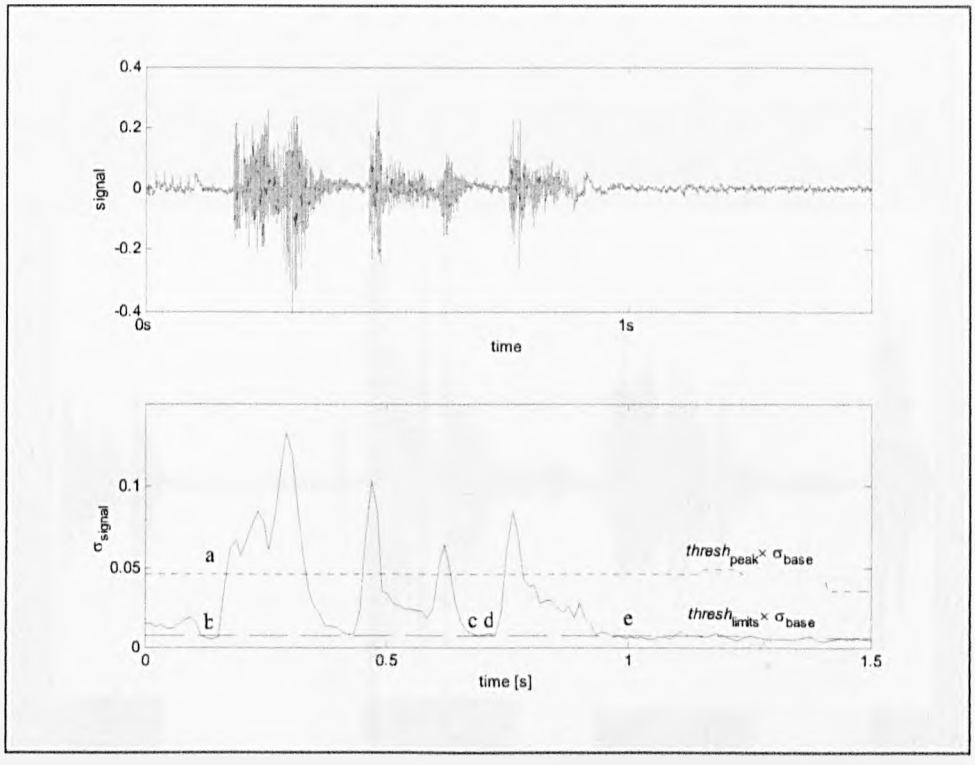


Figure 2
Sound detection. The top graph shows the original sound signal. In the bottom graph depicts σ_{signal} and the two baseline threshold lines in which $thresh_{peak} = 10$ and $thresh_{limits} = 1.5$. Point 2(a) indicates the first standard deviation larger than $thresh_{peak} \times \sigma_{background}$. Points 2(b) and 2(c) are the points nearest to point 2(a) where σ_{signal} is smaller than $thresh_{limits} \times \sigma_{background}$. The whole region between points 2(b) and 2(c) is a sound event. In the same way, the region between points 2(d) and 2(e) will be detected as a sound event.

feature values to zero mean and unit variance [28,29]) then as defined by PCA, only the scores that describe more than 0.5% of the variance are used. Experimental data is scaled using the means and variances of the reference data and projected onto the principal component space using a projection matrix. The reference patterns used for creation of the PNN are obtained by performing two k-means [30] clusterings ($k = 0.5N_{train}$) of approximately 2000 cough and non-cough patterns. The initial 2000 patterns are selected from $\mathbf{X}_{cough}$ and $\mathbf{X}_{non\ cough}$. The reference patterns are then passed through the PNN for future classification of cough and non-cough patterns.

For validation, one hour recordings of a further 10 subjects, not previously used in the creation of cough patterns, were analysed by two independent listeners (methods A and B) and HACC (+ listener for actual cough counting; method C). Listener A was an experienced cough counter that worked in the cough clinic, whilst listener B was a undergraduate project student with no expe-

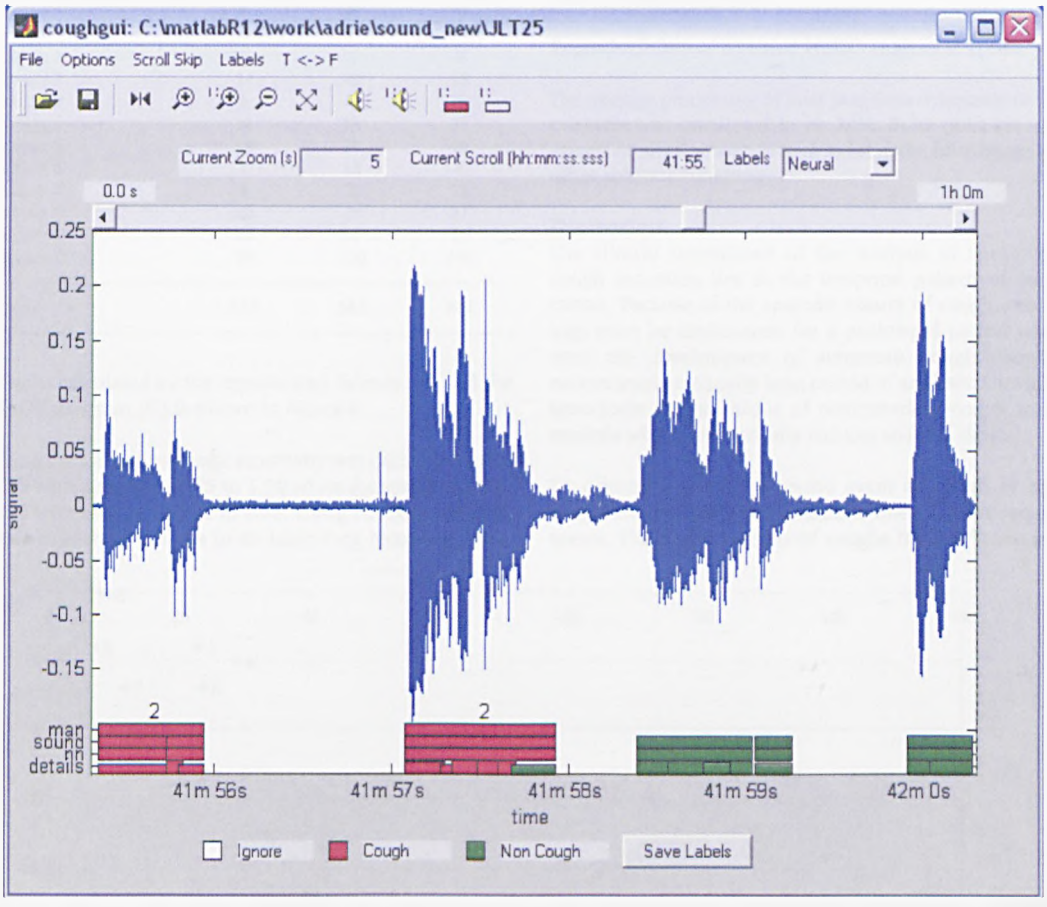


Figure 3
Graphical User Interface (GUI) for human listener.

rience of cough counting. Cough is defined as an explosive sound separated by a fall of sound level to below threshold. Thus, a peel of coughs is counted as a number of separate coughs. We have recognised that a small number of cough events occur with a double sound element of under one second duration and we have programmed HACC to recognise these and identify them to the operator who decides whether they wish to classify them as single or multiple coughs.

Currently, HACC identifies coughs and labels them, though as yet does not automatically count them. Therefore a further listener was also used in method C to count the labelled coughs using the GUI. Subsequently the GUI was used to definitively identify cough and non cough events in the recordings to establish HACC's sensitivity and specificity.

Results

Table 2 lists the total number of coughs reported by the human observers and HACC. The experienced observer frequently reported fewer coughs, mean 23.7 than either the inexperienced observer or HACC both 34.2, $p \leq 0.05$. A Bland-Altman plot comparing the total number of

Table 2: Counted coughs

	A	B	C
subject 1	8	6	8
subject 2	21	22	25
subject 3	5	6	9
subject 4	26	25	31
subject 5	14	30	28
subject 6	9	13	9
subject 7	8	8	15
subject 8	20	29	27
subject 9	28	53	50
subject 10	98	150	140
Mean	23.7	34.2	34.2

coughs calculated by the experienced listener (A) and the HACC program (C) is shown in Figure 4.

Using the GUI, the average sensitivity was calculated to be 0.80 with a range of 0.55 to 1.00 while the specificity was 0.96 with a range of 0.92 to 0.98. Using HACC it was possible to identify coughs in an hour long recording in an average time of 1 minute 35 seconds, a reduction of 97.5% in counting time.

Reproducibility of repeated HACC analysis is 100%.

The average percentage of false positives compared to true positives was calculated to be 20%. False positives were caused by similar sounds such as laughter, loud bangs and other subjects coughing.

Discussion

The clinical importance of the analysis of continuous cough recording lies in the temporal pattern of cough events. Because of the episodic nature of cough, recordings must be undertaken for a prolonged period which until the development of automatic cough counting necessitated an equally long period of analysis. Our study introduces the technique of computerised cough sound analysis which dramatically reduces analysis time.

To optimally classify a sound event as cough or non-cough the feature vectors should obey certain requirements. The feature vectors of coughs from different sub-

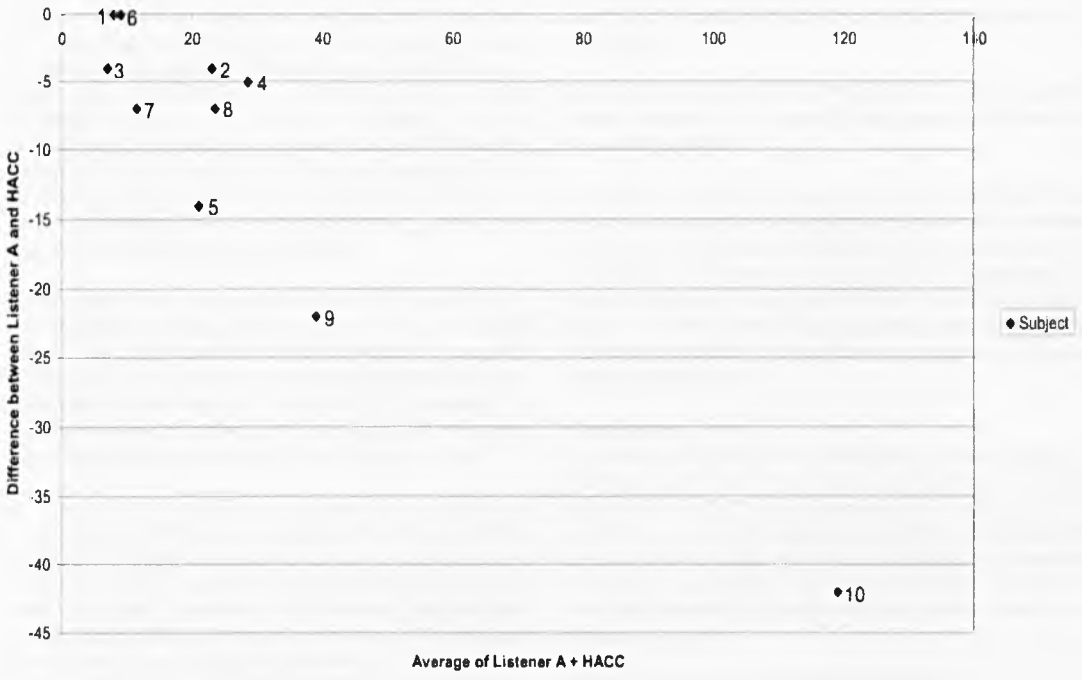


Figure 4
The Bland Altman plot showing the difference between the total number of coughs per subject as recorded by the experienced listener (A) compared to the HACC program (C).

jects should have similar values. Non-cough events should give dissimilar feature vectors than cough events. The features should not be correlated and preferably follow a probability distribution that is well described as a sum of Gaussians. It is also desirable that the features do not depend on the sound amplitude; the cough loudness is not the same for different people and it makes the placement of the microphone less critical.

These requirements are very similar to those in speech recognition. To recognise speech, a reliable, robust and most widely used feature set based on frequency content are cepstral coefficients. Cepstral coefficients are the coefficients of the Fourier transform representation of the log magnitude spectrum. They are good at discriminating between different phonemes (speech sounds), are fairly independent of each other and have approximately Gaussian distribution for a particular phoneme. The cepstral coefficients are normally calculated via one of the following two pre-processing routes: linear predictive coding (LPC) or a bank-of-filters front-end processor [26, 31, 32].

The recognition performance can be improved by extending the representation with temporal cepstral derivative information. Cepstral derivatives are obtained by the published method [32]. The feature vectors used in HACC consist of a pre-treated combination of n_{BOF} mel bank-of-filters cepstral coefficients with their first and second derivatives and n_{LPC} LPC cepstral coefficients (without derivatives). Pre-treatment consists of scaling followed by projection into the principal component space obtained for the reference samples. This pre-treatment reduces the number of features in each pattern from $N_{cepstral} (= n_{BOF} + n_{LPC}$ in this study 56) to N_{PCA} (here 45).

The coughs in the cough events need to be counted by a human listener. For this purpose, the GUI is used. However, in this procedure only events classified as cough by HACC have to be listened to. This procedure yields a huge reduction in listening time (mean 97.5%) compared to human counting. Our aim is to improve HACC in the near future so that human counting is no longer necessary.

The results show a significant increase in the number of coughs reported by the inexperienced listener and HACC compared to those reported by the experienced listener. This difference is caused by both the inexperienced listener and HACC detecting and counting coughs from sources other than the subject under study. The subjects were recorded in a clinic alongside other patients and as a result, other coughs are clearly audible on the recordings. The inexperienced listener B and HACC simply counted all audible coughs which explains why the data from B and C are so similar, and exaggerated. Clearly the experience of listener A discerns between the subject closest to the microphone and the other cough events that are audible on the recordings. Thus it is clear that even with this slight disparity between the computer and the experienced listener, the computer has in fact classified all the coughs on the recordings, but without any distinction as to the source of the coughs.

Since HACC is not subject-specific in its cough classification, improving the counting accuracy is best achieved by excluding the non-subject coughs from the recording. Using a different microphone with a lower sensitivity will ensure only high-amplitude sounds occurring close to the microphone will be detected, thus discerning the subject's coughs from ambient coughs. This modification will also diminish problems with background noise. The recordings for this study were all made in a similar environment, with the subjects ambulatory and television and conversation freely permitted. The use of a lower sensitivity microphone will help to diminish background noise before any processing by HACC is carried out.

For the development of HACC processing, hour long recordings of each subject were made, it was felt that this duration of recordings contained a sufficient number of cough and non-cough events to carry out an assessment of the system.

Future work will test HACC's ability to process much longer duration of recordings containing a wider variety of patient groups.

One of the major advantages of the automated recording is that it is possible to re-analyse the data with minimal effort and achieve consistent results. Thus, when the same recordings were reprocessed the events classified as coughs in one run were also found to be coughs in subsequent runs. This allows development of a statistically stable analysis method with a known statistical confidence limit on the results.

Conclusion

An automated system for the analysis of sound files containing coughs and other non-cough events has been developed, with a high robustness and good degree of accuracy towards the number of actual coughs in the audio recording. Although HACC is unable to distinguish between coughs of the subject under study and ambient coughs, changes to the hardware could resolve this problem in the future.

Declaration of competing interests

The author(s) declare that they have no competing interests.

Authors' contributions

AD developed the HACC program, processed the recordings and was the listener to obtain results for HACC. AD also drafted the manuscript.

AM designed and coordinated the clinical studies and assisted with the manuscript.

AW and SB developed the manuscript.

All authors read and approved the final manuscript.

Acknowledgements

Funding: Engineering and Physical Sciences Research Council (EPSRC).

References

1. Irwin RS, Boulet LP, Cloutier MM, Fuller R, Gold PM, Hoffstein V, Ing AJ, Mccool FD, O'Byrne P, Poe RH, Prakash UB, Pratter MR, Rubin BK: **Managing cough as a defense mechanism and as a symptom. A consensus panel report of the American College of Chest Physicians [see comments]. [Review] [325 refs].** *Chest* 1998, **114:**133S-181S.
2. Fuller RW, Jackson DM: **Physiology and treatment of cough. [Review] [64 refs].** *Thorax* 1990, **45:**425-430.
3. Loundon RG, Brown LC: **Cough frequency in patients with respiratory disease.** *Am Rev Respir Dis* 1967, **96:**1137-1143.
4. Cullinan P: **Persistent cough and sputum: prevalence and clinical characteristics in south east England.** *Resp Med* 1992, **86:**143-149.
5. Fuller RW: **Cough.** In *Respiratory Medicine* Second Edition edition. Edited by: Brewis RAL, Cirrin B, Geddes DM and Gibson GJ. London, W.B. Saunders Company Ltd; 1995:238-242.
6. Widdicombe JG: **Sensory neurophysiology of the cough reflex. [Review] [40 refs].** *J Allergy Clin Immunol* 1996, **98:**S84-S89.
7. Korpas J, Sadlonova J, Vrabec M: **Analysis of the cough sound: an overview.** *Pulm Pharmacol* 1996, **9:**261-268.
8. Piirila P, Sovijarvi ARA: **Objective assessment of cough.** *Eur Respir J* 1995, **8:**1949-1956.
9. French CL, Irwin RS, Curley FJ, Krikorian CJ: **Impact of chronic cough on quality of life [see comments].** *Arch Intern Med* 1998, **158:**1657-1661.
10. Thompson R, Kastelik JA, Ojoo JC, Wright CE, Beaumont LA, Redington AE, Morice AH: **Impact of chronic cough on health.** *Thorax* 2001, **56:**71iii.
11. Falconer A, Oldman C, Helms P: **Poor agrement betwen reported and recorded nocturnal cough in asthma.** *Pediatric Pulmonology* 1993, **15:**209-211.
12. Archer LNJ, Simpson H: **Night cough counts and diary card scores in asthma.** *Arch Dis Child* 1985, **60:**473-474.
13. Subburaj S, Parvez L, Rajagopalan TG: **Methods of recording and analysing cough sounds.** *Pulm Pharmacol* 1996, **9:**269-279.
14. Rece CA, Cherry AC, Reece AT, Hatcher TB, Diehl AM: **Taperecorder for evaluation of coughs in children.** *Am J Dis Child* 1966, **112:**124-128.
15. Hsu JY, Stone RA, LoganSinclair RB, Worsdell M, Busst CM, Chung KF: **Coughing frequency in patients with persistent cough: Assessment using a 24 hour ambulatory recorder.** *Eur Respir J* 1994, **7:**1246-1253.
16. Munyard P, Bust C, Longansinclair R, Bush A: **A new device for ambulatory cough recording.** *Pediatric Pulmonology* 1994, **18:**178-186.
17. Chang AB, Newman RG, Phelan PD, Robertson CF: **A new use for an old Holter monitor: An ambulatory cough meter.** *Eur Respir J* 1997, **10:**1637-1639.
18. Chang AB, Newman RG, Carlin JB, Phelan PD, Robertson CF: **Subjective scoring of cough in children: parent-completed vs child-completed diary cards vs an objective method.** *Eur Respir J* 1998, **11:**462-466.
19. Wasserman PD: **Advanced Methods in Neural Computing.** Van Nostrand Reinhold, New York; 1993.
20. Mulrennan S, Wright C, Thompson R, Goustas P, Morice A: **Effect of salbutamol on smoking related cough.** *Pulm Pharmacol Ther* 2004, **17:**127-131.
21. **The Mathworks Inc. Matlab 6 User Manual.** 2000.
22. Wise BM, Gallagher NB: **PLS_Toolbox for use with Matlab Version 2.1.** Manson, WA, Eigenvector Research, Inc.; 2000.
23. **The Mathworks Inc. Signal processing Toolbox for use with Matlab User's Guide.** 2000.
24. **The Mathworks Inc. Neural Network Toolbox for use with Matlab User's Guide.** 2000.
25. Brookes M: **Voicebox: Speech Processing Toolbox for Matlab.** Department of Electrical & Electronic Engineering, Imperial College; 2002.
26. Jackson JE: **Principal Components and Factor Analysis: Part 2- Additional Topics Related to Principal Components.** *Journal of Quality Technology* 1981, **13:**.
27. P G, B.R. K: **Partial LeasGeladi P.and Kowalski B.R.t-Squares Regression: A Tutorial.** *Volume 185.* Analytica Chimica Acta; 1986:1-17.
28. Massart DL, Vandeginste BGM, Buydens LMC, De Jong S, Lewi PJ, Smeyers-Verbeke J: **Handbook of Chemometrics and Qualimetrics: Part A.** Amsterdam, Elsevier; 1997.
29. MacQueen J: **Some methods for classification and analysis of multivariate observations.** In *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probabilit Volume 1.* Berkeley, California, University of California Press; 1967:281-297.
30. F.J. O: **Signal Processing of Speech.** Macmillan; 1993.
31. L. R, Juang BH: **Fundamentals of Speech Recognition.** Prentice Hall; 1993.