INFORMATION TO USERS

This manuscript has been reproduced from the microfilm master. UMI films the text directly from the original or copy submitted. Thus, some thesis and dissertation copies are in typewriter face, while others may be from any type of computer printer.

The quality of this reproduction is dependent upon the quality of the copy submitted. Broken or indistinct print, colored or poor quality illustrations and photographs, print bleedthrough, substandard margins, and improper alignment can adversely affect reproduction.

In the unlikely event that the author did not send UMI a complete manuscript and there are missing pages, these will be noted. Also, if unauthorized copyright material had to be removed, a note will indicate the deletion.

Oversize materials (e.g., maps, drawings, charts) are reproduced by sectioning the original, beginning at the upper left-hand corner and continuing from left to right in equal sections with small overlaps. Each original is also photographed in one exposure and is included in reduced form at the back of the book.

Photographs included in the original manuscript have been reproduced xerographically in this copy. Higher quality 6" x 9" black and white photographic prints are available for any photographs or illustrations appearing in this copy for an additional charge. Contact UMI directly to order.

UMI

A Bell & Howell Information Company
300 North Zeeb Road, Ann Arbor MI 48106-1346 USA
313/761-4700 800/521-0600

SPEAKER IDENTIFICATION USING THE WAVELET TRANSFORM

By
Andrew H. Mudry. B.Sc.

A THESIS SUBMITTED TO THE FACULTY OF
GRADUATE STUDIES AND RESEARCH
IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF

MASTER OF ENGINEERING

OTTAWA-CARLETON INSTITUTE FOR ELECTRICAL ENGINEERING
FACULTY OF ENGINEERING
DEPARTMENT OF ELECTRONICS
CARLETON UNIVERSITY
OTTAWA, ONTARIO

April 1997

© ANDREW MUDRY 1997

National Library
of Canada

Acquisitions and
Bibliographic Services

395 Wellington Street
Ottawa ON K1A 0N4
Canada

Bibliothèque nationale
du Canada

Acquisitions et
services bibliographiques

395, rue Wellington
Ottawa ON K1A 0N4
Canada

Your file Votre référence

Our file Notre référence

The author has granted a non-exclusive licence allowing the National Library of Canada to reproduce, loan, distribute or sell copies of this thesis in microform, paper or electronic formats.

The author retains ownership of the copyright in this thesis. Neither the thesis nor substantial extracts from it may be printed or otherwise reproduced without the author's permission.

L'auteur a accordé une licence non exclusive permettant à la Bibliothèque nationale du Canada de reproduire, prêter, distribuer ou vendre des copies de cette thèse sous la forme de microfiche/film, de reproduction sur papier ou sur format électronique.

L'auteur conserve la propriété du droit d'auteur qui protège cette thèse. Ni la thèse ni des extraits substantiels de celle-ci ne doivent être imprimés ou autrement reproduits sans son autorisation.

0-612-22123-7

The undersigned recommend to the Faculty Of Graduate Studies and Research
acceptance of the thesis

SPEAKER IDENTIFICATION USING THE WAVELET TRANSFORM

submitted by Andrew H. Mudry, B.Sc.
in partial fulfilment of the requirements for
the degree of Master Of Engineering

Dr. J. S. Wight
Thesis Supervisor

Dr. J. S. Wight
Chair, Department Of Electronics

Carleton University
April 1997

Abstract

The application of the wavelet transform to the generation of feature vectors for a closed set, text independent speaker identification problem is examined. Three sets of wavelet derived feature vectors are applied to the speaker identification problem and the identification accuracy is compared against the accuracy obtained using feature vectors generated from standard cepstral coefficients.

Results obtained indicate that the half band filter bank structure of the standard wavelet transform is too restrictive for the analysis of speech signals in speaker identification applications, since it does not provide the necessary spectral resolution in the range 0 - 1.0 kHz. Standard signal processing techniques based on the Fast Fourier Transform, while involving slightly more operations, do not have this limitation and therefore seem to be more suitable for the processing of speech signals for speaker identification purposes.

Acknowledgements

The author wishes to acknowledge the supervision and direction given by Professor Jim Wight of Carleton University and the technical guidance provided by Mr. Luc Gagnon of the Canadian Department Of National Defence. Also of tremendous value was the support given by Dr. Raymond Carbonneau, Mr. Pierre Yansouni and the staff of the Electronic Support Measures Section of the Defence Research Establishment Ottawa. The artistic skill of Mr. Bob Gervais from DREO was also invaluable as was the computer system support received from Mr. Graham Estey and Mr. Bob Humble of Software Kinetics Ltd.

Also for the inspiration and motivation received, I wish to acknowledge other graduate students who have completed thesis work at DREO. To Sean, Harvey, and Wayne - here it is at long last!

Above all, this thesis could not have been completed without the tremendous support, encouragement, and patience provided to me by my family and by my fiancée Marielle. For this, I will always be grateful.

Glossary

- affricates - sounds which result from the concatenation of a stop sound, followed by a fricative sound which shares the same place of articulation as the stop.
- allophones - phonemes that represent the same sound, but are produced by distinctly different positions of the vocal tract articulators.
- basilar membrane - an inner ear membrane which serves to transfer the mechanical disturbances of the eardrum into spatio-temporal disturbances along its length.
- cilia - a network of tiny filaments which transduce the spatio-temporal basilar membrane disturbances into electrical impulses.
- Corti Organ - the organ which lies on top of the basilar membrane in the inner ear and contains the cilia.
- coarticulation - the overlapping of adjacent phonemes in a sentence caused by anticipatory movement of a given articulator.
- cochlea - the first of two fluid filled chambers in the inner ear which vibrate in response to the mechanical disturbances of the eardrum.
- diphones - pairwise combinations of phonemes.
- diphthongs - sounds produced by varying the vocal tract configuration smoothly from one vowel position to another.

egressive sound - sound generated during expiration.

formant frequencies - frequencies of vocal tract resonance.

frame - a finite segment of speech samples, usually about 10.0 to 20.0 msec in duration. The speech signal is generally considered to be statistically stationary within a frame.

fricatives - sounds produced as a result of turbulence created by forcing glottal excitation through a narrow vocal tract opening.

fundamental frequency - rate with which vocal folds are opened and closed during voiced speech.

glides - vowel like sounds in which the vocal tract has a narrower constriction than for normal vowel sounds.

glottis - space in the larynx between the vocal folds.

hybrid encoder - a speech encoder which uses aspects of both waveform and source encoding.

ingressive sound - sound generated during inhalation.

intonation - pitch variation which occurs during normal conversation.

lexicon - dictionary.

liquids - vowel like sounds in which the air flow is deflected around the tip or dorsum of the tongue.

manner of articulation - the manner in which the vocal tract articulators interact to produce a given phoneme.

meatus - the auditory canal.

nasals - voiced sounds which are produced with the vocal tract totally restricted at some point in the oral cavity.

noise masking - a characteristic of human hearing whereby large amplitude formants mask noise in the same frequency band.

non-parametric algorithm - a pattern classification algorithm which does not make an assumption as to the statistics or distribution of the pattern vectors.

obstruents - fricatives, stops, and affricates.

ossicular bones - three tiny bones in the middle ear which serve to transfer mechanical vibrations from the air filled outer ear to the fluid filled inner ear.

parametric algorithm - a pattern classification algorithm which makes an assumption as to the statistics or distribution for the pattern vectors.

phone - the physical sound produced when a phoneme is uttered.

phoneme - an abstract unit of speech used to designate one of the 20 - 40 distinctive sounds which make up a language.

pinna - the externally visible part of the ear.

pitch - an acoustic phenomena of a sound which enables a listener to perceptually place it on a scale going from low to high.

plosives - see *stops*.

prosodic features - Intensity, duration and intonation of a given segment of speech.

semi-vowel sounds - glides and liquids.

sonorants - vowels, diphthongs, glides, liquids, and nasals.

source encoder - a speech encoder which attempts to reduce the number of data bits necessary for transmission of a speech signal, by relying on a model of the speech signal.

speech encoder - a device which converts an analog speech signal into a series of bits.

speech chain - the complete set of events which occur when one person uses speech in order to convey a message to another person.

speech synthesis - artificial simulation of the speech production process.

stops - sounds generated when air pressure built up behind a total vocal tract constriction is released.

unvoiced speech - see *voiceless speech*.

vector quantization - a source encoding technique which quantizes groups of signal samples, rather than each sample individually.

vestibula - the second of two fluid filled chambers in the inner ear which vibrate in response to the mechanical disturbances of the eardrum.

voiced speech - speech generated as a result of vocal fold vibration.

voiceless speech - speech generated without vocal fold vibration.

vocal folds - two small muscular folds located in the larynx. Also known as the vocal cords.

vowels - sounds produced by exciting a fixed vocal tract with a periodic excitation produced by vocal cord vibration.

waveform encoder - a speech encoder which attempts to create a facsimile reproduction of the input speech signal through direct digitization of the speech signal.

A List of Acronyms

ADPCM -	Adaptive Differential Pulse Code Modulation.
APC -	Adaptive Prediction Coding.
AQB -	Adaptive Quantization with Backward estimation.
AQF -	Adaptive Quantization with Forward estimation.
ADC -	Analog To Digital Converter.
AM -	Amplitude Modulation.
bps -	bits per second.
CELP -	Code Excited Linear Prediction.
CWT -	Continuous Wavelet Transform.
DCT -	Discrete Cosine Transform.
DPCM -	Differential Pulse Code Modulation.
DPWT -	Discrete Parameter Wavelet Transform.
DWT -	Discrete Wavelet Transform.
FWT -	Fast Wavelet Transform.
FM -	Frequency Modulation.
GMM -	Gaussian Mixture Model.

GSM -	Groupe Special Mobile.
HMM -	Hidden Markov Model.
MP-LPC -	Multi-Pulse Linear Predictive Coding.
PARCOR -	Partial Correlation.
pdf -	probability density function.
PCM -	Pulse Code Modulation.
RF -	Radio Frequency.
RPE -	Regular Pulse Excitation.
STFT -	Short Time Fourier Transform.
SAD -	Speech Activity Detection.
SSB -	Single Side Band.
VCO -	voltage-controlled oscillator.
VQ -	Vector Quantization.
WPT -	Wavelet Packet Transform.

A List of Symbols

b -	number of bits per sample.
β -	bandwidth efficiency.
B_c -	channel bandwidth.
B_v -	baseband bandwidth of voice signal.
F_0 -	Fundamental frequency. Frequency with which vocal folds open and close during voiced speech.
f_s -	sampling frequency.
J -	number of classes present in a pattern identification problem.
K -	feature vector dimension.
L -	FIR filter order.
m -	number of symbols used in an m -ary modulation scheme.
M -	number of acoustic classes present in a GMM.
N -	number of signal samples in a frame of speech.
N_C -	number of codewords in a vector quantization codebook.
N_Q -	number of quantization levels in a quantizer.
N_S -	number of states in an HMM.

O -	maximum number of cascaded filter stages in a wavelet transform implementation.
R -	data rate.
$\sigma_{q_e}^2$ -	variance of quantization error.
σ_s^2 -	variance of unquantized signal.
SNR -	signal to noise ratio.
SNR_Q -	signal to quantization noise ratio.
V_L -	quantizer clipping voltage.

Contents

Abstract	iii
Acknowledgements	iv
Glossary	v
A List of Acronyms	ix
A List of Symbols	xi
Contents	xiii
List Of Tables	xvi
List Of Figures	xix
1 Introduction	1
1.1 Speaker Recognition	1
1.2 Thesis Outline	5
2 Introduction To Speech And Speech Processing	7
2.1 The Speech Chain	7
2.2 Message Conception	10
2.3 Speech Production	11
2.4 Speech Transmission	27
2.4.1 Waveform Encoders	33

2.4.2	Source Encoders	45
2.4.3	Hybrid Encoders	51
2.5	Speech Audition	56
2.5.1	Time Domain Parameters	61
2.5.2	Frequency Domain Parameters	64
2.5.3	LPC Coefficients	68
2.5.4	Cepstral Coefficients	73
2.6	Information Perception	78
2.6.1	Temporal Template Matching	83
2.6.2	Vector Quantization	87
2.6.3	Neural Networks	88
2.6.4	Hidden Markov Models	92
3	Wavelets And The Wavelet Transform	98
3.1	Introduction	98
3.2	The Wavelet Transform	107
3.3	Filterbanks and The Fast Wavelet Transform	113
3.4	Practical FWT Implementation Issues	127
4	Gaussian Mixture Modelling	132
4.1	Introduction	132
4.2	Model Development	133
4.3	Model Identification Phase	137
4.4	Model Training Phase	138
4.4.1	Maximum Likelihood Estimation	138
4.4.2	Auxiliary Function	139
4.4.3	Model Parameter Iteration	140
4.4.4	Initial Estimate	142
4.4.5	Practical Considerations	146
5	Experimental Results	148
5.1	Introduction	148

5.2	Speech Activity Detection	151
5.3	Cepstral Feature Vector Sets	155
5.4	Wavelet Based Feature Vector Sets	158
5.5	Results	163
5.5.1	Training Set Selection	163
5.5.2	Speaker Identification Performance	168
6	Conclusions	172
6.1	Discussion Of Results	172
6.2	Suggestions for Future Work	172
	Bibliography	173

List of Tables

2.1	Phonemes Of The English Language	13
2.2	Frequency Bands and Bit Assignments For Crochiere Sub-band Coder	44
2.3	Statistics Of English Phonemes	85
3.1	Differences Between Wavelet Coefficients	130
5.1	Speech Activity Detection Algorithm Parameters	154
5.2	Eliminated SPIDRE Target Files	164
5.3	SPIDRE Target Files With Reversed Channel Information	164
5.4	Target Speaker Training Files (SPIDRE CD-ROM 1)	166
5.5	Target Speaker Training Files (SPIDRE CD-ROM 2)	167
5.6	GMM Training Parameters	168
5.7	Speaker Identification Performance	169

List of Figures

2.1	The Speech Chain	8
2.2	Speech Production Organs	14
2.3	Cross Sectional View Of Larynx	16
2.4	Effect Of Vocal Tract On Glottal Waveform	19
2.5	Speech Synthesizer Model	23
2.6	Power Spectral Densities Of /k/ Allophones	25
2.7	Transcription Of Word <i>Carleton</i> Using Phonemes and Diphones	28
2.8	Spectrum Of Speech Coding Transmission Rates And Associated Quality	32
2.9	Waveform Quantization Techniques	34
2.10	More Waveform Quantization Techniques	35
2.11	Quantizer SNR For Uniform and μ Law Quantizers	37
2.12	Sub Band Coding Filtering Schemes	43
2.13	Source Encoders	46
2.14	Hybrid Encoders	52
2.15	The Human Ear	57
2.16	Spectra Of Phonemes / ɔ / and /s/	60
2.17	Histogram Of Speech Compared To Laplacian PDF	62
2.18	Wideband and Narrowband Spectrograms Of The Word <i>Carleton</i>	65
2.19	Bark and MEL Scale Frequency Re-mappings	69
2.20	Linear Prediction Model	70
2.21	Canonic Representation For Homomorphic System	74
2.22	Characteristic and Inverse Characteristic Systems For Speech	74
2.23	Signal, Complex Cepstrum and Spectrum For Phoneme / ɔ /	77

2.24	Conceptual Pattern Recognition Model	80
2.25	Time Warping	86
2.26	Perceptron	89
2.27	Two Dimensional Linearly Separable Classes	91
2.28	Two Dimensional Linearly Inseparable Classes	91
2.29	Single Perceptron Layer Neural Network ($K=5$, $J=3$)	93
2.30	Three Perceptron Layer Neural Network ($K=5$, $J=3$)	93
2.31	Hidden Markov Model State Diagram (5 States)	95
3.1	Segmentation Of Time Frequency Plane	101
3.2	STFT and WT Analysis Of Delta Function	103
3.3	STFT and WT Analysis Of Sinusoid Pair	104
3.4	Effects Of Shifting and Scaling Operations On Mother Wavelet Function	108
3.5	Continuous Wavelet Transform Mother Wavelet Functions	110
3.6	Daubechies_4 Mother Wavelet Function	114
3.7	Iterated Lowpass Transfer Functions	115
3.8	Constant Q Bandpass Filter Network ($O=5$)	117
3.9	Daubechies_4 Constant Q Network Responses ($O=5$)	118
3.10	Filter Impulse Functions For Daubechies_4 Constant Q Network . . .	120
3.11	Scaling Function For Daubechies_4 Mother Wavelet	121
3.12	Daubechies_12 Network Responses ($O=6$)	122
3.13	DWT Implementation ($O = 5$)	124
3.14	Equivalent Networks	125
3.15	Fast Wavelet Transform ($O = 5$)	126
3.16	FWT Coefficient Packing: $N=32$, $O=5$	129
4.1	GMM Vector Generation Model For Speaker λ_j	135
4.2	GMM Training Procedure	143
4.3	Typical Likelihood Function Convergence	144
5.1	Speaker Identification System	149
5.2	Speech Frame Processing	150

5.3	Speech Activity Detection Algorithm	152
5.4	Speech Activity Detection Algorithm Illustration	153
5.5	Cepstral Feature Vector Extraction	157
5.6	Triangular Filter Bank	157
5.7	Wavelet Feature Vector Extraction	160

Chapter 1

Introduction

1.1 Speaker Recognition

Automatic speaker recognition is the area of speech processing related to the identification of a speaker from an analysis of a received voice sample. Speaker recognition differs from speech recognition in that the latter is concerned with the extraction of some portion of the message information contained within the spoken text. This may include the extraction of linguistic units known as phonemes, the extraction of key words contained in the message, or the extraction of the complete message. Speaker recognition, on the other hand, is devoted to the analysis of subtle differences in the speech received from different speakers in order to distinguish the unique characteristics of each speaker's voice.

In both speech and speaker recognition, the received speech data is segmented into frames, and vectors of characteristic features are extracted from each frame. The type of feature extracted from the speech depends largely on whether the application is speech or speaker recognition. Speech recognition is generally concerned with the recognition of the message units (phones, words, etc.) independent of which speaker spoke the text. The features extracted for speech recognition purposes are designed to average out the slight differences in sound which occur when identical words are uttered by different speakers. Such features are therefore obviously not suitable for

speaker recognition applications. In addition to highlighting the differences which occur when identical words are uttered by different speakers, the characteristic features chosen for speaker recognition applications should ideally be [1, 2]:

- 1) commonly found in natural speech,
- 2) easily measurable,
- 3) constant over time,
- 4) unaffected by the speaker's health or mood,
- 5) unaffected by the transmission channel characteristics,
- 6) reasonably robust in the presence of background noise, and
- 7) insusceptible to mimicry.

In practice, it is highly unlikely to find a single set of features which simultaneously satisfies all of these attributes. Often a partial or total relaxation of some of the criteria is necessary [1].

A complete speaker recognition algorithm involves two stages of operation: *training* and *recognition*. In the training phase, a set of feature vectors is extracted from a sample of speech known to have originated from a particular speaker. This training set is used to create a reference template which characterizes the speaker. During the recognition phase of operation, reference templates are compared with test templates derived from a sample of speech of unknown origin, and a decision as to the identity of the unknown speaker is made.

The area of speaker recognition is commonly divided into two main problems: *verification* and *identification*. Both tasks involve the comparison of reference templates with test templates. Of the two tasks, the easier one is speaker verification, which involves a binary (yes/no) decision as to whether or not a sample of speech was spoken by the claimed originator [2]. This problem occurs commonly in applications involving access to a secure area, where a user's voice must be confirmed before entry is granted. The goal of the speaker verification algorithm in such an application is to maximize the probability of acceptance for a legitimate user who has been properly registered, while simultaneously minimizing the probability of admitting an imposter.

Speaker identification, on the other hand, involves a decision as to which of the N speakers known to the system is most likely to have uttered a given sample of speech.

Identification problems occur most often in police forensic applications, where a sample of speech must be matched to a list of suspects. In a closed set speaker identification problem, it is assumed that the unknown speaker is always contained in the database of speakers known to the system. An open set identification problem, on the other hand, allows for the possibility that the test template has been derived from a speaker which is not known to the system [3]. In the open set identification problem, the system must account for the fact that for a database with a limited number of speakers, there is an extremely high probability that the test template from a totally unknown speaker is not contained in the database. Satisfactory performance of an open set recognition algorithm therefore requires an extremely accurate characterization of the speakers contained in the database, since the rejection of the almost certain possibility that the speaker is not in the database must be statistically outweighed by considerable justification.

An unfortunate implication of the speaker identification problem in general is that as the number of speakers contained in the database increases, errors become more likely due to the fact that an N level decision must be made. While there is only one correct answer, there are $N - 1$ wrong answers. With a finite error probability for each comparison, the probability of an error increases with the number of speakers. A verification system, on the other hand always involves a binary detection problem, so the performance is independent of the number of speakers contained in the database.

Both the speaker verification and identification problems may be further classified as either *text dependent* or *text independent* problems. Text dependent speaker recognition occurs when the words uttered during training phase match the words used for the creation of a test template during the recognition phase of operation. In the text independent case, the text is totally unconstrained and phrases used during training may differ from those used during the recognition. Text dependent problems generally imply a cooperative set of speakers and are generally used in the verification task where, for example, a user may be prompted to say the same words used in training before gaining access to a secure facility. Text dependency allows the reference and test templates to be aligned on specific words in the message thereby eliminating the effects of different text from the recognition decision. Text independent

problems are generally associated with an identification task involving uncooperative speakers, where similarity between training and test phrases cannot be enforced. In order to compensate for this, text independent recognition algorithms must rely on longer term statistical calculations. This implies that more data is required during both the training and recognition phases of a text independent algorithm than a text dependent algorithm. Error rates for text independent speaker recognition are also generally higher than in text dependent cases.

One difficulty which has been faced in the development of speaker (and speech) recognition algorithms has been the unavailability of standard sets of speech upon which training and testing may be performed. Historically, developers were forced to collect their own data sets in order to validate their algorithms. This made the comparison of the performance of different developers' algorithms difficult since results achieved were often dependent on the data set used. Factors such as different levels of background noise, microphone distortion, dialect, etc. were difficult to normalize between different data sets. In order to address this limitation, the United States Defense Advanced Research Projects Agency (DARPA) and various companies formed the Linguistic Data Consortium which provides several different large collections of speech on CDROM for various applications [4]. Three commonly used speech corpora are available for both speech and speaker recognition applications: TIMIT, SWITCHBOARD, and SPIDRE. TIMIT is a collection of 6300 phonetically labelled sentences spoken by 63 different speakers. Although useful, this corpus has some limitations in that the text is read in each case, and was recorded under ideal conditions. SWITCHBOARD is a much more extensive collection of speech recorded over the telephone. It contains over 250 hours of transcribed conversational speech from 550 different speakers. SPIDRE is a subset of SWITCHBOARD which contains selected conversations specifically collated for speaker recognition work. It contains over 7 hours of conversational speech selected from 45 different target speakers. Each target speaker has been recorded under varying conditions on different microphones. The corpus also contains over 50 hours of conversations from non target speakers, making it suitable for both closed set and open set speaker recognition development.

1.2 Thesis Outline

This thesis begins in Chapter 2 with an overview of speech processing in general, with a focus on speech encoding algorithms and commonly used parameters for the characterization of speech in speech and speaker recognition applications. The discussion of sub-band coding and frequency domain parameters used in standard speech processing applications, although somewhat lengthy, is important and serves as justification for the application of a relatively new signal processing technique, the Wavelet Transform, to the speaker identification task. The rest of Chapter 2 serves as a background to the widely diverse area of speech processing and also serves to point out the similarities and subtle differences which exist in the processing of a voice signal for speech and speaker recognition applications.

Chapter 3 provides an overview of the Wavelet Transform, which has recently been developed out of the requirement to simultaneously provide high frequency resolution and accurate temporal localization in the analysis of statistically non-stationary signals. The Wavelet Transform is discussed with an overview of the Continuous Wavelet Transform (CWT), leading to its discrete time implementation, the Discrete Wavelet Transform (DWT), and the numerically efficient Fast Wavelet Transform (FWT). It will be seen that the FWT is built around the selection of a small set of lowpass filter coefficients which are used to form a mother wavelet function. Once these coefficients are determined, the FWT may be used to segment the time frequency plane in a way that allows for accurate time isolation of high frequency transients and fine resolution of low frequencies. This segmentation of the time frequency plane is analogous to the segmentation provided by sub-band speech encoding algorithms.

Chapter 4 provides a detailed overview of the Gaussian Mixture Model (GMM) pattern matching technique which has been applied to the text independent speaker identification problem. As with most pattern recognition techniques, it is assumed that a set of multidimensional feature vectors is available for both training and recognition. In the case of speaker identification, these vectors are derived from active frames of speech. They are modelled statistically using a probability density function (pdf) defined as the weighted summation of multivariate Gaussian pdfs where each

component density represents the distribution of feature vectors within a specific set of sounds, known as an *acoustic class*. The various GMM parameters are estimated from a set of feature vectors known to have originated from a specific speaker. All required estimation equations are discussed in Chapter 4. The GMM parameters estimated for a given speaker form the model for that particular speaker. Speaker models are then used to form a set of likelihood functions during the identification phase of the algorithms. The speaker corresponding to the highest likelihood functions is the most likely speaker for the analyzed speech frames.

Chapter 5 discusses the experimental portion of this thesis in which the speaker identification performance of five distinctly different feature vectors are examined. Two of these feature vectors are derived from standard cepstral coefficients. The remaining three are derived from a wavelet transform. All feature vectors are extracted from active frames of speech and the speech activity detection algorithm used to separate active and non active speech frames is also discussed. The feature vectors are tested in a closed set, text independent speaker identification problem which is implemented using speech data obtained from the SPIDRE speech corpus [5]. Active speech frames from specially selected training files are used to create feature vector sets and train GMMs for each of the forty five target speakers present in the SPIDRE corpus. Speaker identification accuracy is then assessed using the non-training files in order to ensure text independence. A brief discussion of the results obtained is presented along with some speculation as to why the wavelet based feature vectors failed to achieve satisfactory speaker identification performance.

Chapter 6 concludes the thesis and makes some recommendations for future work.

Chapter 2

Introduction To Speech And Speech Processing

2.1 The Speech Chain

The *speech chain* consists of the complete set of events which occur when one person uses speech in order to convey a message to another person [6]. As shown in Figure 2.1, the process begins with the conception of a message in the mind of the speaker. Following this, the sounds required to convey the message are generated in the speech production organs of the speaker. Although the basic function of speech production is to produce the sounds necessary to carry the linguistic message, collateral information related to the identity of the speaker, the mood of the speaker, the gender of the speaker, as well as the nationality of the speaker may be contained in the sounds generated. While the linguistic message is usually consciously encoded by the speaker, the collateral information contained in the speech signal is often unconsciously included, and may even be unintentional. Following production, the speech is transmitted as sound waves which propagate through the air as alternating condensations and rarefactions of the air [7]. These pressure variations are detected at the ear of the listener, where they are converted into electrical impulses which are carried by the auditory nerve to the listener's brain. The speech chain ends when these electrical impulses are decoded by the listener's brain, and are translated into a

Figure 2.1: The Speech Chain

linguistic message. Any collateral information included in the speech signal may also be decoded by the listener at this time.

Although the conceptual and perceptual links of the speech chain are not well understood, much effort is devoted to the artificial simulation of all processes involved in the speech chain. Conventional speech processing focuses on the artificial generation, transmission, and audition of speech using digital computers and electrical transmission techniques. Some aspects of the information perception stage have been examined as well, including word recognition and speaker recognition. Significant progress has been made in these areas in the past forty years [6, 8]. Artificial intelligence techniques continue to be developed which have potential for the simulation of the message conception and complete information perception stages of the speech chain [9, 10, 11, 12]. Simulation of these two links of the speech chain is generally referred to as *natural language processing*.

Automation of the speech chain is driven primarily by the desire of various industries to provide realistic interactive voice services over the public switched telephone network. Examples of such services include automated telephone operator assistance, automated banking transactions, automated airline reservation services, and the automatic retrieval of information such as weather reports and stock market quotations [13]. Some of these automated applications have already been successfully demonstrated. AT&T and Northern Telecom have both deployed limited vocabulary, speaker independent word recognition systems in order to automate the processing of collect calls. Since the vocabulary of these systems is limited, voice input by the user must be constrained to the few words requested in the prompts given. Work is ongoing to make these systems more flexible and responsive to natural unconstrained speech.

The personal computer software industry is also actively pursuing the automation of parts of the speech chain in an attempt to provide hands free data entry utilities for office administration, inventory control, and material handling applications [13]. Other automation applications include the development of artificial aids for people with speech and hearing impairments and the development of voice recognition tools for law enforcement and security applications.

2.2 Message Conception

An ironic fact, which was discovered in the early days of artificial intelligence research, is that those processes performed by human beings without significant conscious effort are the most difficult to simulate using computers [9]. Those processes which require significant human mental effort are generally easier to duplicate on a computer. One reason for this irony is that human beings tend to break up mentally complex tasks into a series of simple steps. Once the precise steps involved in a process have been defined it is usually easy to program them into a computer. Those processes which do not require significant mental effort are very difficult to break down into a series of computationally defined steps. The steps required to perform these tasks are learned at an early age, become second nature, and are impossible to describe. If it is unknown exactly how the human brain learns to perform these tasks, it is very difficult to program a computer to duplicate them. Examples of these processes include the cognitive and perceptual abilities which are developed early in life.

Message conception is another example of such a process. Artificial simulation of the message conception stage of the speech chain has been identified as one of the most difficult areas of natural language processing [9]. The basic steps involved in message conception are learned early in childhood. These steps become refined as the child's vocabulary is expanded, and rules of proper grammar are acquired. Although the refinement of the basic skill is a lifelong process, and the rules acquired during this refinement may be accurately described and programmed into a computer, message conception is largely an unconscious process which is learned early in life.

While the complete set of steps involved in the process of message conception is undefined, it is clear that a program simulating the process of message conception will be required to fulfil three main functions [9]. First, the program must decide the appropriate time to formulate a message. In most situations, the message will be formulated as the response to a user request. This would be the case for most automated service programs. In some advanced expert system applications however, the program may be required to volunteer information during the course of a session, without a specific prompt from the user.

The second function which must be performed by a program simulating message conception is the determination of the basic message content. The information requested by the user must be assembled and collated. If additional search parameters are required by the program, they must be requested from the user. This function is probably one of the easiest to implement, since data retrieval is a function ideally suited for computer implementation.

The third function which must be performed by a program simulating message conception is the detailed construction of the message. This is the most critical and difficult aspect of message conception since automatic implementation of this function requires knowledge of many of the undefined steps involved in natural message conception. The message ultimately constructed by the program must be grammatically and syntactically correct. Rules for grammar are well defined and it should be possible to program them into a computer. Syntax rules are somewhat less defined. The program must also resolve the fact that there are many equally correct ways of saying the same thing. If it is necessary to stress the importance of certain parts of the message, the program will have to determine which words in the message to emphasize. For advanced applications, the program may require the ability to tailor its messages to the level of experience of the user [9].

Most current automated service applications avoid the problems of automatic message conception by using either pre-recorded messages, a concatenation of pre-recorded phrases, or a predefined format for the generation of the message. As the demand increases for more realistic interactive voice applications, pre-recorded messages and phrases will not provide the flexibility required. Complete automation of telephone operator assistance, for example, could not be done using pre-recorded messages alone. The program would require the flexibility to construct unique messages in response to many non routine situations and requests.

2.3 Speech Production

Every spoken language is composed of a relatively small set of unique sounds which are grouped together to form all of the words which make up the language. These

sounds are called *phonemes*. The phonemes comprising the English language are shown in Table 2.1. It should be noted however that a phonetic description of any language is somewhat subjective and that the forty one phonemes shown in Table 2.1 are not the only symbols which may be used to transcribe the English language. Different transcription systems use anywhere between thirty nine [9] to fifty three [14] different symbols for the English language, depending on the exact dialect to be transcribed. Allophonic variation, which is discussed later in this section, also accounts for some differences between English phoneme tables. The symbols shown in Table 2.1 represent one of the commonly used systems for the phonetic transcription of English [6, 8].

By definition, a phoneme is the smallest meaningful, contrastive unit in a language [6]. The physical sound produced when a phoneme is uttered is called a *phone*. The subtle difference between a phoneme and a phone is that a phoneme is an abstract unit. A phone, on the other hand, is an actual implementation of a phoneme. It may be recorded, processed and quantitatively characterized. Since the human sound production mechanism is not a discrete system, and there is significant variation in the size and shape of speech production organs between individuals, an infinite number of phones can correspond to a single phoneme [6]. Stated another way, a phoneme symbolically represents the entire population ensemble of possible phones for a characteristic sound in a language.

The organs involved in the production of human speech are shown in Figure 2.2 [6]. These organs may be divided into three main groups: lungs, larynx, and vocal tract. The lungs provide the airflow necessary for the generation of speech. The larynx contains the vocal cords, or more appropriately called the vocal folds, which serve to modulate the airflow thereby creating sound. The vocal tract contains a variety of articulators which serve to modify the basic sounds produced by the vocal folds, and also to generate other sounds.

Almost all speech sounds are generated by air expired from the lungs. Such sounds are known as *egressive sounds*. Sounds generated during inhalation, such as clicking sounds, are called *ingressive sounds*. The English language, as in the case of most other languages, is composed exclusively of egressive sounds [6]. Vietnamese,

Phoneme Symbol	Manner Of Articulation Classification	Voiced/Unvoiced	Example Word
i	vowel	voiced	<u>beat</u>
I	vowel	voiced	<u>bit</u>
e (eI)	vowel (diphthong)	voiced	<u>bait</u>
ε	vowel	voiced	<u>bet</u>
æ	vowel	voiced	<u>bat</u>
α	vowel	voiced	<u>hot</u>
ɔ	vowel	voiced	<u>caught</u>
o (oU)	vowel (diphthong)	voiced	<u>boat</u>
U	vowel	voiced	<u>book</u>
u	vowel	voiced	<u>boot</u>
ʌ	vowel	voiced	<u>but</u>
ɜ	vowel	voiced	<u>bird</u>
ə	vowel	voiced	<u>about</u>
αI	diphthong	voiced	<u>buy</u>
ɔI	diphthong	voiced	<u>boy</u>
αU	diphthong	voiced	<u>bout</u>
ju	diphthong	voiced	<u>cued</u>
j	glide	voiced	<u>you</u>
w	glide	voiced	<u>wow</u>
l	liquid	voiced	<u>lull</u>
r	liquid	voiced	<u>roar</u>
m	nasal	voiced	<u>maim</u>
n	nasal	voiced	<u>none</u>
ŋ	nasal	voiced	<u>bang</u>
f	fricative	unvoiced	<u>fluff</u>
v	fricative	voiced	<u>valve</u>
θ	fricative	unvoiced	<u>thin</u>
ð	fricative	voiced	<u>then</u>
s	fricative	unvoiced	<u>sass</u>
z	fricative	voiced	<u>zoos</u>
ʃ	fricative	unvoiced	<u>shoe</u>
ʒ	fricative	voiced	<u>measure</u>
h	fricative	unvoiced	<u>how</u>
p	stop	unvoiced	<u>pop</u>
b	stop	voiced	<u>bob</u>
t	stop	unvoiced	<u>tot</u>
d	stop	voiced	<u>did</u>
k	stop	unvoiced	<u>kick</u>
g	stop	voiced	<u>gig</u>
tʃ	affricate	unvoiced	<u>church</u>
dʒ	affricate	voiced	<u>judge</u>

Table 2.1: Phonemes Of The English Language

Note: Figure Taken From [6]

Figure 2.2: Speech Production Organs

Zulu, Sindi (a language spoken in India and Pakistan), Uduk (a language spoken in Southern Sudan), and some Native American languages do contain ingressive sounds [15].

Expired air from the lungs passes via the trachea (windpipe) through the larynx. The larynx, which is illustrated in Figure 2.3 [6], is composed of various cartilages, membranes, tendons, and muscles. As air is forced through the larynx, it must pass between two small muscular folds called the *vocal folds* [15]. The space between these folds is called the *glottis*. During normal breathing, the vocal folds remain completely separated to allow unrestricted airflow. During most speech, the glottis is narrowed or closed completely. As air is expired, the air pressure increases around the glottis if the opening is sufficiently small. This air pressure builds to a point where it forces the vocal folds apart and a puff of air is rapidly released through the glottis into the vocal tract. The sudden, rapid flow of air through the glottis decreases the air pressure between the vocal folds. The reduction of air pressure, combined with the elasticity of the vocal folds, allows the folds to once again narrow the glottal opening causing the air pressure to again build. This repeating pattern of Bernoulli forces acting on the vocal folds causes periodic bursts of air to be released into the vocal tract, resulting in a periodic sound wave emanated from the mouth. Speech generated as a result of vocal fold vibration is known as *voiced* speech. Spectral analysis of the sound waveform occurring during voiced speech reveals that it contains energy at multiples of the rate with which the vocal cords are opened and closed. This frequency is known as the *fundamental frequency* or F_0 frequency of the signal. It is also sometimes called the *pitch* of the voice signal, although strictly speaking pitch is an acoustic phenomena of a sound which enables a listener to perceptually place it on a scale going from low to high. It is related to the fundamental frequency of a voice signal in that a voice signal with a low fundamental frequency is perceived as a low pitched signal. A voice signal with a higher fundamental frequency is perceived as a higher pitched signal. The relationship between fundamental frequency and perceived pitch is not linear however [15]. Since the periodic bursts are not ideal impulses, there is a natural roll off to the excitation spectrum. The typical roll off has been found to be approximately 6 dB/octave [6].

Figure 2.3: Cross Sectional View Of Larynx

The normal fundamental frequency in male speakers varies from approximately 50 to 250 Hz, with a mean value of approximately 130 Hz. Since the vocal cords in females are generally smaller than those of males, the fundamental frequency for female voiced speech is generally higher than that of males, having a range between 120 Hz and 500 Hz with a mean value of approximately 260 Hz [16]. All speakers have the ability to vary their normal F_0 frequency. This is accomplished by using muscles in the larynx to alter the length and tension of the vocal folds. Variation of F_0 frequency occurs often during the course of normal conversation. As a sentence is uttered, the F_0 frequency of a speaker will vary about a nominal value. Whether the variation is above or below the speaker's nominal value depends on the context of the spoken sentence. For example, questions often end with a slight increase in the F_0 frequency of the speaker. The statement "*His name is Peter.*" differs from the question "*His name is Peter?*" solely by a rise in F_0 frequency at the end of the sentence [15]. F_0 (or pitch) variation during normal conversation is known as *intonation*.

Most speech is voiced, however not all speech is generated as a result of vocal cord vibration. *Unvoiced* or *voiceless* speech occurs when the glottis is not sufficiently narrow to allow the build up of air pressure required for the initiation of vocal fold vibration. The turbulence created by the air forced through the glottis creates a noise like sound which is subsequently modified by the vocal tract. A good example illustrating the difference between a voiced and unvoiced sound is the difference between the /v/ sound in the word *vowel* and the /f/ sound in the word *foul* [15]. The former is a voiced sound and the latter is unvoiced. Altering between a prolonged /v/ sound and a prolonged /f/ sound reveals that the shape of the vocal tract is the same in both cases. In the case of the /v/ sound, the vocal folds vibrate. This may be verified by placing one's fingertips against the larynx during the utterance. In the case of the /f/ sound, there is no vibration. A second example of a voiced sound is the /o/ sound in the word *hot*. A prolonged /o/ utterance reveals the characteristic vocal cord vibration. If whispered however, the same sound is unvoiced.

Although the vocal cords are responsible for the generation of a basic (either periodic or noise-like) sound wave, it is the vocal tract which modifies this basic sound

in order to produce the many different sounds characterizing a spoken language. The oral and nasal passages act as resonant cavities amplifying certain frequencies and attenuating others. The frequencies of vocal tract resonance are known as *formant frequencies*. Formant frequencies are generally abbreviated as F_i , where F_1 designates the lowest formant frequency, F_2 designates the next higher frequency, etc. The effect of the vocal tract on the sound wave produced by the glottis is shown in Figure 2.4.

The articulators of the vocal tract, which are shown in Figure 2.2, serve to vary the shape of the oral and nasal cavities thereby altering the frequency response of the vocal tract. Since these articulators are varied constantly during a conversation, the vocal tract forms a time varying filter for the glottal excitation. The position and interaction of these articulators during the generation of a specific speech sound is known as the *manner of articulation* for that sound. As discussed earlier, speech sounds may be classed as either voiced or unvoiced. They may also be further subdivided into groups based on the manner of articulation. This leads to eight groups of sounds: vowels, diphthongs, fricatives, glides, liquids, nasals, stops, and affricates. Examples of each class of sound are shown in Table 2.1.

Vowels are produced by exciting a fixed vocal tract with a periodic excitation produced by vocal cord vibration. *Diphthongs* are produced by varying the vocal tract smoothly from one vowel position to another, while maintaining the vocal cord excitation. Classification of diphthongs is not standard and often depends on the exact dialect of the language spoken. In Table 2.1, the phonemes /e/, as in the word *bait*, and /o/, as in the word *boat*, are classed as vowels. In some English dialects however these phonemes may be pronounced as the smooth transition between two separate vowel sounds. The phoneme /e/, for example, is sometimes pronounced /eI/. Similarly, the phoneme /o/ may be pronounced as /oU/. In these cases, the sounds are classed as diphthongs [8].

In the case of vowels and diphthongs, the vocal tract has a fairly large opening. *Fricative* sounds are produced by forcing the glottal excitation through a narrow vocal tract opening. They may be either voiced or unvoiced. Forcing the glottal excitation through a narrow opening creates air turbulence at the constriction. This turbulence generates a waveform which is generally richer in higher frequency components than

Note: Figure Taken From [2]

Figure 2.4: Effect Of Vocal Tract On Glottal Waveform

in the case of vowels or diphthongs. *Glides* are vowel-like sounds in which the vocal tract has a narrower constriction than it does for the generation of a vowel sound. Under conditions of unusually strong airflow, as in the case of shouting for example, glides may produce fricative-like sounds due to the air turbulence created at the constriction [6]. *Liquids* are also similar to vowels, but are unique in that the air flow is deflected around the tip or dorsum of the tongue. Both glides and liquid sounds are voiced. They are sometime collectively referred to as *semi-vowel* sounds. *Nasals* are voiced sounds which are produced with the vocal tract totally restricted at some point in the oral cavity. The velum is lowered so that air flows through the nasal cavity with sound being radiated at the nostrils [8].

Stops or *plosive* sounds are generated by a total vocal tract constriction. Air pressure is built up behind the constriction, and sound is generated when the constriction is opened and the air pressure is released. During the period when there is a total constriction of the vocal tract, no sound emerges from the mouth, although in the case of a voiced stop such as the /g/ sound in the word *gig*, there may be some low frequency energy radiated through the walls of the vocal tract. This radiation occurs during the vocal cord vibration which takes place as pressure is built up at the constriction. The sound created during the release of the vocal tract constriction is a transient sound, which generally lasts approximately 10 msec [6].

The last class of sound indicated in Table 2.1 are the *affricates*. These sounds are the concatenation of a stop sound, followed by a fricative sound. There are two affricate sounds in the English language. The /č/ phoneme is the concatenation of the sounds /t/ and /ʃ/. The /ž/ affricate is the concatenation of the sounds /d/ and /ʒ/. The classification of affricates is rather arbitrary since combinations of stops followed by fricatives occur frequently in English, but are not classed as separate phonemes. The end of the word *cats*, for example, contains the stop /t/ followed by the fricative /s/. Pluralization of many English nouns results in similar combinations. The affricates indicated in Table 2.1 are unique in that they are the only stop-fricative combinations that may occur either at the start or end of English words. The fricatives used in the phonemes /č/ and /ž/ also have the characteristic that their respective places of articulation in the vocal tract are the same as the

place of articulation for the preceding stop [15]. Both the /t/ and /ʃ/ phonemes in the affricate /tʃ/, for example, are articulated by using the tongue to create a constriction just behind the incisor teeth. The same is not true of the stop-fricative combination in the word *cats*. The stop /t/ is articulated by using the tongue, but the fricative /s/ is articulated at the constriction formed solely by the incisor teeth. Since the stop-fricative combinations in affricates share a common place of articulation, the concatenation of sounds is generally more rapid than non-affricate stop-fricative combinations. For these reasons, /tʃ/ and /ʒ/ are classed as distinct phonemes in Table 2.1.

Vowels, diphthongs, glides, liquids, and nasals are sometimes collectively referred to as *sonorant* sounds. Fricatives, stops, and affricates are collectively referred to as *obstruent* sounds. Sonorant sounds have the characteristics of being exclusively voiced, periodic, and fairly intense sounds. Obstruent sounds, on the other hand, may be either voiced or unvoiced. In fact, as may be seen from Table 2.1, voiced obstruent sounds always have an unvoiced counterpart having an identical voice tract configuration. Obstruent phonemes are also generated with a narrower vocal tract opening than in the case of sonorants. They are therefore not as intense as sonorants. Finally, obstruent sounds are generally richer in high frequency components than sonorants.

Artificial simulation of speech production is an area of signal processing known as *Speech Synthesis*. Unlike the message conception link of the speech chain, the state of the art of speech synthesis is relatively advanced. Speech synthesizers are divided into two main groups: voice response systems, and unrestricted text to speech systems. Both classes of system commonly consist of stored speech units and a set of rules for the concatenation of these speech units [6].

Voice response systems are the most common type of synthesizer seen in commercial applications. They are used in automatic information retrieval systems, as well as telephone directory assistance applications. In many Canadian directory assistance systems, once the operator has found the requested telephone number a computer will often be tasked to read the number over the phone to the customer in both official languages. Human operator interaction is required only during the initial query.

Voice response systems produce limited vocabulary sentences from the concatenation of stored words and phrases. The words are generally stored using a high precision encoding or recording of the waveforms for the words and phrases in the synthesizer vocabulary. This allows high quality intelligible sentences to be created. Unfortunately a common limitation of voice response systems is that they generally have a small vocabulary. High precision storage of a large vocabulary requires a large amount of memory. A second limitation of voice response systems is that although the speech is intelligible, it often sounds unnatural. This is because words which are spoken and recorded in isolation do not have the proper intensity, duration or intonation when used in the context of different sentences. Intensity, duration and intonation are often referred to collectively as the *prosodic features* of the utterance. Improper adjustment of the prosodic features of words recorded in isolation and then concatenated to form phrases is one of the main reasons for the “synthetic” sound of many voice response systems [17]. This problem is noticeable in many voice response systems including Carleton University’s Touch Tone Information system, which provides synthesized voice prompts for course registration and admissions information. A simple solution to the problem of poor prosody for recorded words is to record the words in the synthesizer vocabulary several times, with each recording of the word used in a slightly different context. This solution however increases the memory requirements for the synthesizer, and also increases the complexity since rules for the use of various forms of the same word must be devised.

Unrestricted text to speech synthesizers attempt to produce unlimited vocabulary sentences from text supplied by a system controller. Memory requirements are minimized by storing units of speech which are smaller than words. These synthesizers are usually based on the model shown in Figure 2.5. The excitation is chosen to be either a periodic impulse like function for voiced speech intervals, or a white noise function for unvoiced speech intervals. The excitation is applied to a filter, the coefficients of which are chosen to correspond to the vocal tract configuration for a particular phoneme shown in Table 2.1. By varying these filter coefficients and the excitation parameters in a controlled manner, unrestricted speech may be generated.

Typically, each vector stored in the synthesizer memory contains 10 - 20 filter

Figure 2.5: Speech Synthesizer Model

coefficients as well as a bit indicating the excitation for the phoneme modelled. A few extra bits are also stored with each vector indicating the F_0 frequency to be used if the excitation is voiced. Most phonemes in Table 2.1 require a single vector, although diphthongs require two. As a result, complete representation of all phonemes shown in Table 2.1 would require the storage of approximately 41-47 vectors. The storage of a special vector to indicate a small interval of silence is also required in order to allow unrestricted construction of all phrases in the English language. A few variants of each phoneme must be stored in order to allow for proper intonational representation. A basic phoneme based text to speech synthesizer would therefore require less than 500 stored vectors, if ten variants of each phoneme is assumed.

One obvious difficulty with the design of phoneme based text to speech converters is that rules must be established for the formation of words from the concatenation of phonemes. A second (not so obvious) difficulty with the design of realistic phoneme based synthesizers relates to the fact that in human speech the exact pronunciation of individual phonemes in a word is highly dependent on the neighboring phonemes in the word. The variation of phonetic pronunciation is caused by the fact that an articulator which is not involved in the production of an immediate phoneme will nearly always start moving toward the position required for the following phoneme, if it is the primary articulator for the production of the latter phoneme [15]. This effect is known as *coarticulation*, and causes a slight overlapping of adjacent articulations in a sentence.

The effects of coarticulation are especially noticeable in the English language during the articulation of consonants which precede vowels. The anticipated vowel will often affect the exact manner of articulation for the consonant. A good example of this effect is illustrated by the articulation of the phoneme /k/ in the words *key* (/k/-/i/) and *caw* (/k/-/ɔ/) [15]. In the word *key*, the /k/ phoneme is articulated with the tongue raised, making contact at about the middle of the roof of the mouth. The mouth is also widened in a smiling motion, in anticipation of the following /i/. In the word *caw* however, the /k/ is articulated with the tongue making contact near the back of the roof of the mouth. The lips are also more rounded than in the case of the word *key*. This is in anticipation of the following /ɔ/ phoneme. The two

Figure 2.6: Power Spectral Densities Of /k/ Allophones

articulations of the phoneme *k* are known as *allophones*, since they represent the same phoneme, but are articulated differently and have a slightly different sound. Forcing pronunciation of the /*k*/ phoneme in the word *key* with the lips slightly rounded and the tongue making contact near the back of the roof of the mouth, although awkward, does not change the meaning of the word, and may not even be noticed in a normal conversation. The difference in the sound of the two /*k*/ allophones is difficult to quantify, however spectral analysis of the two sounds reveals that in the case of the word *key* the /*k*/ sound contains more energy at higher frequencies than in the case of the word *caw*. This is illustrated in the power spectral density plots shown in Figure 2.6.

It is interesting to note that sounds classed as allophones in one language may correspond to totally different phonemes in other language. An example of this are the various allophones of the phoneme /*t*/ . In English, it normally does not matter if the /*t*/ is articulated with the tongue just behind the incisor teeth, as in the word *tee*, or is articulated with the tongue making contact with the roof of the mouth further back as in the word *ton*. In some Indian languages however, these different sounds are classed as different phonemes, which change the meanings of the articulated words if they are interchanged. A second, more dramatic, example of allophones are the sounds /*l*/, as in the word *lap* and the sound /*r*/, as in the word *rap*. In English, these are very unique phonemes. In Japanese on the other hand, they are merely acceptable allophones of the same phoneme [17]

Realistic modelling of allophones and coarticulation in speech synthesizers is difficult. Some synthesizers employ spectral smoothing at the boundaries between adjacent phonemes which simulates the effect of coarticulation. Other synthesizers avoid the problem of modelling coarticulation by storing units of speech known as *diphones*. These units are obtained by combining phonemes together and storing them in pairwise combinations. The units are stored with cuts in the middle of each phoneme. This preserves the coarticulation effects, which typically occur during the transition between adjacent phonemes.

Theoretically, a diphone based synthesizer would require the storage of approximately 1764 diphones [6]. (All pairwise combinations of 42 x 42 phonemes and

silence.) In practice however, not all combinations are used in English. Only about 1200 diphones are necessary. Since diphones are larger than phonemes, each one requires the storage of approximately 3-5 coefficient vectors. A diphone based synthesizer would therefore require storage capacity for approximately 6000 coefficient vectors.

The transcription of the word *Carleton* using phonemes and diphones is illustrated in Figure 2.7. This figure also illustrates another consideration in the synthesis of speech; natural speech involves considerable variations in intensity. As discussed previously, sonorant phonemes such as the /l/ are generally more intense than obstruent phonemes such as the /k/. Figure 2.7 also reveals however that intensity variation may occur even during the utterance of a single phoneme, as in the case of the /l/ and /n/ phonemes. Rules for intensity variation during the utterance of a phoneme or diphone must also be programmed into a speech synthesizer, in addition to the rules for other prosodic features.

In addition to phonemes and diphones, the storage of other speech units such as word syllables or demi-syllables in speech synthesizers is also possible [6]. With each unit however the memory requirements of the synthesizer increase since more units are required to represent all words in the English language. The complexity of the synthesizer also increases since more complicated rules are required for the concatenation of the speech units, and for the adjustment of the prosodic features during the utterance of the stored speech unit.

2.4 Speech Transmission

As discussed in Section 2.1, the most common mechanism for the transmission of speech is via sound waves transmitted through the air. It is well known that sound waves may be artificially generated by mechanical speakers. This mechanism for simulating the speech transmission link of the speech chain has been practiced for many years in the design of commercial audio equipment. Most current work in the artificial transmission of speech involves the digital coding of the speech waveform for efficient long distance transmission over electrical or radio frequency communications

Figure 2.7: Transcription Of Word *Carleton* Using Phonemes and Diphones

links.

The coding of a waveform for transmission usually involves a trade-off between the quality of the received signal and the bandwidth (or data rate) of the channel required for the transmission of the signal. Other encoder design considerations include the complexity of the design, the robustness of the encoder in the presence of channel transmission errors, and for full duplex voice communications, the total processing delay. A long delay between the time a speaker utters a sound, and the time the encoded bits are derived and transmitted may be objectionable during two way communications, especially when this encoder delay is added to transmission delays already present in some systems [18]. Ideally, the performance of an encoder should also be independent of the input signal type. For example, an encoder should represent music and voice signals equally well. In practice however, performance concessions are often made in an effort to reduce channel bandwidth. As a result, some encoder algorithms may be satisfactory for voice signals, but not music. Other encoder algorithms may perform well for the transmission of male voices, but not female voices.

For analog systems, the channel bandwidth required for transmission is proportional to the bandwidth of the voice signal. Although voice signals, especially during unvoiced speech intervals, can contain frequency components over 8 kHz [8], most energy required for intelligible speech is contained in a band of frequencies between 200 Hz and 3.5 kHz [19]. For telecommunications systems, a bandwidth of 4 kHz is usually allocated for a voice grade channel. This bandwidth has been found to be sufficient for the intelligible transmission of the phonemes and is also sufficient for the preservation of the first three formant frequencies for most speakers. These formants are important for the preservation of speaker identification information [8].

For SSB transmission of a 4 kHz voice grade channel a bandwidth of 4 kHz is required. For AM transmissions, the required bandwidth is 8 kHz, and for FM transmissions the required bandwidth is approximately 10 - 30 kHz, depending on the peak frequency deviation chosen. A larger peak frequency deviation allows a higher signal to noise ratio at the receiver for a given transmitted power. Generally, a peak frequency deviation of 4 - 12 kHz is chosen for mobile communications applications

[20].

For digital transmissions the required channel bandwidth depends on the data rate of the digitally encoded voice signal. As will be seen, this data rate is either directly or indirectly dependent on the bandwidth of the baseband voice signal. The channel bandwidth required to transmit a digital waveform of data rate R bits per second (bps) is dependent on the data encoding technique (NRZ, RZ or Manchester encoding), and the modulation scheme chosen. A figure of merit for the spectral efficiency of a digital transmission is given by the bandwidth efficiency parameter of the modulation technique, which is defined as [21]

$$\beta = R/B_c, \quad (2.1)$$

where R is the data rate of the bit stream and

B_c is the channel bandwidth.

For direct baseband transmission of a digital waveform, a corollary to Nyquist's Sampling Theorem states that a minimum bandwidth of $R/2$ Hz is required to minimize the effects of intersymbol interference [21]. The ideal bandwidth efficiency of baseband digital transmission is therefore 2 bps/Hz. In practice however, due to the effects of non ideal filtering, the efficiency of baseband transmission usually degrades to approximately 1 bps/Hz.

In the case of a modulated signal, ideal bandwidth efficiencies for digital transmission may be improved over those of baseband transmission through the use of a multi-level modulation technique such as m -ary PSK which has a bandwidth efficiency of [21]

$$\beta = R/B_c = \log_2 m \text{ bps/Hz}. \quad (2.2)$$

where m is the number symbols used in the modulation.

Regardless of the modulation technique employed, a reduction in the data rate of an encoded voice signal results in a reduction of the channel bandwidth required for transmission. The goal of speech encoder research is to digitally encode the speech

signal using as low a data rate as possible, while preserving the quality of the speech. In English speech, phonemes occur at a rate of approximately 12 to 14 per second [6]. As shown in Table 2.1, there are 41 phonemes in the English language. If a silence interval is included in the list of symbols, and each symbol is assumed to be equally likely, then at least six information bits are required to uniquely represent each phoneme since $2^6 = 64$, which is greater than the required 42 symbols. Transmitting six bits of information at a rate of 14 times per second corresponds to a data rate of 84 bits per second. This would indicate that the theoretical minimum data rate required for encoding the text of English speech for transmission is less than 100 bits per second. In practice however, the speech encoder must often also encode the prosodic features of the utterance in order to get a realistic representation of the speech, and in some cases may be required to encode speaker dependent information such as fundamental frequency and formant frequencies. This increases the required data rate of the encoder algorithm.

Speech encoder algorithm design is closely related to the design of parameter extraction algorithms for speech recognition systems since the parameters used to represent a frame of speech are often the same. Some of the more common parameter extraction algorithms will be discussed in Section 2.5. Speech encoder design is also related to speech synthesizer design since a low data rate speech encoding algorithm often leads to memory efficient vector coefficients for the storage of vocal tract configuration models representing the various speech units employed in the synthesizer. Both synthesizers and encoders are required to represent verbal text using a set of digitally encoded vectors. The main difference is that a voice encoder is often required to code the residual information related to speaker identity. Preservation of speaker dependent characteristics is important for the transmission of voice signals in telecommunications systems. A synthesizer, on the other hand, is rarely required to simulate the speaker dependent qualities of different speakers and need only represent the verbal text in a natural, realistic manner. Storage of additional coefficients such as formant frequencies for speaker dependency information is not required.

The range of data rates and associated quality for standard digital speech encoders is given in Figure 2.8 [22]. Broadcast quality speech refers to wide (7 kHz)

Note: Figure Taken From [22]

Figure 2.8: Spectrum Of Speech Coding Transmission Rates And Associated Quality

bandwidth signals which have been encoded with no perceptible noise. Toll quality is a standard for speech signals transmitted over the wireline portion of the public switched telephone network. These signals generally have a signal to noise ratio better than 30 dB, with less than 2-3% harmonic distortion. Communications quality speech is intelligible, but does have more distortion than toll quality signals. It is generally an acceptable standard for cellular telephone communications. Synthetic quality speech is 80-90% intelligible, but sounds noticeably machine like. As a result, speaker identification becomes very difficult [6]. In addition to the proper transmission of speech signals, toll and communications quality transmissions also often have the requirement to transmit audio frequency modem signals with an acceptable bit error rate. An encoder design based exclusively on the statistics of a speech signal may not properly encode a modem signal, which has a different statistical characteristic [23].

As may be seen from Figure 2.8, speech encoders are commonly divided into three main groups: waveform coders, source encoders and hybrid encoders. Each class of encoder is discussed below.

2.4.1 Waveform Encoders

Waveform coders strive for a facsimile reproduction of the input signal [22]. They make few assumptions about the nature of the input signal and are based largely on selective quantization techniques. Ideally, waveform coders are able to represent all types of signals such as voice, music, or voice band modem tones with equal fidelity. The higher data rate waveform encoders are indeed capable of this, but in practice some assumptions about the nature of the input signal statistics are often made in order to reduce the data rate of the encoder. Some of the more common waveform encoding techniques are illustrated in Figures 2.9 and 2.10 [18]. The simplest waveform encoder consists of a sampler, followed by a uniform quantizer and a binary encoder. These blocks are commonly grouped together as an Analog to Digital Converter (ADC). The reverse process, known as Digital to Analog Conversion, involves a binary decoder and a low pass filter.

During the encoding process of simple A/D conversion, which is also called Pulse

Figure 2.9: Waveform Quantization Techniques

Figure 2.10: More Waveform Quantization Techniques

Code Modulation (PCM), the input signal is sampled at a rate greater than or equal to twice its bandwidth. The data rate is given by

$$R = f_s \times b, \quad (2.3)$$

where R is the encoder data rate,

f_s is the sampling frequency, and

b is the number of bits/sample.

The number of bits per sample is determined by the quality required for the encoded signal. The quantization of a signal into discrete amplitude levels introduces a certain amount of error which may be represented as additive noise as follows:

$$\hat{s}(n) = s(n) + q_e(n) \quad (2.4)$$

where $\hat{s}(n)$ is the value of the quantized signal sample,

$s(n)$ is the value of the unquantized signal sample, and

$q_e(n)$ is the quantization error.

The signal to quantization noise ratio of a quantizer is given by the ratio of the input signal variance to the variance of the quantization error. This parameter is often expressed in decibels as follows:

$$SNR_Q = 10.0 \log\left(\frac{\sigma_s^2}{\sigma_{q_e}^2}\right) \quad (2.5)$$

where SNR_Q is the Signal To Quantization Noise Ratio expressed in dB,

σ_s^2 is the variance of the quantizer input signal, and

$\sigma_{q_e}^2$ is the variance of the quantization error.

In order to obtain an expression of SNR_Q in terms of the input signal parameters, and the number of bits used in the quantizer, it is necessary to make an assumption about the statistics of the input signal. In Section 2.5, it will be shown that speech may be modelled approximately by a Laplacian distribution. If this type of distribution is assumed for the signal input to a uniform quantizer, the SNR_Q of the

Figure 2.11: Quantizer SNR For Uniform and μ Law Quantizers

quantized signal may be derived following the procedure given in reference [24]. For a uniform midtread quantizer, it is given by

$$SNR_Q = -10.0 \log_{10} \left[1 - \frac{\sqrt{2}}{N_Q - 1} \left(\frac{V_L}{\sigma_s} \right) - \frac{2\sqrt{2}}{N_Q - 1} \left(\frac{V_L}{\sigma_s} \right) \sum_{p=1}^{\frac{N_Q}{2}-1} e^{-\frac{2\sqrt{2}}{N_Q-1} \left(\frac{V_L}{\sigma_s} \right) p} \right], \quad (2.6)$$

where V_L is the clipping voltage for the quantizer,

σ_s^2 is the variance of the quantizer input signal, and

N_Q is the number of quantization levels (2^b).

This expression is plotted in Figure 2.11 as a function of the ratio $\frac{\sigma_s^2}{V_L^2}$, for both an eight bit and a ten bit uniform quantizer. It is immediately obvious that the SNR_Q is highly dependent on the power (σ_s^2) of the input signal. If the power is too high, then the clipping error introduced by the quantizer dominates the quantization error and reduces the SNR_Q . If the signal power is too low, the granularity of the quantizer becomes significant and dominates the quantization error. The optimum SNR_Q occurs only over a narrow range of signal power. In normal speech however, signal power variations of 20 - 30 dB may occur between voiced and unvoiced speech intervals. Intra-speaker variance variations can be as much as 40 dB [22]. Assuming that a toll quality SNR of at least 30 dB must be maintained over an input dynamic range of approximately 20 dB, it is clear from Figure 2.11 that an eight bit uniform quantizer is not sufficient. A ten bit quantizer is necessary which, from Equation 2.3, implies that an encoder data rate of 80 kbits/second would be required for an 8.0 kHz sampling rate.

Normally, it is desirable for the SNR_Q to be relatively independent of the power level of the input signal. This is accomplished by using a nonlinear quantization function, where the input signal is applied to a nonlinear transfer function prior to uniform quantization. A commonly used transfer function is that of the μ -law compression function which is given by [8]

$$s_c(n) = V_L \left(\frac{\ln \left(1 + \frac{\mu s(n)}{V_L} \right)}{\ln(1 + \mu)} \right) \quad \text{for } 0 \leq s(n) \leq V_L \text{ and.}$$

(2.7)

$$= -V_L \left(\frac{\ln \left(1 - \frac{\mu s(n)}{V_L} \right)}{\ln(1 + \mu)} \right) \text{ for } -V_L \leq s(n) < 0,$$

where V_L is the clipping voltage for the quantizer,

$s_c(n)$ is the output of the compression function and,

$s(n)$ is the value of the input signal sample.

The signal to quantization noise ratio of a Laplacian distributed input signal quantized by a μ -law quantizer is given by the following equation:

$$SNR_Q = -10.0 \log \left(\left[\left(\frac{V_L}{(N_Q - 1)\sigma_s} \right)^2 + \sqrt{2} \frac{V_L}{(N_Q - 1)\sigma_s} + 1 \right] e^{-\frac{\sqrt{2}V_L}{\sigma_s}} + \eta \left[\left(\frac{V_L}{\sigma_s} \right)^2 + \sqrt{2}\mu \frac{V_L}{\sigma_s} + \mu^2 - \left[\left(\frac{(1 + \mu)V_L}{\sigma_s} \right)^2 + \sqrt{2} \frac{(1 + \mu)V_L}{\sigma_s} + \mu^2 \right] e^{-\frac{\sqrt{2}V_L}{\sigma_s}} \right] \right) \quad (2.8)$$

where $\eta = \frac{\ln^2(1 + \mu)}{3\mu^2(N_Q - 1)^2}$,

V_L is the clipping voltage for the quantizer,

σ_s^2 is the variance of the quantizer input signal, and

N_Q is the number of quantization levels (2^b).

This expression is plotted in Figure 2.11 for an eight bit μ -law quantizer with $\mu = 255$. This type of quantizer is commonly used in North American telecommunications systems for encoding speech signals. As may be seen, the eight bit μ -law quantizer provides an SNR_Q of at least 30 dB over an input dynamic range of approximately 35 dB. Not only does the eight bit μ -law quantization provide a wider dynamic range than the ten bit uniform quantizer, it also reduces the data rate required for toll quality from 80 kbits/sec to 64 kbits/sec.

Both linear and non-linear PCM encoders suffer from the effects of fixed step quantization. As mentioned previously, in any quantization scheme there is a trade-off involved between clipping noise and granular noise. If the step size is chosen too small, there are not enough quantization levels to represent large power signals. In

this case, the clipping error dominates the quantization noise. If the step size is chosen too large, the granularity of the quantization is not fine enough to represent small power intervals of the signal. In this case, the granularity dominates the quantization noise. Non-linear quantization techniques reduce this effect by assigning small quantization levels to small signal amplitudes, and larger quantization levels to larger signal amplitudes. If however, there are not enough bits to allow a sufficient range of quantization step sizes, even non-linear quantizers suffer from SNR limitations. One solution to this problem is to dynamically assign a step size based on a short time estimation of the input signal variance. Depending on whether the input variance estimate is made before or after the quantizer, the quantization technique is known as Adaptive Quantization with Forward estimation (AQF) or Adaptive Quantization with Backward estimation (AQB) respectively. Adaptive quantization reduces the number of bits required to represent the input signal with an acceptable SNR. For a given data rate, the application of forward or backward adaptive quantization can provide a 4 - 7 dB improvement over μ -law encoding [8]. A general rule of thumb which applies to all waveform encoders is that SNR is roughly proportional to 6.0 dB times the number of quantization bits per sample. An SNR improvement of 6.0 dB therefore implies that a voice signal may be encoded using one less bit per sample with an adaptive quantization technique in order to maintain a given SNR. A small reduction in data rate is therefore possible with adaptive quantization. In the case of AQF, some additional overhead is required in order to transmit the quantization step level information. AQB does not require the extra bits for transmitting the step size information, but does suffer from the limitation that the variance estimation of the input signal is based on quantized signals. The quantization error therefore tends to corrupt the variance estimate [24].

Another technique which is often employed in waveform encoders to reduce the data rate for the transmission of an acceptable quality signal is that of Differential Pulse Code Modulation (DPCM). As shown in Figure 2.9, DPCM involves the application of a linear prediction filter to form an estimate of the input signal sample for any given time. The prediction is formed by a linear combination of previously

quantized input signal samples as follows:

$$\tilde{s}(n) = \sum_{l=1}^L a_l \hat{s}(n-l) \quad (2.9)$$

where $\tilde{s}(n)$ is the prediction of the signal sample at time n ,

L is the order of the prediction filter,

a_l for $l = 1$ to L are the prediction filter coefficients.

$\hat{s}(n-l)$ for $k = 1$ to L are previously quantized samples of the input signal.

The linear prediction of the sample value, is then subtracted from the actual input signal sample to form the following error signal:

$$e(t) = s(t) - \tilde{s}(t). \quad (2.10)$$

Since there is a high degree of correlation between adjacent samples of the speech signal, the value of $\tilde{s}(n)$ is reasonably close to the actual value of $s(n)$. As a result, the variance of the error signal $e(n)$ is much smaller than the variance of the input signal $s(n)$. This allows the error signal to be quantized using fewer bits, while maintaining a desired SNR for the reconstructed signal.

Using equations 2.4 and 2.10 the quantized error signal may be shown to be equal to [8]

$$\hat{e}(n) = s(n) - \tilde{s}(n) + q_e(n). \quad (2.11)$$

From this, it can be seen that the input to the prediction filter at time n is given by:

$$\hat{e}(n) + \tilde{s}(n) = s(n) + q_e(n) = \hat{s}(n), \quad (2.12)$$

which is the quantized value of the input signal $s(n)$. The prediction filter therefore simply acts to filter the quantization noise from $s(n)$. To reconstruct the encoded signal, the quantized error signal is applied to an identical prediction filter as shown in Figure 2.9. This forms $\hat{s}(n)$, which corresponds to a quantized form of $s(n)$, but the quantization error term $q_e(n)$ is based not on a direct quantization of $s(n)$ but on a quantization of the lower variance error signal.

The quantizer employed in DPCM may be either uniform or nonuniform. For a given number of quantization bits, the application of DPCM results in an SNR_Q advantage of 6-8 dB obtained over μ -law PCM, depending on the order of the prediction

filter used [24]. DPCM may also employ AQF or AQB techniques for a further gain in SNR_Q (or corresponding reduction in data rate). Adaptation in DPCM encoders may also be applied to the estimation filter in an attempt to adaptively minimize the mean squared error between the estimated signal value $\tilde{s}(n)$ and the actual value $s(n)$. This yields a further improvement in SNR_Q . The result of this adaptation is that the predictor filter coefficients will vary according to the variance of the input signal, and will also depend on whether the speech is voiced or unvoiced, since the statistics of each type of signal are different. Adaptation of the prediction filter coefficients is known as Adaptive Prediction Coding (APC). One encoding standard for DPCM that has been adopted by the International Telephone and Telegraph Consultative Committee (CCITT) employs AQB quantization as well as APC. This encoder technique is known simply as Adaptive Differential Pulse Code Modulation (ADPCM). ADPCM coding provides a toll quality encoding of a voice signal using 4 bits per sample, for a corresponding data rate of 32 kbps [24].

Following the ADPCM coder, the next waveform encoding technique illustrated in Figure 2.10 is that of Sub-band Coding. This technique uses bandpass filters to divide the speech signal up into several contiguous bands. Each band may then be sampled at a lower rate than the sampling rate required for the input signal. This division alone does not achieve a decrease in overall data rate, but if the perceptually less important signal bands are quantized with a fewer number of bits than the perceptually important bands, an overall reduction in data rate at the output of the multiplexer may be achieved. Typically the lower frequency bands, which contain pitch and formant information, are encoded with a higher number of bits per sample than the higher frequency bands. The encoders used for each band generally employ some AQB techniques [24]. The frequency bands may be of equal bandwidth or may increase in bandwidth with frequency as shown in Figure 2.12. One sub-band coding technique, which provides communications quality speech uses the frequency bands and bit assignments indicated in Table 2.2 [24]. From Table 2.2, the data rate at the output of the multiplexer may be calculated to be 15.529 kbps. A few extra bits are used for synchronization purposes, bringing the total data rate up to 16 kbps.

Another type of sub-band coder, which is occasionally used in practice, employs

Figure 2.12: Sub Band Coding Filtering Schemes

Band Number	Frequency Range	Band Sampling Frequency (Hz)	Bits Per Sample Rate (Hz)
1	178 - 356	356	4
2	296 - 593	593	4
3	533 - 1067	1067	2
4	1067 - 2133	2133	2
5	2133 - 3200	2133	2

Table 2.2: Frequency Bands and Bit Assignments For Crochiere Sub-band Coder

a special set of filters known as a Quadrature Mirror Filter (QMF) bank [25]. This filter bank segments the voice frequency band by cascading highpass and lowpass filters, each of which splits the preceding band in half and decimates the sampling rate by two. In Chapter 3, it will be shown that this type of filter bank arrangement is related to the concept of a Wavelet Transform.

Adaptation may be added to sub-band coders in an attempt to minimize the effects of quantization noise in each band. This is done by dynamically allocating a higher number of bits to the more active bands and fewer bits to the less active bands as the band activity changes over time. The total number of bits per second at the output of the multiplexer is maintained constant. A second technique occasionally used in sub-band coding takes advantage of the a characteristic of human hearing known as *noise masking* [18]. A strong formant tends to mask the noise in the vicinity of the formant frequency, provided that the noise is more than 15 dB below the level of the formant. As a result, a larger coding error may be tolerated near the formant frequencies. Therefore fewer bits are required for a band where a strong formant frequency is detected, as in the case of a voiced speech interval. These bits may be reallocated to other bands as required.

Sub-band coding is capable of providing a toll quality signal at 32 kbps and a communications quality signal at 16 kbps [24]. Although no standard for 16 kbps voice coding has yet been adopted by the CCITT, sub-band coding is one of the leading contenders. The CCITT has adopted a 64 kbps sub band coding scheme which uses QMF filtering in conjunction with ADPCM quantizers in order to transmit very high

quality audio signals [26]. These signals are typically allocated a bandwidth of 50 - 7000 Hz, making them somewhat wider than the standard voice frequency channels used in most telephony systems.

2.4.2 Source Encoders

Unlike waveform encoders, which attempt to encode speech information without any assumptions regarding how the speech was generated, source encoders rely very heavily on either a model for the speech production process or on correlation between consecutive samples of the speech signal. Source encoders, or vocoders as they are also called, are typically used in special communications applications where voice data must be transmitted using very low data rate channels. They are capable of conveying a voice signal using data rates as low as 2 kbps, but although the words in the message are clear, the voice is often synthetic sounding and speaker identification from the voice signal is impossible [25]. Typically source encoders are not as robust as waveform encoders under conditions of low SNR environments or multiple talkers, nor are they as useful as waveform encoders for the transmission of non voice signals. These limitations preclude the widespread use of vocoders for telephony applications.

The first two techniques shown in Figure 2.13, Direct Extraction LPC Coding and Analysis By Synthesis (ABS) Coding, are based on the model used in Section 2.3 for speech synthesis (Figure 2.5). As may be seen, the encoding procedure for these two techniques consists of two steps: excitation specification and filter parameter extraction. As discussed in Section 2.3, the excitation is modelled either as a periodic impulse train for voiced speech intervals, or as band limited white noise for unvoiced speech intervals. For both voiced and unvoiced speech intervals, the encoder must provide an estimate of the signal power in order to properly adjust the gain at the receiver. In the case of voiced speech segments, pitch information must also be extracted from the speech signal and sent to the receiver. A variety of pitch extraction algorithms are available [8]. The main difference between the two model based source encoder techniques is that in the case of the Direct Extraction LPC source encoder the excitation parameters are measured directly from the input signal, while in the case of the Analysis By Synthesis source encoder these parameters are optimized via

Figure 2.13: Source Encoders

the minimization of the mean squared error between the input speech segment and a synthesized version of the speech segment. ABS encoders generally apply a weighting function to the error signal so that the minimization is perceptually meaningful [25]. This time varying weighting function is calculated from the input speech spectrum in order to take advantage of the noise masking effect. The weighting function shapes the error spectrum in such a way that a larger error is allowed in regions of the spectrum surrounding the measured formant frequencies [25].

The filter coefficients for the Direct Extraction LPC Coding and ABS source encoders are determined from an open loop analysis of the input speech signal [25]. The filter is generally modelled as an all pole filter, having the transfer function

$$H(z) = \frac{1.0}{1 - \sum_{l=1}^L a_l z^{-l}}, \quad (2.13)$$

where $H(z)$ is the filter transfer function,

L is the order of the filter,

a_l for $l = 1$ to L are the filter coefficients.

This type of filter is identical to the prediction filter used in the APC waveform coding technique. As in the case of an APC encoder, the filter coefficients are obtained through the minimization of the mean squared error between the input speech signal, and an estimate of the speech signal based on previous samples of the signal, as in equation 2.9. It will be seen in Section 2.5, that these filter coefficients are also known as Linear Predictive Coding (LPC) coefficients.

Once the LPC coefficients and excitation parameters are determined, they are sent over the communications channel to the receiver where the speech signal is reconstructed from the model parameters. Typically, twelve bits is sufficient for the representation of excitation and gain information in a source encoder model [8]. A filter order of eight to twelve is generally chosen for the vocal tract model. The coefficients for this filter must be quantized prior to transmission. Unfortunately, direct quantization of the filter coefficients can lead to stability problems when the filter is duplicated at the receiver [25]. For this reason, the quantization is generally applied to alternate parameters derived from the filter coefficients. The most commonly used

parameters are the Partial Correlation (PARCOR) coefficients which are a byproduct of the Levinson Durbin algorithm for the calculation of the filter coefficients of equation 2.13 [8]. They may be calculated from the filter coefficients using the following backwards recursive relationship [8]:

$$\begin{aligned}\Gamma_i &= \alpha_i^i \\ \alpha_j^{i-1} &= \frac{\alpha_j^i + \alpha_i^i \alpha_{i-j}^i}{1 - \Gamma_i^2}\end{aligned}\tag{2.14}$$

where Γ_i are the PARCORR coefficients for $1 \leq i \leq L$, and L is the order of the filter in Equation 2.13.

The iteration is repeated for an inner loop with $1 \leq j \leq i - 1$, and an outer loop with $L \geq i \geq 1$. The α coefficients are intermediate parameters where α_j^k is the j th coefficient for the k th iteration of the outer loop. These intermediate coefficients are initialized with the values $\alpha_l^L = a_l$ for $1 \leq l \leq L$, where a_l are the filter coefficients given in Equation 2.13.

An inverse sine or hyperbolic tangent function is usually applied to the PARCOR coefficients prior to quantization [6, 25]. An average of 5 or 6 bits is used to represent each quantized PARCOR coefficient, but the lower order coefficients typically require more bits than the higher order coefficients [8]. A total of 40 to 72 bits is therefore required for the transmission of the filter coefficient information in a source encoder. When combined with the twelve bits required for the excitation information, it may be seen that approximately 52 - 84 bits is required for the transmission of the model information.

Although the transmission of the model information in a source encoder requires more bits than the transmission of a quantized signal sample, the model coefficients need not be updated on a sample by sample basis. Typically an update rate of 10 - 25 msec for model information in vocoders is used. From this, it may be seen that typical data rates for model based vocoders range from approximately 2 kbps to approximately 8.4 kbps [6, 8, 25]. An LPC based source encoder has been adopted by the United States government and the North Atlantic Treaty Organization (NATO) for the transmission of secure military communications signals [27, 28]. This system

uses 41 bits for the coding of the 10 vocal tract filter coefficients, 5 bits for the coding of the frame energy, 7 bits for the coding of pitch information, and an extra bit used for synchronization purposes. One frame of 54 bits is sent every 22.5 msec, resulting in a data rate of 2400 bps [27]. This coding scheme was also used in the original version of the United States government STU-III secure telephone system [29].

The last source encoding technique illustrated in Figure 2.13 is that of Vector Quantization [23]. Unlike Direct Extraction LPC and ABS coding, which rely heavily on a signal model, VQ relies on the correlation between adjacent speech samples. This technique is a multidimensional extension of simple PCM. In PCM, each signal sample is represented by a group of b bits. The Twos Complement representations of the 2^b possible words in the quantizer form a codebook of quantization levels used to represent the amplitude of each signal sample. In Vector Quantization, consecutive signal samples are grouped together and the 2^b possible codewords are used to represent the group of samples. Vector quantizers are characterized by the number of entries in the codebook (N_C) and the number of signal samples grouped together in the quantization (K). The data rate of a Vector Quantizer is given by

$$R = f_s \frac{\log_2 N_C}{K}, \quad (2.15)$$

where R is the encoder data rate in bits per second,

N_C is the number of entries in the codebook,

K is the number of consecutive signal samples grouped together, and

f_s is the signal sampling frequency.

As may be seen from Equation 2.15, PCM is a simple case of Vector Quantization with $K = 1$. The Vector Quantization encoding technique consists of an exhaustive search of the codebook to find the codeword best representing the vector composed of K consecutive signal samples, subject to a predefined error criterion. One commonly used error measurement is [30]

$$d(\vec{s}, \vec{c}) = \|\vec{s} - \vec{c}\|^2 = \sum_{l=0}^{K-1} [s((n+l)T_s) - c_l]^2. \quad (2.16)$$

where d is the Euclidean distance between the unquantized and quantized signal vectors.

$\vec{s}$ is the unquantized signal vector at time nT_s , which consists of the signal samples $[s(nT_s), s((n+1)T_s), \dots, s((n+k-1)T_s)]$.

$\vec{c}$ is the codeword used to represent vector $\vec{s}$. It consists of the members $[c_0, c_1, \dots, c_{k-1}]$.

K is the number of consecutive signal samples grouped together, and T_s is the sampling frequency.

The codeword which minimizes Equation 2.16 is used to represent the vector $\vec{s}$. The position of the codeword in the codebook is sent to the receiver, where an identical codebook is used to reconstruct the representation of $\vec{s}$. The vector $\vec{s}$ may represent consecutive signal samples as in Equation 2.16, or may represent a vector of LPC coefficients which have been extracted from a signal frame.

The key to the successful use of a VQ source encoder lies in the design of the codebook. It is important to remember that the quantization of a vector $\vec{s}$ in a VQ algorithm does not simply correspond to the scalar quantization of the individual members of the vector. The codebook must be designed using a large training signal representative of the type of signal to be encoded. During the training, an iterative algorithm [30] is used to find the N_C best codewords which minimize the distance function given by Equation 2.16, averaged over the length of the training sequence. The procedure consists of three steps [6].

Step 1 - Obtain an initial estimate of the codebook and determine if the average distance function is sufficiently small. If so, this codebook is sufficient. The initial estimate may be obtained using random vectors, or a simple clustering algorithm.

Step 2 - If the codebook obtained in Step 1 is not sufficient, replace each codeword with the K -dimensional centroid of all training vectors mapped into that codeword.

Step 3 - Repeat steps 1 and 2 using the new codebook until the error is minimized.

Assuming that enough training data has been used, and that there is sufficient correlation between adjacent samples of the signal, the codebook obtained from this training procedure may be used to encode signals statistically similar to the training sequence [30]. The encoding process simply involves a search of the codebook in order to find the best codeword for a given set of signal samples.

Vector quantization has been demonstrated successfully for speech signals using a data rate of 8 kbps, with $K = 8$ and $N_C = 256$ [23]. The main difficulty in extending the VQ technique to larger values of K is that the time required to perform an exhaustive search of the codebook precludes real time encoding of the speech signal. Tree and multistage search techniques can be used to speed this up somewhat, but most applications of vector quantization involve relatively low values of K .

2.4.3 Hybrid Encoders

One of the main reasons for the synthetic sound of speech which is encoded using a source encoder is the simplistic two source excitation model used to represent voiced and unvoiced speech intervals. This model makes it difficult to represent transitions between voiced and unvoiced intervals, and is somewhat unrealistic for the modelling of certain fricatives, where there are aspects of both voiced and unvoiced excitation. (The sound / $\tilde{z}$ / in the word *measure*, for example). Recently, demands for high quality encoding techniques for the transmission of voice signals over narrowband cordless and cellular telephone channels has prompted the development of a new class of speech encoder called *hybrid* encoders. A few of the more common hybrid encoding techniques are shown in Figure 2.14. Most hybrid algorithms are based on the Analysis By Synthesis (ABS) model illustrated in Figure 2.13 [25]. The LPC filter is still used to model the vocal tract, but an optimized glottal excitation signal is used instead of the simple dual mode excitation scheme applied in vocoder models. A higher order filter is also generally added to hybrid models in order to model the detailed characteristics of the glottal excitation during voiced intervals. This filter.

Figure 2.14: Hybrid Encoders

which is called the *pitch synthesis* filter, has the general form [25]

$$P(z) = \frac{1.0}{1 - \sum_{l=-1}^1 g_l z^{-(\Delta+l)}} \quad (2.17)$$

where $P(z)$ is the filter transfer function,

g_l for $l = -1, 0$ and 1 are the filter coefficients and,

Δ is a multiple of the estimated pitch period.

The Δ term and the filter coefficients g_l are calculated as part of the error minimization procedure [25]. Since the estimated pitch period is usually about 20 to 150 sampling periods, the order of the pitch synthesis filter is much higher than that of the vocal tract filter. The pitch synthesis filter however only requires a few non-zero coefficients, so the transmission of these filter coefficients does not add a great deal of overhead to the bit allocation scheme for the encoder. Some hybrid encoder schemes use only a single non-zero coefficient for the pitch synthesis filter [31]. The Δ term must also be quantized and sent to the receiver, but this may also be done using a minimal number of bits.

The optimization of the excitation in hybrid encoders does require that significantly more bits be allocated towards the transmission of excitation information than in the case of source encoders. This implies that the data rate for hybrid encoders is typically higher than that of source encoders. Typically, these algorithms provide high communications quality speech with data rates between 4.8 - 13 kbps [25].

The first hybrid technique shown in Figure 2.14, that of Multi-Pulse Linear Predictive Coding (MP-LPC), uses a series of pulses as the glottal excitation for both voiced and unvoiced speech intervals. The pulses are generated at an average rate of 0.8 per msec, but the exact amplitude and position of each pulse is determined through the minimization of the weighted mean squared error between the original and synthesized speech segments [25]. The pulse locations and amplitudes are obtained sequentially for each encoded speech segment. Once the parameters for the first pulse have been determined, a new error function is computed by subtracting out the contribution of the first pulse. The location of the next pulse is then determined through the minimization of the new error function. The process is continued

until the error is reduced to an acceptable value, or the number of pulses reaches a maximum value which is defined by the number of encoding bits allocated for the source model [32]. A variant of the Multi-Pulse technique, which is known as Regular Pulse Excitation (RPE) makes the assumption that the excitation pulses are evenly spaced. The optimization of the excitation therefore only involves the calculation of the pulse amplitudes. RPE with a data rate of 13 kbps has been adopted for use in Groupe Special Mobile (GSM) cellular telephone systems [25].

Code Excited Linear Prediction (CELP) encoding uses a principle similar to that of MP-LPC coding, except that the excitation is modelled as a Gaussian process [25]. This type of encoding technique is also called Stochastic Excitation coding [32]. Various types of Gaussian excitation functions are stored in a codebook using Vector Quantization (VQ) techniques. Optimization of the excitation sequence consists of an exhaustive search of the excitation codebook in order to find the sequence which minimizes the error between the input speech segment and the synthesized speech segment. Codebooks of between 256 and 1024 entries are most common in CELP systems. The details of the glottal excitation in CELP coders may be through the use of a pitch synthesis filter, as in the case of the MP-LPC coder[29], or may be specified through the use of a second codebook as shown in Figure 2.14. This codebook uses a set of excitation sequences which are derived from the pitch characteristics of the speaker [25, 29]. The pitch characterization sequence is added to the Gaussian excitation sequence in order to create a combined excitation sequence for the vocal tract filter [33]. CELP encoders generally segment the input speech into frames of duration 20 - 30 msec, and perform the vocal tract characterization for each frame. Each frame is also divided into two to four subframes, where the excitation codebook entry, excitation gain, and pitch characterization is updated for every subframe.

CELP coding has been selected by the United States government as a standard for voice encoding at 4800 bps, and is used on the newer generation STU-III secure telephone systems [29, 34]. The adopted standard uses a tenth order filter for modelling the vocal tract. These ten filter coefficients are updated every 30 msec and sent to the receiver using 34 bits. Each 30 msec frame is also divided into 4 subframes. For each subframe, the pitch synthesis information is sent to the receiver using an

average of 11 bits per subframe (44 bits per frame). A codebook of 1024 entries is used for the excitation, requiring that ten bits be used to specify the excitation for each subframe. An additional 5 bits is used to specify the excitation gain for each subframe. The specification of the excitation therefore requires a total of 60 bits per frame. For each frame, an additional bit is used for synchronization purposes, 4 bits are used for forward error correction, and 1 bit is allocated to provide for future expansion of the standard. A total of 144 bits is therefore sent every 30 msec resulting in a data rate of 4.8 kbps [34].

One of the main problems with high quality CELP coding is that the codebook search for the Gaussian excitation sequence is computationally very intensive for larger codebooks [29]. Vector Sum Linear Prediction (VSELP) coding attempts to reduce this problem by using two smaller Gaussian excitation codebooks instead of one large codebook. As in the case of CELP coding, a pitch synthesis codebook is also used for the detailed characterization of the glottal excitation. This pitch synthesis sequence is added to the sum of the two Gaussian sequences in order to provide the vocal tract excitation. VSELP coding at 8.0 kbps has been adopted for use in the North American IS-54 digital cellular telephone system. In this system, the input speech signal is divided into 20 msec frames, which are further subdivided into 5 msec intervals. For each frame the vocal tract model is specified using a tenth order filter, the coefficients of which are quantized using a total of 38 bits per frame. The energy per frame is also measured and sent to the receiver using 5 bits. The three excitation codebooks used each contain 128 entries. The codebooks entries which minimize the mean squared error between the synthesized signal and the input signal are updated for every subframe. Therefore a total of 84 bits ($3 \times 7 \times 4$) per frame is required in order to send the excitation information to the receiver. The gain parameters for all three excitation sequences are vector quantized and sent to the receiver using 8 bits per subframe, for a total of 32 bits per frame. An extra bit per frame has also been allocated to the standard for future expansion purposes. A total of 160 bits is therefore sent every 20 msec for a data rate of 8.0 kbps [35].

2.5 Speech Audition

The audition of speech involves the conversion of sound waves into electrical impulses, which are sent to the brain. This conversion takes place within the ear which is detailed in Figure 2.15. The ear may be divided into three main sections: the outer ear, the middle ear, and the inner ear. The outer ear consists of the externally visible portion or *pinna*, and the *meatus* which is the auditory canal leading to the eardrum. The function of the outer ear components is to direct pressure variations in the air towards the eardrum, which forms the divider between the outer ear and the middle ear. The auditory canal also forms a resonant cavity serving to amplify sounds in the frequency range 3 - 5 kHz [6].

As sound waves impinge upon the eardrum, they cause vibrations which are transferred through the inner ear via the *ossicular* bones. These three tiny bones, known as the *malleus*, the *incus*, and the *stapes* compose the middle ear and transfer mechanical vibrations from the air filled outer ear to the fluid filled inner ear.

The inner ear is composed of the *cochlea*, the *vestibula*, and the *auditory nerve*. There is also a series of semicircular, fluid filled canals which are not related to hearing but serve as organs of balance [36]. The cochlea and the vestibula are chambers filled with fluids and membranes which vibrate in response to the mechanical excitation of the ossicular bones. The most important of these membranes is the *basilar membrane*, which is located in the cochlea. This membrane is approximately 32 - 35 mm in length and varies from a width of 0.04 mm at one end to roughly 0.5 mm at the other end [6]. The variation in shape, as well as tautness, along its length causes a spatial variation in the frequency response of the membrane to the excitation of the ossicular bones. Mechanical disturbances at the eardrum are therefore transformed into spatio-temporal disturbances on the basilar membrane, with each section of the basilar membrane vibrating in response to a certain characteristic excitation frequency [37]. Functionally, the basilar membrane acts as a bank of bandpass filters applied to the eardrum excitation, with each filter located at a physically different area on the membrane. It has also been shown that the relationship between the spatiotemporal disturbance and the frequency of the eardrum excitation is such that

Figure 2.15: The Human Ear

for frequencies below 500 - 800 Hz, the displacement of the vibration along the length of the membrane is linearly proportional to the frequency of the excitation. Above 800 Hz, the displacement of the membrane vibration is logarithmically proportional to the frequency of excitation [6, 37]. Furthermore, it has also been shown that the membrane disturbances above 800 Hz are such that the displacement of the vibration along the length of the membrane is proportional to the size of the disturbance area. As a result, the basilar membrane response may be modelled as a bank of filters which have constant bandwidths and are linearly spaced below 500 - 800 Hz, but are logarithmically spaced above 800 Hz. The filters above 800 Hz are generally modelled such that the ratio of center frequency to bandwidth is a constant. This is known as a constant Q filter bank, and will be discussed in Chapter 3.

The mechanical vibrations along the basilar membrane are transduced into electrical impulses through a network of tiny filaments called *cilia*. These filaments are attached to the *Corti* organ which lies on top of the basilar membrane. Each filament is attached to a transduction cell within the Corti known as an *inner hair cell*. As the basilar membrane vibrates, it displaces fluid which bends the cilia on the Corti. This in turn causes the flow of ionic currents through the inner hair cells which generate impulses within the auditory nerve fibers connected to the hair cells. These impulses are conveyed by the auditory nerve to the brain. Each auditory nerve fibre innervates only one inner hair cell, however as many as ten fibres may converge on a hair cell. In the human auditory system, there are approximately 30,000 auditory nerve fibers innervating approximately 3000 inner hair cells along the length of the basilar membrane [37].

As in the case of artificial speech transmission, artificial audition of speech is not new. It has been practiced for years in the design of microphone equipment, where an electrical signal is created from the vibration of thin metal film in response to the air pressure created by sound waves. In the case of dynamic microphones, this vibration is used to move a coil through a magnetic field, thereby creating a changing current proportional to the input speech. In the case of piezo-electric microphones, the vibrating membrane is used to stress a small piece of piezo-electric material which creates a voltage proportional to the input speech signal [7]. Most current research

in the area of speech audition focuses on the extraction of digital parameters from an analog audio signal which lend themselves to computer analysis in order to characterize the speech. These parameters must be carefully chosen in order to average out the effects of noise, but not the time varying nature of the speech signal.

As discussed in Section 2.3, the vocal tract forms a time varying filter to which either a periodic or a noise like excitation is applied. The time varying nature of this filter implies that speech is a statistically non-stationary signal. Its variance, bandwidth, and power spectrum are all functions of time. The power spectra of two separate phonemes are shown in Figure 2.16. As can be seen, the spectrum of the voiced phoneme (/ɔ/) reveals both the pitch and formant structure of the waveform. The voiced phoneme has a pitch of approximately 120 Hz, an F_1 frequency of approximately 700 Hz, an F_2 frequency of approximately 1000 Hz, and an F_3 frequency of approximately 2500 Hz. The unvoiced phoneme has a spectrum which is more noise like, and does not reveal pitch or formant information. It also has a slightly higher bandwidth than the voiced phoneme. In practice, voiced speech intervals occur approximately four times as much as unvoiced speech intervals [25] and are typically considered more important for the analysis of speech in speech and speaker recognition applications.

It is the non-stationarity of speech which makes it a very difficult signal to process. Generally, a quasi-stationary model is assumed and the signal is processed as a stationary signal over a finite time segment ranging from 10 to 20 msec. This finite time interval is known as a *frame*. The choice of frame length often leads to a trade-off between the requirements of measuring different characteristics of speech. A short time segment is more appropriate for the measurement of pitch frequency and the exact location of stop phoneme transients for example. A long time segment is more appropriate for the measurement of formant frequencies [6].

The difficulty of processing the non-stationary speech signal is compounded by the fact that there is a high percentage of inactive intervals during most speech signals. A typical speaker talks approximately 40 percent of the time [25]. In addition to the natural breaks in a conversation, there are also small intervals of silence between words in a sentence and in some cases between phonemes in words. Although the

Figure 2.16: Spectra Of Phonemes /ɔ/ and /s/

duration of the intra-phoneme silence intervals may be perceptually important, and is generally included in the processing of a speech signal, the processing of a speech signal during intra-word silence intervals usually results in an excessive amount of information which is not directly related to the speech signal. Generally, some form of silence-speech detection is required so that only active speech intervals are processed. The separation of speech and silence intervals is made difficult by the fact that the distribution of speech signal amplitudes is such that most speech is distributed at the smaller amplitudes. This is especially problematic in situations of high background noise. As may be seen from the histogram shown in Figure 2.17, the probability density of speech signal amplitudes is narrowly distributed around zero. The histogram illustrated in Figure 2.17 was calculated using over four and a half hours of conversational speech contained in the SPIDRE speech corpus. Silence intervals were removed from the histogram using an algorithm which will be discussed in Chapter 5, and the variance of the remaining samples was measured. As may be seen, the histogram is very similar to a Laplacian distribution of identical variance [8].

During active speech intervals, many different types of parameters are used to characterize the speech signal. Some parameters are based in the time domain, while other types are based on a transformation of the time domain signal into a set of spectral parameters. Common parameters are described below. Prior to the parameter extraction, most speech processing systems perform some form of pre-emphasis on the received speech. This compensates for the 6 dB/octave roll off of the glottal excitation spectrum. Generally, a simple filter of the form

$$H(z) = 1.0 - \alpha z^{-1} \quad (2.18)$$

is applied. α is generally chosen to have a value in the range 0.9 to 0.95.

2.5.1 Time Domain Parameters

Time domain parameters have the advantage that they are easy to calculate. The most frequently used time domain parameter is that of frame energy, which is given

Figure 2.17: Histogram Of Speech Compared To Laplacian PDF

by

$$E(nT_s) = \sum_{j=-\infty}^{\infty} |s(jT_s)w((n-j)T_s)|^2, \quad (2.19)$$

where $E(n)$ is frame energy at time nT_s ,

T_s is the sampling frequency,

$w(i)$ for $i = 0$ to $N-1$ is a sliding window which defines

the length of the speech frame. NT_s is the length of the speech frame.

It is important to note that the window in Equation 2.19 is non zero only over a finite time interval. The summation is therefore performed only over a small window at time nT_s . The resulting frame energy is a time varying function. The window in Equation 2.19 also usually incorporates some data tapering feature, allowing for the smoothing of discontinuities in the processed speech frames as the window is stepped in time. Frame energy is often used in speech processing systems as an indication of speech activity in the separation of speech and silence intervals. It is also used as a parameter in the preliminary segmentation of voiced and unvoiced speech intervals and in the preliminary segmentation of phonemes [6].

Other common time domain parameters include the average zero crossing rate, which is used in the calculation of pitch (F_0) frequency, and the Short Time Autocorrelation function which is given by

$$r_n(k) = \sum_{j=-\infty}^{\infty} s(jT_s)w((n-j)T_s)s^*((j-k)T_s)w^*((j-k)T_s), \quad (2.20)$$

where $r_n(k)$ is the autocorrelation value at time nT_s and a relative delay of kT_s ,

T_s is the sampling frequency.

$w(i)$ for $i = 0$ to $N-1$ is a sliding window as in Equation 2.19.

As in the case of frame energy, the Short Time Autocorrelation function is time dependent. For a relative delay of $k = 0$, the function reduces to that of Equation 2.19. Simple analysis of the time period between the peaks of the Short Time Autocorrelation function can lead to an accurate estimate of the pitch frequency [8]. Equation 2.20 is also required in the calculation of the Linear Predictive Coefficients, which are used in the measurement of formant information [6].

2.5.2 Frequency Domain Parameters

Spectral parameters are the most commonly used parameters in the processing of speech signals. In general, spectral parameters involve the description of a time domain signal in terms of a set of coefficients derived from a series of scalar product operations between the original signal and a set of orthonormal basis functions [38]. The most common set of basis functions used are the complex exponentials, which result in the common Fourier Transform coefficients. These coefficients are commonly extracted from a frame of speech through the application of the Fourier Transform operation on the windowed time signal as follows:

$$S_n(k) = \sum_{m=-\infty}^{\infty} s(mT_s)w((n-m)T_s)e^{-j\frac{2\pi mk}{T_s}}. \quad (2.21)$$

where $S_n(k)$ is the spectral coefficient for frequency $\frac{k}{T_s}$ at time nT_s ,

T_s is the sampling frequency, and

$w(i)$ for $i = 0$ to $N-1$ is a sliding window as in Equation 2.19.

Equation 2.21 is commonly referred to as a Short Time Fourier Transform, and indicates the distribution of signal energy with frequency for the speech frame contained within the window function. Although Equation 2.21 contains information about both the magnitude and phase of the spectral components contained within the speech signal, it appears that the human hearing mechanism ignores phase information [6]. For this reason most artificial speech processing systems use only the magnitude information contained in Equation 2.21, in the form of either $|S_n(k)|$, $\log|S_n(k)|$, $|S_n(k)|^2$, or $\log|S_n(k)|^2$. A common way of displaying the Short Time Fourier Transform over time is through the use of a *spectrogram* plot, in which the magnitude information is indicated using intensity on a two dimensional display of frequency vs time [6].

The multiplication of the speech frame with the window function in Equation 2.21 corresponds to the convolution of the frame spectrum with a lowpass filtering function defined by the Fourier Transform of the window function. Since the bandwidth of the lowpass function is inversely proportional to the time duration of the window function (NT_s), the frequency resolution of the spectrogram may be varied in order to bring

Figure 2.18: Wideband and Narrowband Spectrograms Of The Word *Carleton*

out certain features of the speech waveform. Typically two types of spectrograms are used in speech analysis: wideband spectrograms and narrowband spectrograms.

A wideband spectrogram uses a time window which is typically 3 to 8 msec in duration [6, 8, 39]. The short duration time window results in a spectrogram which is more appropriate for the analysis of formant information since the frequency resolution is rather coarse. In the wideband spectrogram shown in Figure 2.18, the formants appear clearly as broad dark bands during the voiced speech intervals corresponding to the phonemes /ɔ/, /r/, and /l/ in the word *Carleton*. An 8 msec Hanning window was used in the calculation of this wideband spectrogram.

Narrowband spectrograms employ a longer time window, which results in more resolution in the frequency domain. This type of spectrogram is more appropriate for the extraction of pitch information. In the narrowband spectrogram shown in Figure 2.18, the pitch is evident from the narrow dark lines which occur during the voiced speech intervals. These lines are spaced approximately 150 Hz apart, indicating a pitch frequency of approximately 150 Hz. A Hanning window 51 msec in duration was used for the calculation of the narrowband spectrogram in Figure 2.18. Window durations of 20 to 50 msec are common in the calculation of narrowband spectrograms [6, 8, 39].

One of the limitations of the Short Time Fourier Transform is a trade-off between temporal (time) resolution and frequency resolution which results with the choice of window width in Equation 2.21. A long duration window, while allowing for high resolution in the frequency domain, does not allow the accurate isolation of transients in the time domain. The accurate isolation of these transients is often important for the accurate segmentation of words and phonemes. Similarly a short duration time window, while allowing the accurate isolation of transients, does not provide the frequency domain resolution necessary for the extraction of accurate pitch information. A recent signal processing technique which has been developed in an attempt to simultaneously provide high temporal and frequency resolution is that of the Wavelet Transform, which will be discussed in Chapter 3.

In many speech processing systems, the extraction of signal energy contained within various frequency bands is obtained through the direct filtering approach, as

opposed to the Short Time Fourier Transform operation in Equation 2.21. The signal is applied to a bank of bandpass filters, each covering a different frequency range. The output of each bandpass filter is given by

$$s_p(n) = \sum_{l=0}^L h_p(l) s((n-l)T_s), \quad (2.22)$$

where $s_p(n)$ is output of the bandpass filter covering the p th band,

T_s is the sampling frequency,

$h_p(l)$ for $l = 0$ to L are a set of digital bandpass filter coefficients for the p th band, and

L is the order of the digital filter.

The energy in each band is then obtained using Equation 2.19 applied to the output of each bandpass filter. Direct filtering has the advantage that the energy in the various frequency bands may be extracted simultaneously, allowing for an increase in throughput over systems employing the Short Time Fourier Transform. Direct filtering also offers the flexibility that the frequency bands may be varied in order to more accurately model the constant Q response of the ear. Generally speech processing systems use overlapping filters, the center frequencies of which are linearly spaced below a frequency of approximately 1.0 kHz, and nonlinearly spaced above 1.0 kHz.

Two popular frequency scales are employed for the spacing of filters. Both use a re-mapping of the linear frequency. The *Bark* scale, which was developed from a consideration of physical measurements performed on the basilar membrane, re-maps frequencies using the following relationship [6]:

$$f_b = 1300 \arctan \left(\frac{0.76 f_l}{1000} \right) + 3.5 \arctan \left(\frac{f_l}{7500} \right) \quad (2.23)$$

where f_b is Bark Scale frequency in Hz, and

f_l is the linear frequency in Hz.

The MEL scale re-maps linear frequency spacings using the relationship [6]

$$f_m = 2595 \log_{10} \left(1 + \frac{f_l}{700} \right). \quad (2.24)$$

where f_b is MEL Scale frequency in Hz and,

f_l is the linear frequency in Hz.

As may be seen from Figure 2.19, both Bark and MEL mappings are approximately a 1:1 mapping for frequencies below 1.0 kHz, but they compress the linear frequency scale above 1.0 kHz. Normally center frequencies of the filter banks used in speech processing systems are chosen so that they are evenly spaced on either the Bark or MEL scales [6]. This results in a set of filters which are approximately evenly spaced below 1.0 kHz but are wider and further apart, as shown in the second filter bank of Figure 2.12, for frequencies above 1.0 kHz.

2.5.3 LPC Coefficients

Another common parameter set used to characterize a frame of speech is the set of filter coefficients used in the model shown in Figure 2.5. Along with an indication of voiced or unvoiced activity, and some pitch information in the case of voiced speech intervals, these filter coefficients give a complete characterization of the speech frame.

The extraction of these filter coefficients is based on the premise that the correlation between adjacent samples of a speech signal allows a reasonable estimate of a speech sample to be made from a linear combination of previous speech samples. It is also assumed that the speech signal is stationary within the analysis frame. As shown in Figure 2.20, an estimate of the speech signal at time nT_s is made from a combination of the previous L samples. The expression for the linear estimate of the speech signal at time nT_s is given by Equation 2.9, with $\hat{s}(n-l)$ replaced by the actual signal samples $s(n-l)$. The error between the actual signal sample $s(n)$ and the estimated signal sample $\hat{s}(n)$ is given by Equation 2.10. The prediction filter coefficients are determined through the minimization of the mean of the square of the error signal $e(n)$.

Once the minimization coefficients are determined, the filter response forms a model for the vocal tract filter function in a source encoder. As discussed in Section 2.4 the filter is generally an all pole type as shown in Figure 2.20. One of the limitations of LPC coefficients for the characterization of a frame of speech is that the all pole

Figure 2.19: Bark and MEL Scale Frequency Re-mappings

Figure 2.20: Linear Prediction Model

model does not accurately represent the coupling of the vocal and nasal tracts for the generation of nasal sounds. The modification of the filter to a more complicated pole-zero model can correct this, but it is rarely done in speech processing applications.

Equations 2.9 and 2.10 may be re-written in vector notation and combined as follows [40]:

$$e(n) = s(n) - \tilde{\mathbf{a}}^H \tilde{\mathbf{s}}(n-1) \quad (2.25)$$

where $e(n)$ is difference (error) signal at time nT_s ,

$s(n)$ is the value of the speech signal at time nT_s ,

$\tilde{\mathbf{a}}$ is a column vector composed of the coefficients of the prediction filter, and

$\tilde{\mathbf{s}}(n-1)$ is a row vector composed of the previous L samples of the speech signal.

ie. $[s(n-1) \ s(n-2) \ \dots s(n-L)]$.

The H in Equation 2.25 denotes a Hermitian Transpose operator on the column vector $\tilde{\mathbf{a}}$. Multiplying both sides of Equation 2.25 by its complex conjugate, and taking the mathematical expectation of both sides yields:

$$E|e(n)|^2 = \sigma_s^2 - \tilde{\mathbf{r}}^H \tilde{\mathbf{a}} - \tilde{\mathbf{a}}^H \tilde{\mathbf{r}} + \tilde{\mathbf{a}}^H \mathbf{R} \tilde{\mathbf{a}}, \quad (2.26)$$

where $\sigma_s^2 = E|s(n)|^2$ which is the variance of the input signal s .

$\tilde{\mathbf{r}} = E[\tilde{\mathbf{s}}(n-1)s^*(n)]$ is a row vector which may be written as:

$[E(s(n-1)s^*(n)) \ E(s(n-2)s^*(n)) \ \dots E(s(n-L)s^*(n))]$, or
 $[r(0) \ r(-1) \ \dots r(1-L)]$.

$\mathbf{R}$ is the correlation matrix for the input signal $s(n)$. It is defined by

$$\mathbf{R} = E \begin{bmatrix} s(n-1)s^*(n-1) & s(n-1)s^*(n-2) & \dots & s(n-1)s^*(n-L) \\ s(n-2)s^*(n-1) & s(n-2)s^*(n-2) & \dots & s(n-2)s^*(n-L) \\ \cdot & \cdot & \dots & \cdot \\ \cdot & \cdot & \dots & \cdot \\ \cdot & \cdot & \dots & \cdot \\ s(n-L)s^*(n-1) & s(n-L)s^*(n-2) & \dots & s(n-L)s^*(n-L) \end{bmatrix} \quad (2.27)$$

$$= \begin{bmatrix} r(0) & r(1) & r(2) & \dots & r(M-1) \\ r(-1) & r(0) & r(1) & \dots & r(M-2) \\ \cdot & \cdot & \cdot & \dots & \cdot \\ \cdot & \cdot & \cdot & \dots & \cdot \\ \cdot & \cdot & \cdot & \dots & \cdot \\ r(1-L) & r(2-L) & r(3-L) & \dots & r(0) \end{bmatrix}$$

The vector $\vec{a}$ which minimizes the mean of the square of the error signal in Equation 2.26 may be obtained by taking the gradient of the right hand side of Equation 2.26 and setting the result equal to zero. Following this procedure, it may be seen that the optimum vector $\vec{a}$ must satisfy the equation:

$$\mathbf{R}\vec{a} = \vec{r}. \quad (2.28)$$

In practice, estimates of the individual elements of the matrix $\mathbf{R}$ and the vector $\vec{r}$ are obtained for the speech frame using the windowing approach of Equation 2.20, where the estimate of $r(i)$ for the speech frame at time nT_s is given by $r_n(k)$.

Equation 2.28 is solved for $\vec{a}$ through the application of the Levinson Durbin algorithm. This iterative algorithm takes advantage of the Toeplitz structure of $\mathbf{R}$ in order to calculate the solution of Equation 2.28 without a matrix inversion operation. The algorithm is initialized by setting $P_0 = r(0)$ and $a_p^0 = 1$ for $1 \leq p \leq L$. The coefficients are calculated from the following recursion:

For $1 \leq l \leq L$

$$\Gamma_l = -\frac{1.0}{P_{l-1}} \left[r(-l) - \sum_{k=1}^{l-1} a_k^{l-1} r(k-l) \right]$$

For $1 \leq j \leq l-1$ (2.29)

$$a_l^j = a_{l-1}^j + \Gamma_l a_{l-j}^{l-1}$$

$$a_l^l = \Gamma_l$$

$$P_l^j = [1 - |\Gamma_l|^2] P_{l-1}$$

a_l^j represents the l th coefficient calculated during the j th iteration of the inner loop. The coefficients calculated at the end of the outer loop are the set of coefficients $\{a_l\}_{l=1}^L$ which minimize the left hand side of Equation 2.26, and can be used to characterize the speech frame. The set of coefficients $\{\gamma_m\}_{m=1}^M$ are known as the reflection coefficients. These may be used in place of the filter coefficients to characterize the speech frame. These coefficients may also be used to generate the PARCOR coefficients which are an equivalent set of coefficients for describing the frame. The conversion between the reflection coefficients and the PARCOR coefficients is given in Equation 2.15.

2.5.4 Cepstral Coefficients

Cepstral coefficients are derived from the theory of homomorphic system analysis. This analysis is often used in acoustic and image processing to analyze and separate signals which have been combined nonlinearly [41]. Homomorphic systems obey a generalized principle of superposition where

$$H[x_1(n) \square x_2(n)] = H[x_1(n)] \circ H[x_2(n)] \quad (2.30)$$

and

$$H[c \wr x_1(n)] = c \diamond H[x_1(n)]. \quad (2.31)$$

$\square$ is a general operation between two input sequences,

$\circ$ is a general operation between two output sequences,

$\wr$ is a general operation between the scalar c and an input sequence,

$\diamond$ is a general operation between the scalar c and an output sequence, and

H is a system transformation.

In standard linear systems, the $\square$ and $\circ$ operations are replaced by addition. The $\wr$ and $\diamond$ operations are replaced by multiplication. All homomorphic systems are

Figure 2.21: Canonic Representation For Homomorphic System

Figure 2.22: Characteristic and Inverse Characteristic Systems For Speech

capable of being represented in a canonic form as illustrated in Figure 2.21. In this form, the transformation H is replaced by three separate transformations. The first block $D_{\square}$, which is called the *characteristic system*, maps the input operation ($\square$) into a simple addition of the two transformed components. The characteristic system must obey the following properties:

$$D_{\square} [x_1(n) \square x_2(n)] = D_{\square} [x_1(n)] + D_{\square} [x_2(n)] \quad (2.32)$$

and

$$D_{\square} [c \wr x_1(n)] = c \diamond D_{\square} [x_1(n)]. \quad (2.33)$$

This allows the signal to be processed using a standard linear system L . The final block is simply the inverse of the characteristic system.

In speech processing applications, it is common to model the input and output operations of Figure 2.21 as convolutions for both the $\square$ and $\circ$ operations. The characteristic system used for speech, which transforms the convolution operation into a simple addition is shown in Figure 2.22. The logarithm operation employed in Figure 2.22 is the complex logarithm operation where

$$\log_{10}(z) = \log_{10}|z| + j \arg(z). \quad (2.34)$$

Some care must be taken in the definition of the imaginary portion of the logarithm operation. In general, if the principle value of an argument is used, such that $-\pi < \text{ARG}(z) \leq \pi$, superposition does not hold since $\text{ARG}(z_1 + z_2) \neq \text{ARG}(z_1) + \text{ARG}(z_2)$. To avoid this problem, the phase is generally unwrapped and allowed to vary over the range $-\infty < \arg(z) < \infty$ [41].

The output of the characteristic system for speech is known as the complex cepstrum of the speech signal. This can be a bit of a misnomer, since for real input signals, the "complex" cepstrum is always real. The complex cepstrum is commonly used to characterize speech signal frames since it has the property that for voiced intervals it allows the simple separation of the periodic excitation and vocal tract filter components. Low time components of the complex cepstrum correspond to the vocal tract transfer function. Filtering the complex cepstrum such that only low

time components are applied to the inverse characteristic transfer function allows the deconvolution of the periodic excitation and vocal tract filter functions, with the subsequent extraction of the vocal tract parameters. Using a filter which allows the high time components to pass through allows the recovery of the excitation component. Cepstral analysis is also used for the separation of voiced and unvoiced speech intervals since for voiced speech intervals, the complex cepstrum contains a peak at the time corresponding to the period of the excitation. Lack of this peak, contained in the higher time portion of the cepstrum, indicates an unvoiced speech interval. The cepstrum for the phoneme /ɔ/ is illustrated in Figure 2.23, along with its time domain representation and its spectrum. The small cepstral peak at $t = 8.8$ msec is an indication of voiced excitation with a pitch frequency of approximately 113 Hz. This pitch frequency is also evident in the power spectral density plot.

The complex cepstrum also has the property that for minimum phase signals, it may be represented by the Inverse Fourier Transform of the real component of Equation 2.34. The Inverse Fourier Transform of $\log_{10} |z|$ is commonly called the **real cepstrum** or simply the **cepstrum** of the speech signal. It has almost all of the properties of the complex cepstrum, but is somewhat easier to calculate. Since speech is often modelled as a minimum phase signal, the cepstrum is often used instead of the complex cepstrum for the characterization of speech signals [41]. For minimum phase signals, the $\log_{10} |z|$ operation may also be replaced by a $\log_{10} |z|^2$ operation. The properties of the characteristic system are still preserved but it allows signal energy to be used in the calculation of the cepstrum.

It is important to realize that the resolution of the Fourier Transform and Inverse Fourier Transform operations in Figure 2.22 need not be the same. Fourier transform coefficients may be added prior to the logarithm operation in order to combine the energy from different frequency bands. The bands may be evenly spaced or unevenly spaced as in Figure 2.12. It is very common for the bands to be spaced according to a MEL frequency scale. In this case, the Inverse Fourier Transform operation yields a set of coefficients commonly referred to as **MEL-Spaced Cepstral Coefficients**.

Figure 2.23: Signal, Complex Cepstrum and Spectrum For Phoneme /ɔ/

2.6 Information Perception

The final link in the speech chain shown in Figure 2.1, is the information perception stage. This stage involves the recognition of patterns created by the parameter sets extracted from active speech frames. As with the message conception link of the speech chain, the processing involved in the perception stage is largely unconscious. A great deal of work has gone on in the past however in order to simulate selected portions of the perception stage. Word recognition, language recognition, phoneme recognition, vowel recognition, and speaker recognition are all topics of considerable interest in speech processing [42, 43]. Generally, human listeners do not make a conscious distinction between speaker recognition and speech (word, phoneme, voice) recognition. The same parameters appear to be extracted by the ear in each case [37] and the brain simultaneously processes both speaker and speech related information.

In the case of artificial information perception applications, the parameters extracted from the speech signal are generally optimized for the particular application in order to save memory and processing requirements. For example, phonemes may generally be grouped into manner of articulation classes using simple time domain parameters such as the zero crossing rate, but discriminating among vowels or the place of articulation among consonants requires spectral processing. As may be seen from the wideband spectrogram in Figure 2.18, the transition between the voiced phonemes /ɔ/, /r/, and /l/ in the word "Carleton" is determined largely by the movement of the third formant (F3) frequency between the values of 1.5 to 2.5 kHz. The first three formant frequencies are generally required for the distinction of voiced phonemes in speech processing applications [17]. Other speech and speaker recognition applications require the extraction of more complex parameters.

It is important to realize that parameters suitable for the creation of templates in one speech recognition application may not be suitable for other applications. For example, parameters which provide good performance in the recognition of words and phonemes are usually not suitable for speaker recognition applications. The goal of properly selected speech processing parameters is to maximize the separation between the words or phonemes spoken by a wide range of speakers. They are chosen

so that the parameter variance due to the effects of different speakers, the inter-speaker variance, is minimized. Speaker recognition algorithms, on the other hand, strive for the maximization of inter-speaker parameter variation, while minimizing the intra-speaker variation caused by different phonemes being uttered by the same speaker. One often used measurement for the effectiveness of parameters for speaker recognition is that of the *F-ratio* [6], which compares inter- and intra-speaker variances as follows:

$$F = \frac{\sigma_{\mu}^2}{E[\sigma_p^2]} \quad (2.35)$$

where σ_{μ}^2 is the variance of the parameter mean, measured over several different utterances from the same speaker.

μ is the mean of the extracted parameter, measured over a single sample of speech from the same speaker.

σ_p^2 is the variance of the extracted parameter, measured over a single sample of speech from a single speaker.

$E[\sigma_p^2]$ is the mean of the parameter variance measured over many samples of speech from different speakers. This is known as the mean intra-speaker variance.

A similar criterion could be used for the evaluation of parameters for speech recognition. In this case, the numerator would indicate the variance of the parameter mean measured over different utterances of the word or phoneme of interest by several speakers, and the denominator would indicate the mean of the parameter variances measured over utterance of other words or phonemes by several speakers.

Once parameters appropriate for the speech or speaker recognition application have been extracted from a speech segment, the artificial simulation of the information perception stage of the speech chain becomes a problem of statistical pattern recognition. As shown in Figure 2.24, pattern recognition involves the analysis of feature vectors in a K -dimensional space, where K is the dimension of the feature vectors extracted from active speech frames. Each vector $\vec{x}_i$ is a specific occurrence of a K -dimensional random vector $\vec{X}$. These random vectors may be grouped into a

Figure 2.24: Conceptual Pattern Recognition Model

finite number of classes which correspond to a given word or speaker. The pattern recognition problem is to find the most likely class corresponding to a given observation of the random vector. This is formally stated as a Maximum A Posteriori (MAP) detection problem where class C_j is chosen if [44]

$$Pr [C_j|\vec{X}] > Pr [C_i|\vec{X}] \text{ for all } i \neq j, \quad (2.36)$$

where $Pr [C_j|\vec{X}]$ is a posteriori probability of class C_j given an observation of the random vector $\vec{X}$.

The problem can be restated in terms of the apriori probabilities if both sides of Equation 2.36 are multiplied by the unconditional probability density of $\vec{X}$. Class C_j is then chosen as the most likely class if

$$Pr [C_j] p_{\vec{X}} [\vec{x}|C_j] > Pr [C_i] p_{\vec{X}} [\vec{x}|C_i] \text{ for all } i \neq j, \quad (2.37)$$

where $Pr [C_j]$ is the probability of occurrence of class C_j , and
 $p_{\vec{X}} [\vec{x}|C_i]$ is the probability density of the random vector $\vec{X}$
 given the occurrence of class C_i .

If the apriori probabilities for the occurrence of each class are assumed to be equal, the problem becomes a Maximum Likelihood detection problem, where class C_j is chosen as the most likely class if

$$p_{\vec{X}} [\vec{x}|C_j] > p_{\vec{X}} [\vec{x}|C_i] \text{ for all } i \neq j. \quad (2.38)$$

In most cases, the K -fold conditional probability density functions for the random vector $\vec{X}$ are not known. Pattern classification algorithms which make an assumption regarding the statistics or probability density functions in Equations 2.37 or 2.38 are known as *parametric* pattern classification algorithms. Algorithms which make no assumption regarding the underlying statistics are known as *non-parametric* algorithms. In general, parametric models are simpler than non-parametric models since the classification problem is specified in terms of a few statistical parameters such as mean vectors or covariance matrices which may be estimated from the feature vectors. Non-parametric models are typically more complex and either involve the estimation

of the underlying probability density functions from the data or are based on supervised clustering algorithms [45]. Under certain conditions they can be more robust than parametric algorithms but they are often somewhat sensitive to control variables such as the model order, and tend to give biased results unless these variables are carefully chosen [46, 47].

All pattern recognition algorithms involve two stages: training and recognition. During the training phase of a speech or speaker recognition algorithm, relevant parameters are extracted from several frames of selected speech. In parametric algorithms, the training phase is necessary in order to estimate the statistical parameters necessary to completely specify the model. In non-parametric algorithms, the training set is used either to estimate the underlying probability density functions, or to find the centroids of the various training sets allowing the divisions of the feature space for each class to be established. In each case, the training forms a set of reference parameters which are stored as a template for the particular word or speaker contained in the training set. During the recognition phase of the algorithms, the same parameters are extracted from a test signal and a test template is formed. If the test template is determined to be sufficiently similar to the reference template, a match is declared.

Historically four main pattern matching techniques have been used for speech and speaker recognition applications. Each technique is discussed briefly below. The first three techniques: Temporal Template Matching, Vector Quantization and Neural Networks are non-parametric. The last technique, that of Hidden Markov Modelling, is a parametric technique. In Chapter 4 another parametric technique, the Gaussian Mixture Model, will also be discussed.

Many sophisticated speech recognition machines also use processing after the initial pattern matching step in order to examine the decisions made during the pattern recognition process. Three levels of post processing are used, depending on the application. These post processing levels are lexical processing, syntactic processing, and semantic processing. [48]. Lexical processing is used in word recognition systems which base the recognition of a word on a sequence of smaller recognized patterns such as phonemes. Generally some form of processing is included to determine if the

concatenation of patterns does indeed correspond to a word in the lexicon (dictionary) of the system. Clever lexical processing can be used to reject false concatenations of phonemes without performing an exhaustive search of the dictionary. Lexical processing is often difficult to separate from the pattern matching processing in systems where words form the smallest pattern of interest. Syntactic processing is used in speech to text systems in order to determine if a certain concatenation of words has the proper relationship given the rules of grammar and sentence construction for the language. Semantic processing attempts to combine the pattern recognition techniques and syntactic processing in order to determine the meaning of the spoken sentence. For example, a simple word recognition system would have trouble distinguishing between the phrases “one cent” or “one sent” unless some further syntactic and semantic processing was used [49]. Syntactic and semantic algorithms are often based on Artificial Intelligence programming techniques [9].

2.6.1 Temporal Template Matching

Many word and phoneme recognition systems simply compare temporal templates created by the extraction of relevant parameters over the duration of the utterance. The same technique is also often used in the implementation of text dependent speaker recognition systems. The test template is created by extracting relevant parameters over a sequence of successive frames of speech. Each frame has a duration of approximately 10 - 30 msec, and produces a single vector. The sequence of these vectors forms the test template. The individual elements of the template are modelled as random variables. The reference template is obtained beforehand during the training portion of the algorithm, where a significant amount of averaging is used to form a reference template from the random variables of the extracted parameter vectors.

In many cases, the Euclidean distance measurement of Equation 2.16 is used to determine the similarity between the test and reference templates. If however, the individual elements of the templates are correlated, or exhibit widely different variances, a better distance measurement is the Mahalanobis distance, which is given by [50]

$$d(\vec{r}, \vec{t}) = (\vec{r} - \vec{t})^H \mathbf{W}^{-1} (\vec{r} - \vec{t}) \quad (2.39)$$

where $\vec{r}$ is a column vector containing the individual elements of the reference template.

$\vec{t}$ is a column vector containing the individual elements of the test template.

$\mathbf{W}$ is the autocovariance matrix corresponding to the reference template.

The $\mathbf{W}^{-1}$ term accounts for the correlation between random variables in the templates and ranks the contribution of each random variable to the distance measurement, such that the random variables with lower variance contribute more to the distance metric [6]. In some applications, the correlation between the individual elements of the reference and test templates may be ignored. In these cases, $\mathbf{W}$ is simply replaced by a diagonal matrix, the elements of which are equal to the variance of the individual elements of the reference template.

The main difficulty with the application of temporal template matching is that the duration of vowels or phonemes in conversational speech is rarely a fixed value even if the same phoneme is uttered by the same speaker. If inter-speaker variations are taken into account, the variation in phoneme duration becomes even more significant. A table of phoneme statistics collected from the TIMIT speech corpus is given in Table 2.3. This table was calculated using the phonetic transcription files included in the TIMIT corpus. It is based on over 4.5 hours of prompted speech uttered by 630 different speakers. [14].

Since the length of the temporal templates used in Equation 2.39 are directly proportional to the duration of the utterance, and the vectors used in Equation 2.39 must have the same length, it is difficult to compare two identical utterances which may have different durations. Some form of time normalization must be performed on one template. Simple linear compression, where the endpoints of the test template are forced to align with the endpoints of the reference template, is sufficient for templates representing individual phonemes. For longer templates, those representing words for example, simple linear compression is not sufficient since the relative durations of

Phoneme	Mean Duration (msec)	Standard Deviation (msec)	Max. Duration (msec)	Min. Duration (msec)
i	89.9	96.4	428.2	23.3
I	78.7	83.6	399.1	23.3
e (eI)	128.3	135.13	390.6	31.7
ε	90.6	96.2	281.1	21.4
æ	148.6	155.2	425.0	35.0
α	123.0	129.7	483.4	20.0
ɔ	123.6	131.2	396.6	24.4
o (oU)	126.7	133.0	438.6	34.9
U	76.5	82.1	253.7	18.3
u	107.8	117.8	334.2	24.4
ʌ	89.3	95.9	340.1	18.8
ɜ	118.3	124.6	366.4	30.0
θ	48.5	51.6	173.6	46.3
αI	145.0	154.7	424.4	38.5
ɔI	162.9	167.7	318.5	62.6
αU	163.6	171.6	415.0	62.7
ju	107.8	117.8	334.2	24.4
j	66.7	75.7	231.3	11.4
w	67.4	74.6	216.4	9.5
l	60.5	64.6	229.8	11.9
r	61.0	65.9	180.2	2.3
m	61.2	66.0	229.75	4.3
n	54.3	58.8	212.3	7.8
ŋ	62.6	67.0	240.25	13.5
f	103.0	108.6	305.0	13.3
v	60.0	63.9	190.0	12.0
θ	90.3	97.7	310.0	12.6
δ	37.9	42.6	128.7	4.5
s	113.2	119.4	381.25	23.4
z	84.1	89.7	276.4	18.2
ʃ	116.1	119.8	299.8	24.1
z	83.0	88.3	263.6	22.2
h	65.6	72.2	250.0	13.8
p	44.2	49.3	150.0	6.3
b	17.5	18.9	98.6	2.9
t	48.0	53.7	152.5	4.6
d	23.0	26.1	120.0	2.4
k	51.4	57.3	193.8	6.1
g	30.4	33.1	106.1	7.5
č	86.3	90.6	345.6	20.0
ž	58.7	64.1	231.0	14.9

Table 2.3: Statistics Of English Phonemes

Figure 2.25: Time Warping

phonemes in a word is not constant. From Table 2.3, it may be seen that vowels and nasals tend to have larger durational variances than do most fricatives. Some form of nonlinear compression is therefore necessary in order to align the word templates. The problem is illustrated in Figure 2.25, where the test template must be aligned to the reference template. The algorithm for performing this alignment is known as Dynamic Time Warping, and involves the determination of a re-mapping (warping) function which re-maps the time scale of the test template onto the time scale of the reference template in order to minimize Equation 2.39 [50, 51, 52].

One of the main limitations of temporal pattern matching is that it is limited to text dependent applications where words or phonemes are guaranteed to be repeated in both the training and recognition portions of the algorithm. For text independent applications, such as text independent speaker recognition, other pattern matching techniques are necessary. A second limitation is that Dynamic Time Warping is very computationally intensive. Real time alignment of test and reference templates is not possible.

2.6.2 Vector Quantization

Although discussed in Section 2.4 as a source encoding technique, Vector Quantization is also frequently used for pattern recognition in speech and speaker recognition systems [50, 53]. The process of codebook creation may be viewed as the supervised clustering of training vectors in order to find the centroids corresponding to the various classes of information used in the training data. Codebooks trained with data containing only one speaker will often serve as good reference templates for speaker recognition applications for example. If the same algorithm used to train the codebook is used to extract parameters from a test signal and form a test template, the Euclidean or Mahalanobis distance between the two templates will be smaller for templates corresponding to the same speaker. The advantage of VQ applied to speaker recognition problems is that it is inherently text independent [54]. Similar codebooks and test templates could also be formed for word recognition applications. Since the reference codebook and test template are formed using an iterative procedure over the entire duration of the training and test utterances respectively, there is no need

to align the reference and test templates temporally. The relatively computationally intensive operation of Dynamic Time Warping is therefore eliminated. Unfortunately a large amount of data is typically required to apply an iterative codebook design algorithm. Although this is not generally a problem for the forming of a reference template, during the recognition phase of the algorithm there may not be sufficient data present in the test sequence to form a proper test template.

2.6.3 Neural Networks

Artificial neural network pattern recognition algorithms seek to emulate the recognition capacity of the human brain, by duplicating its structure of many interconnected nonlinear computational elements operating in parallel. Neural network structures are characterized by the nature of the computation performed in the individual elements as well as the pattern of interconnections. The type of neural network most commonly used for speech and speaker recognition applications is based on a simple computational unit known as a *perceptron*. A single perceptron neural network is shown in Figure 2.26. It is used as a decision making system in order to classify an input vector as being from one of two classes (A or B). During the recognition phase of operation, the perceptron takes K weighted inputs, adds them, and applies a non-linear transfer function. At the output of the non-linear transfer function a simple decision is made. In the case of the hard limiter, if the output is over the threshold ϕ then the K -dimensional input vector is assumed to be from class A. If not, the input vector is assumed to be from class B.

The algorithm for training a perceptron is as follows [46]:

Step 1 - Initialize The Weights and Threshold

Set $w_i(0)$ ($0 \leq i \leq K - 1$) and ϕ to small random values.

$w_i(t)$ is the weight from the input node at time t and ϕ is the threshold in the output node.

Step 2 - Present New Input and Desired Output

Present the new values of the input vector

Figure 2.26: Perceptron

$\vec{X}(t) = [x_0(t), x_1(t), \dots, x_{k-1}(t)]$ along with the desired output.

Step 3 - Calculate Actual Output

$$y(t) = f\left(\sum_{j=0}^{K-1} w_j(t)x_j(t) - \phi\right)$$

$f(\xi)$ is one of the nonlinear functions illustrated in Figure 2.26.

Step 4 - Adapt Weights

$$w_i(t+1) = w_i(t) + \eta [d(t) - y(t)] x_i(t), 0 \leq j \leq k-1.$$

$d(t) = +1$ if input vector is from class A

$= 0$ if input vector is from class B.

η is a positive gain fraction less than 1 which is used to control the adaptation rate of the training algorithm. $y(t)$ is the desired output for the input at time t .

Step 5 - Repeat By Going To Step 2

until the weights converge for the given input.

It should be noted that during the training procedure, the weights are only adjusted if an incorrect decision is made by the net. In other words if $d(t) \neq y(t)$. Otherwise the weights remain unchanged.

Given a suitable training sequence, a single perceptron is capable of separating any linearly separable pair of classes. Linearly separable classes are those in which the probability density functions for the multi-dimensional random variables do not overlap, and may be separated by a hyperplane. This is illustrated in Figure 2.27 for a two dimensional space [55]. A hyperplane in two dimensional space is a simple line. The line defined by the weights is also given in Figure 2.27. In general, the weights applied to the K -dimensional perceptron serve to define the position of the hyperplane in multi-dimensional space in a similar manner. The nonlinear operation can also be changed from the hard limiting operation into either a threshold logic or a sigmoid function in order to minimize the mean squared error between the desired output and the true output in the case of overlapping distributions which cannot be completely separated.

Figure 2.27: Two Dimensional Linearly Separable Classes

Figure 2.28: Two Dimensional Linearly Inseparable Classes

The single perceptron neural network may be expanded into a more complex network in order to group input vectors into one of J possible classes. This extension is shown in Figure 2.29, and is commonly referred to as a single layer perceptron network. Each perceptron functions as described previously, inserting a hyperplane into the decision space. A decision is made at each one of these nodes as to the classification of the input vector, relative to its hyperplane. The pattern of 1's and 0's at the output of the single layer network serves to uniquely define the class to which the input vector belongs. The main limitation of this structure is that the separation of classes is still limited to hyperplane separation. For more generalized distributions, more layers are required in order to generalize the shape of the decision surface in the classifier algorithm. The addition of a second perceptron layer allows the generation of multidimensional convex surfaces for the separation of decision spaces. The addition of a third perceptron layer, as shown in Figure 2.30, allows the generation of arbitrarily shaped decision boundaries where the complexity is limited by the number of nodes present in the network. The layers between the input nodes and the output nodes in a multi-layer perceptron network are known as *hidden* layers.

As discussed previously, neural network models are non-parametric pattern classification systems. No assumption is made about the underlying vector probability density functions during the training. This makes them more robust to some classification problems. Under situations where the probability density functions are known, the weights and thresholds in a neural network may be varied in order to adjust the decision boundaries to implement the MAP or ML solutions exactly as described in Equations 2.37 and 2.38.

The main limitation of neural network pattern classifiers is that it is difficult to add another class to the network once it has been trained. This normally involves the addition of more perceptrons at each level, and the re-training of the complete network, using the training data from the new class as well as all previous classes [1].

2.6.4 Hidden Markov Models

Hidden Markov Model (HMM) based pattern recognition techniques are currently used in most of the more successful speech recognition systems [49]. They have

Figure 2.29: Single Perceptron Layer Neural Network ($K=5$, $J=3$)

Figure 2.30: Three Perceptron Layer Neural Network ($K=5$, $J=3$)

proven to be very effective at applications such as word and phoneme recognition, where the speech signal may be modelled as a series of transitions between a set of states [48]. They are not used in speaker recognition applications other than as word or phoneme recognition pre-processors in order to reduce text independent speaker recognition problems to text dependent problems.

The term “Hidden” is used since the exact states themselves are not observable: only the features which characterize the states. HMMs are normally described in terms of a state diagram similar to that shown in Figure 2.31. The HMM is characterized by the total number of states (N_S), a state transition matrix ($A = [a_{ij}]$), and a set of probability density functions $p_i(\vec{X})$.

Each state in the HMM is characterized by a certain set of observable features or feature vectors. These states often correspond to certain sounds or acoustic events. In the case of a word recognition system, the states may correspond to distinct phonemes. In the case of a phoneme recognition system, the states correspond to smaller portions of speech, but it becomes difficult to attach a physical meaning to these states. Associated with each state is a probability density function $p_i(\vec{X})$ which specifies the probability of observing the feature vector $\vec{X}$ while in state i . The probability density functions used in the model may be discrete or continuous. Discrete distribution functions are appropriate in applications where a codebook is used to vector quantize the extracted feature vectors. Continuous probability density functions are appropriate where vector quantization is not used. In these cases a parametric model is assumed for the distribution of the feature vectors in each state. The parameters of the model are estimated during the training phase. Continuous HMMs generally provide better performance for speech recognition applications than the discrete models [6].

The members of the state transition matrix denote the probability of a transition from state i to state j during the observation interval. The parameter a_{ii} denotes the probability that there is no transition during the observation interval. The observation interval is usually related to the frame length in speech processing systems. It is assumed that the probability of transition from state i to state j depends only on the conditions of state i , and not on any previous states. In general a_{ij} can be any number less than 1, with the constraint that $\sum_{j=1}^{N_S} a_{ij}$ is equal to one. For the modelling of

Figure 2.31: Hidden Markov Model State Diagram (5 States)

speech, an additional constraint is often added such that $a_{ij} = 0$ for $j < i$. This particular HMM, known as the Bakis or left right model, is better suited for speech recognition problems since there is a temporal ordering of the states which allows words to be easily represented [56]. The lower numbered states correspond to the states occurring earlier in time. A further restriction often placed on the transitions is that $a_{ij} = 0$ for $j > i + 1$. In other words, only one state may be skipped at any transition in the model. This restriction considerably improves the computational efficiency of HMM calculations, without degrading the performance of the speech recognition [6].

The state transition matrix, as well as the probability density functions for each state allow the HMM to account for effects such as coarticulation, speaker rate, and the effects of different speakers, provided that the model is properly trained [6]. Training the HMM involves the estimation of the model parameters, which includes the state transition matrix, the state probability functions, and any parameters used in a parametric state probability density function. The training procedure is similar to the design of a VQ codebook. An initial estimate is made of the model parameters and then this estimate is refined using an iterative re-estimation algorithm employing the gradient of $Pr[\tilde{\mathbf{X}}|HMM]$ to tune the model parameters. $P[\tilde{\mathbf{X}}|HMM]$ denotes the probability of the occurrence of a given observation vector, given the current values for the HMM. The algorithm for iterating the parameters is known as the Baum-Welch Technique, and is outlined in reference [56].

Two possibilities exist for the HMM modelling of words and phonemes in speech recognition applications. In the first technique, the collection of states is used to form a lattice in which each word or phoneme is characterized by a unique state sequence. During the recognition phase, a numerically efficient algorithm known as the Viterbi Algorithm is used to determine the most likely state sequence for a given utterance [48]. This state sequence uniquely determines the most likely word or phoneme uttered. In most word recognition applications however, the number of states required for this modelling technique results in excessive computational requirements. A more common technique for word recognition applications involves modelling each word in the vocabulary with its own unique HMM consisting of 5-8

states. The recognition operation then involves finding the probability associated with the most likely state sequence through each model using the Viterbi algorithm or a similar technique known as the Forward/Backward procedure and then choosing the word corresponding to the model with the highest probability. In addition to being less computationally intensive, this technique also lends itself to the parallel computation of probabilities for each word model.

As in the case of VQ algorithms, explicit alignment of the temporal features is not required in an HMM. This, and the fact that numerically efficient algorithms exist for the recognition operations, are their main advantages. One limitation of the HMM in the modelling of speech is that a large amount of training data is generally required in order to obtain reasonable estimates for the model parameters. A second limitation of HMMs is that the single order Markov assumption of a state transition probability being dependent only on the current state, and not on any previous states is often invalid, for many words [48]. Higher order Markov models can be used to address this limitation at the cost of increased complexity.

Chapter 3

Wavelets And The Wavelet Transform

3.1 Introduction

Wavelets and the Wavelet Transform are signal processing tools which have grown out the requirement to provide high frequency resolution, while simultaneously providing accurate time isolation of transients in non-stationary signals. As in the case of all time-frequency analysis techniques, wavelet analysis attempts to transform a single dimensional time domain signal $s(t)$ into a two-dimensional function of time and frequency, providing an indication of the distribution signal energy. The two dimensional plane over which this energy distribution is represented is known as the time-frequency plane.

An understanding of non-stationary signal analysis via wavelets begins with a consideration of the standard Fourier Transform:

$$S(\omega) = \mathcal{F} \{s(t)\} = \int_{-\infty}^{\infty} s(t)e^{-j\omega t} dt, \quad (3.1)$$

where $s(t)$ is the time domain signal,

$\mathcal{F}$ is the Fourier Transform operator, and

$S(\omega)$ is the Fourier Transform of $s(t)$, which provides an indication of the distribution of signal energy with frequency for all time.

Equation 3.1 transforms a time domain signal $s(t)$ into the frequency domain by performing an inner product operation between the time domain signal and an infinite set of orthonormal basis functions $\{e^{j\omega t} : -\infty < \omega < \infty\}$. The result is an infinite set of coefficients $\{X(\omega) : -\infty < \omega < \infty\}$ which are used as a unique frequency domain description of the time domain signal. Since the basis functions are orthogonal, the set of coefficients is the minimum set needed to perfectly reconstruct the time domain signal. This reconstruction is accomplished from the infinite summation of the basis functions weighted by the value of Equation 3.1 at each frequency. The result is the common Inverse Fourier Transform operation, which is given by

$$s(t) = \mathcal{F}^{-1} \{S(\omega)\} = \frac{1}{2\pi} \int_{-\infty}^{\infty} S(\omega) e^{j\omega t} d\omega. \quad (3.2)$$

The main limitation of Equation 3.1, when applied to non-stationary signals, is that it provides no time localization for the occurrence of transient events. The integral is evaluated over all of time, so there is no time dependency in the frequency domain coefficients. The standard modification made to Equation 3.1 in order to allow non-stationary signal characterization involves changing the basis functions to a set of functions which are more localized in time. This is usually done by employing a sliding window which allows the Fourier Transform operation to be performed only over a short segment of the time signal as follows [57]:

$$S(\omega, t) = \int_{-\infty}^{\infty} s(\tau) w^*(\tau - t) e^{-j\omega\tau} d\tau, \quad (3.3)$$

where $s(t)$ is the time domain signal,

$w(t)$ is a finite duration window function, and

$S(\omega, t)$ is the Short Time Fourier Transform of $s(t)$, which

provides an indication of the distribution of signal energy over time and frequency.

A discrete time implementation of Equation 3.3 was given in Equation 2.21.

The sliding window introduces a time dependency to the Fourier Transform and the Short Time Fourier Transform (STFT) becomes both a function of time and frequency. One result of this modification is that the set of basis functions $\{w(\tau - t)e^{j\omega\tau}\}$

are no longer orthogonal. The set of coefficients $S(\omega, t)$ derived are somewhat redundant and although perfect reconstruction of the signal from these coefficients is possible, they do not represent the minimum set necessary for signal reconstruction. In the continuous case, the difference between an orthogonal basis set and a non orthogonal basis set is academic since an infinite number of coefficients is always required for perfect reconstruction, but in the discrete time and frequency approximation of Equation 3.3, orthogonal basis functions allow an accurate representation of the signal to be made using fewer coefficients than in the case of non-orthogonal functions.

An equivalent way of viewing Equation 3.3 is that a frequency shifted version of the signal $s(t)$ is convolved with a lowpass filtering function defined by $w^*(-t)$. Properly chosen windows for the STFT result in a lowpass filtering function [58]. The convolution of the signal with this lowpass filtering function determines the contribution of signal energy in a narrow band centered around ω , not just at ω as in the case of Equation 3.1. The time localization provided by Equation 3.3 therefore sacrifices the infinite frequency resolution of Equation 3.1.

The STFT therefore segments the time frequency plane into a series of bins through which the energy of the time varying signal may be estimated, as shown in Figure 3.1. Two important parameters are required to characterize this segmentation: the time resolution (Δt) and the frequency resolution ($\Delta \omega$). The time resolution of the transform specifies the ability of two closely spaced transient events to be separated in time. For example, the occurrence of two delta functions spaced less than Δt units apart could not be distinguished using Equation 3.3. Similarly, the frequency resolution specifies the ability of the Short Time Fourier Transform to separate the frequencies of two closely spaced sinusoids.

The exact values of Δt and $\Delta \omega$ are dependent on the window function chosen, but Δt is roughly proportional to the time duration of the window function. $\Delta \omega$ is proportional to the bandwidth of $W(\omega)$, which varies inversely with the duration of the window function. As a result, time resolution and frequency resolution cannot be independently chosen. According to the Heisenburg Uncertainty Principle, the time-bandwidth product $\Delta t \Delta \omega$ of the Short Time Fourier Transform is constrained

Figure 3.1: Segmentation Of Time Frequency Plane

such that [57]

$$\Delta t \Delta \omega \geq \frac{1}{2} \quad (3.4)$$

for any choice of window function $w(t)$. Time resolution may therefore only be enhanced at the cost of a poorer frequency resolution, and visa versa. The Fourier Transform operation of Equation 3.1 is at one extreme of this trade-off, since it provides an infinite frequency resolution, but no time resolution.

The Wavelet Transform (WT) is designed to overcome the time frequency resolution trade-off inherent in the STFT by letting Δt and $\Delta \omega$ vary in the time frequency plane, as shown in Figure 3.1. Different windows are chosen for various positions in the time frequency plane, with the area ($\Delta t \Delta \omega$) being constant. As shown in Figure 3.1, the bandwidth of the filters at low frequencies is low, allowing for an accurate frequency estimation to be made. At high frequencies, the bandwidths are wide but the time resolution is small allowing for an accurate isolation of time transients. The ability of the WT to accurately isolate the occurrence of transients is illustrated in Figure 3.2, where a STFT is compared with a WT for the analysis of a delta function. In this case, the time waveform contains an impulse at $t = 64$ msec. The entire data frame is 0.128 msec in length, sampled at an 8.0 kHz rate for a total of 1024 points. The STFT based spectrogram plot was calculated by segmenting the 1024 points into frames of 64 points each. A 32 point overlapping was used between each frame. Each frame was then windowed with a Hanning window and the frame FFT was calculated. The spectrogram was then calculated using the magnitude squared of the frame FFTs. This technique provides the spectrogram with a frequency resolution of approximately 166 Hz (due to the Hanning window) and a time resolution of 8.0 msec. The wavelet transform scaleogram was calculated by doing a 5 level transform on the 1024 point set using the 4 tap Daubechies filter coefficients. The details of the WT calculation will be discussed in Section 3.2. As may be seen from Figure 3.2, the WT provides a much more accurate time localization of the delta function than the STFT.

In Figure 3.3 two sinusoids of equal power are analyzed using a WT and a STFT. The first tone occurs at $t = 25.135$ msec and has a frequency of 160.0 Hz. The second tone occurs at $t = 56.25$ msec and has a frequency of 1600.0 Hz. The combined signal

Figure 3.2: STFT and WT Analysis Of Delta Function

Figure 3.3: STFT and WT Analysis Of Sinusoid Pair

is sampled at a rate of 8.0 kHz. Both the spectrogram and the scaleogram of Figure 3.3 were calculated as in Figure 3.2. Although the wavelet scaleogram does provide a better time isolation for the start of the higher frequency tone, the frequency estimate is not as precise as the estimate provided by the spectrogram. The lower frequency tone is evidenced by the dark segments in the $f = 231.0$ Hz band of the scaleogram, but there is leakage into the adjacent band at $f = 78.0$ Hz. The frequency scale for the WT scaleogram will be discussed in Section 3.4. Time resolution in the two lower bands is approximately equal to the time resolution of the spectrogram. It may also be seen from Figure 3.2 that interpretation of a wavelet based scaleogram is more difficult than the interpretation of a spectrogram due to the nonlinear frequency scale, and the variations in intensity. Since there is less temporal averaging involved in a WT, there tends to be a wider variation in the intensity displayed in active bands than in the case of a spectrogram. It can also be seen that signals of equal power may not be displayed with the same intensity in a wavelet based scaleogram. The reason for this is discussed in Section 3.2.

In general, wavelet based signal processing techniques are applicable to signals which are[59]:

- a) composed of time-varying tones of finite duration where
 - the durations of the tones vary as a function of the frequency
 - such that higher frequency tones have a shorter duration,
- b) transient-like in behavior, with only a few cycles present, or
- c) chirp-like, where the frequency variation within the chosen window exceeds the frequency resolution of the window.

Such signals are often encountered in practice, in applications ranging from radar signal processing to image compression. In the case of radar signal processing, wavelet based analysis techniques have been shown to be more appropriate than the STFT for the processing of wideband radar returns from fast moving targets [60], or from targets where the radar return is intermittent due to a fluctuating reflector [59]. Wavelet based techniques are also showing promise for the reduction of background clutter in image processing for target recognition purposes [61]. In the case of many

transient signals wavelet techniques also offer the ability to accurately characterize the signal using fewer coefficients than a STFT [62]. This makes the WT an excellent tool for data compression applications. In image compression applications, where edges have the effects of transients in the image data, wavelet based techniques are able to compress moving pictures by a factor of one hundred and fifty, with the regenerated images being almost indistinguishable from the originals [63]. A wavelet based standard developed by the Los Alamos National Laboratory has also been adopted by the United States Federal Bureau of Investigation for the compression and digital storage of fingerprint images [64]. Compression ratios in this application are in the order of twenty to one [62].

Speech signals satisfy the first two characteristics of signals suitable for processing with wavelet based techniques. It was seen in Section 2.5 that the conflicting requirements of time and frequency resolution in the STFT result in the requirement for both wideband and narrowband spectrograms in many speech processing applications. Wideband spectrograms allow for the accurate processing of formant information, whereas the narrowband spectrogram is more suitable for the processing of pitch information. Wavelet transform techniques offer the potential of simultaneously satisfying both requirements [37]. Wavelets have been successfully used in speech processing systems for the measurement of pitch frequencies [65]. In this application they offer advantages over the classical techniques discussed in Section 2.5 in that they allow more accurate measurement of the changes in pitch which occur during the transition between phonemes. This information has been shown to be very useful for speaker recognition applications [2]. Wavelet based pitch extraction techniques also appear more robust than classical pitch estimation algorithms in the presence of noise and perform well over a wider range of pitch periods. Wavelets are also being investigated for speech coding applications and have achieved data rates as low as 4.5 kbps with good quality [66, 67].

3.2 The Wavelet Transform

The Continuous Wavelet Transform (CWT) of a time domain signal $s(t)$ is defined as:

$$\tilde{S}_C(t, \alpha) = \int_{-\infty}^{\infty} s(\tau) \frac{1}{\sqrt{|\alpha|}} \psi^* \left(\frac{\tau - t}{\alpha} \right) d\tau \quad (3.5)$$

where $\tilde{S}_C(t, \alpha)$ is the CWT of the time domain signal $s(t)$ and $\psi(t)$ is a mother wavelet function.

As in the case of the STFT, the CWT is characterized by an infinite set of coefficients calculated from the inner product of the signal $s(t)$ with a set of basis functions. In the case of the CWT, the basis functions are derived from a single prototype function known as a *mother wavelet* function through shifting and scaling operations. The time shifts are characterized by the parameter τ , and the scaling operations are characterized by the α parameter. A value of α greater than 1 causes an expansion of the mother wavelet function. A value of α less than 1 causes a contraction of the function, as shown in Figure 3.4. Valid values of α are $0 < \alpha < \infty$. The factor of $\frac{1}{\sqrt{|\alpha|}}$ is added to the scaled function in order to ensure that the energy of the scaled function is the same as the energy of the original mother wavelet. All valid mother wavelets must obey the following admissibility criterion [68, 69]:

$$C_g = \int_{-\infty}^{\infty} \frac{|\Psi(\omega)|^2}{|\omega|} d\omega < \infty, \quad (3.6)$$

where C_g is called the constant of admissibility, and

$\Psi(\omega)$ is the Fourier Transform of the mother wavelet function $\psi(t)$.

Generally the mother wavelet is normalized such that $C_g = 1$.

Properly chosen mother wavelet functions also have a property known as *compact support*, meaning that the function is non-zero only over a finite time interval. This property ensures that the basis functions are localized in time, unlike in the case of a Fourier Transform where the complex exponential basis functions extend over an infinite time window. This time localization property of the wavelet transform allows the isolation of transients as in case of the STFT. One of the advantages of the

$$\psi_{\alpha}(t, \tau) = \frac{1}{\sqrt{|\alpha|}} \psi\left(\frac{t-\tau}{\alpha}\right)$$

$$\psi(t) = \frac{2}{\sqrt{3}} \pi^{\frac{1}{4}} (1-t^2) e^{-\frac{t^2}{2}}$$

Figure 3.4: Effects Of Shifting and Scaling Operations On Mother Wavelet Function

Wavelet Transform (WT) over the STFT is that under certain conditions the wavelet basis functions may be chosen such that orthogonality is maintained over the finite time interval. Although in the case of the CWT, this orthogonality is impossible to maintain over a finite time window for a continuous range of time shifts and scaling operations, it is relatively easy in the discrete case as will be seen shortly. Another interesting feature of the WT is that, unlike in the case of the STFT, the basis functions may be chosen to be real rather than complex. This can simplify processing requirements in the case of the wavelet transform. Some common mother wavelet functions for the CWT are illustrated in Figure 3.5.

The CWT is most often displayed in a manner analogous to a spectrogram display, where $|\tilde{S}(\alpha, t)|^2$ is displayed above a time-scale plane. The term *scaleogram* is used to describe this wavelet derived counterpart of a spectrogram. Since the CWT is an isometric transformation, it preserves the energy of the signal such that [70]

$$E_s = \int_{-\infty}^{\infty} |s(t)|^2 dt = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} \frac{|\tilde{S}_C(t, \alpha)|^2}{\alpha^2} dt d\alpha. \quad (3.7)$$

The parameter $\frac{|\tilde{S}(t, \alpha)|^2}{\alpha^2}$ is therefore an approximation of the signal energy contained in a small region on the time scale plane surrounding the point (t, α) .

As may be seen, the CWT is strictly speaking not a function of time (t) and frequency (f), but is a function of time (t) and scale (α). It is the scaling parameter which is used to vary both time and frequency resolution of the CWT across the time frequency plane. Equation 3.5 is the convolution of the time domain signal $s(t)$ with a filtering function defined by $\psi^*\left(\frac{t}{\alpha}\right)$. As in the case of the STFT, the shifting parameter in the CWT merely serves to isolate a small segment of the signal for application of the integral operation. The length of the time segment processed by the integral is dependent on the mother wavelet chosen and the scale parameter α as shown in Figure 3.4. For large values of α the duration of the time window is long. For small values of α the duration is short. In the frequency domain since [21]

$$\mathcal{F}\left\{\psi\left(\frac{t}{\alpha}\right)\right\} = |\alpha| \Psi(\alpha\omega), \quad (3.8)$$

it may be seen that the opposite effect occurs. As α increases, the bandwidth of the filtering function defined by the Fourier Transform of $\psi^*\left(\frac{t}{\alpha}\right)$ decreases.

Figure 3.5: Continuous Wavelet Transform Mother Wavelet Functions

Unlike the lowpass windowing functions used in a STFT, proper mother wavelets are always bandpass functions. The shifting operation therefore tends to move the center frequency of the bandpass filtering function for every scale shift α . As the filter center frequency is moved up in frequency, its bandwidth becomes proportionately wider. As α increases the filter center frequency is reduced and the bandwidth becomes narrower. Thus, the wavelet transform may be viewed as a bank of bandpass filters of varying center frequency and bandwidth. Since the bandwidths and center frequencies both move in direct proportion to the scale, the filter bank exhibits a constant Q characteristic such that

$$\frac{\Delta\omega}{\omega_c} = \text{constant} \quad (3.9)$$

where $\Delta\omega$ is the filter bandwidth for each filter in the bank, and

ω_c is the bandpass filter center frequency.

As discussed in Section 2.5, the response of the inner ear to audio signals may be modelled using a similar type of filter bank. The constant Q property of the WT leads to a segmentation of the time-frequency plane similar to that shown in 3.1. The center frequencies of the bandpass filters used in the calculation of the wavelet transform may be used to calculate a frequency scale for the scaleogram. This technique is discussed in Section 3.4.

Another implication of the fact that properly chosen mother wavelets are bandpass and not lowpass functions is that the wavelet transform is insensitive to any DC component contained in the signal. Assuming that there is no DC component in the signal $s(t)$ and that the mother wavelet function satisfies Equation 3.6, $s(t)$ may be reconstructed from its CWT via the Inverse CWT relationship[69]

$$s(t) = \frac{1}{C_g\alpha^2} \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} \tilde{S}_C(\tau, \alpha) \frac{1}{\sqrt{|\alpha|}} \psi\left(\frac{t-\tau}{\alpha}\right) d\tau d\alpha. \quad (3.10)$$

Most practical implementations of the CWT involve optical processing [71, 72]. For digital implementations of the Wavelet Transform, it is necessary to discretize both time and scale (frequency). Development of the Discrete Wavelet Transform (DWT) begins by sampling the scale and delay parameters. The parameters t and α

are discretized such that $\alpha = \alpha_0^j$ and $t = i\alpha_0^j T_s$. When substituted into Equation 3.5 the result is

$$\tilde{S}_{DP}(i, j) = \int_{-\infty}^{\infty} s(\tau) \frac{1}{|\alpha_0|^{j/2}} \psi^* \left(\frac{\tau - i\alpha_0^j T_s}{\alpha_0^j} \right) d\tau. \quad (3.11)$$

Equation 3.11 is known as the Discrete Parameter Wavelet Transform [73]. Generally a value of $\alpha_0 = 2$ is chosen. The resulting WT then only employs one contracted mother wavelet as a basis function. All other basis functions are power of 2 expansions of the mother wavelet. The scaling parameter is said to be *diadically* sampled.

Once the parameter α_0 is fixed, it becomes possible to select wavelet functions which are orthogonal. In other words

$$\int_{-\infty}^{\infty} \psi(2^{-j_1} \tau - i_1 T_s) \psi^*(2^{-j_2} \tau - i_2 T_s) d\tau = 0, \quad (3.12)$$

unless $i_1 = i_2$ and $j_1 = j_2$. All mother wavelet functions shown in Figure 3.5 satisfy this property when sampled diadically with $T_s = 1.0$. Orthogonal wavelets allow the representation of non-stationary functions using as few wavelet transform coefficients as possible. They are most appropriate for the characterization of signals spanning several octaves in frequency. Non-orthogonal wavelets, because of the redundancy they provide, are more appropriate in applications where a large amount of detail is required within a single octave of frequency [60].

Although the DPWT is discrete in time and scale, implementation of Equation 3.11 involves a continuous variable of integration (τ) and therefore implies that $s(t)$ must be a continuous function of time. For a totally digital implementation of the WT, it is necessary to sample $s(t)$. This results in the Discrete Wavelet Transform (DWT), which may be written as

$$\tilde{S}(i, j) = \sum_{k=-\infty}^{\infty} s(kT_s) \frac{1}{2^{j/2}} \psi^* \left(\frac{kT_s - i2^j T_s}{2^j} \right). \quad (3.13)$$

Equation 3.13 shares most of the properties of Equation 3.5. In some cases the mother wavelet used in Equation 3.13 is simply a sampled version of one of the mother wavelet functions given in Figure 3.5. In most practical implementations of the DWT however, the mother wavelet used in Equation 3.13 does not have a continuous

time functional representation. The continuous time function is constructed using an iterative technique which will be discussed in Section 3.3. One example of such a mother wavelet function is the Daubechies₄ wavelet function which is shown in Figure 3.6 [74].

An important observation to be made regarding the DWT is that for every value of j , Equation 3.13 is the discrete time convolution of the signal $s(nT_s)$ with a filtering function given by $\psi\left(\frac{-nT_s}{2^j}\right)$, where the convolution operation followed by a decimation of 2^j . This observation will become important in Section 3.3.

3.3 Filterbanks and The Fast Wavelet Transform

The development of a computationally efficient implementation of the DWT begins with a consideration of the following iteration [75, 57]:

$$G_k(z) = \prod_{l=0}^{k-1} G(z^{2^l}) \text{ for } k = 1, 2, \dots, \quad (3.14)$$

where $G(z)$ is the z -Transform of a lowpass filtering function $g(n)$.

If $G(z)$ is chosen to be a lowpass filter function, each iteration of Equation 3.14 results in a lowpass filter of half the bandwidth of the previous filter. For example, if $g(n)$ is chosen to be a simple 4 tap FIR filter defined by the following coefficients:

$$g(n) = \left[\frac{1 + \sqrt{3}}{4\sqrt{2}}, \frac{3 + \sqrt{3}}{4\sqrt{2}}, \frac{3 - \sqrt{3}}{4\sqrt{2}}, \frac{1 - \sqrt{3}}{4\sqrt{2}} \right], \quad (3.15)$$

then its frequency domain transfer function is as shown in Figure 3.7. In the iteration of Equation 3.14, $G_1(z)$ is equal to $G(z)$. The transfer function of $G(z^2)$ is simply the original lowpass function compressed and repeated in the frequency domain. As may be seen from Figure 3.7 when $G(z)$ is cascaded with $G(z^2)$ the result, $G_2(z)$, is a lowpass filter of half the bandwidth of $G_1(z)$. At each stage of the iteration, the number of taps in the filter increases such that

$$L_k = [2^k - 1] L - 2^k + 2. \quad (3.16)$$

Figure 3.6: Daubechies_4 Mother Wavelet Function

Figure 3.7: Iterated Lowpass Transfer Functions

where L_k is the number of taps in the filter $g_k(n)$, and

L is the number of taps in the original lowpass filter $g(n)$.

At each stage of the iteration, an equivalent set of halfband highpass filters may be obtained by the following transformation:

$$h_k(n) = (-1)^n g_k(L_k - 1 - n), \quad (3.17)$$

where $h_k(n)$ are the highpass filter coefficients for $0 \leq n \leq L_k - 1$, and

$g_k(n)$ are the lowpass filter coefficients for $0 \leq n \leq L_k - 1$.

By appropriately cascading scaled versions of the halfband highpass function with a set of cascaded lowpass functions, as shown in Figure 3.8, a series of constant Q bandpass filters may be obtained. The network is defined such that

$$\begin{aligned} P_1(z) &= H(z) \\ \text{and} \\ P_k(z) &= \prod_{l=0}^{k-2} [G(z^{2^l})] H(z^{2^{k-1}}) \text{ for } k = 2, 3 \dots O, \end{aligned} \quad (3.18)$$

where O is the maximum number of cascaded filtering stages in the network.

This is also known as the order of the Fast Wavelet Transform.

$P_k(z)$ is the bandpass transfer function at the k th stage.

The network of bandpass filters covers almost all of the range of frequencies from 0 to $\frac{f_s}{2}$, where f_s is the system sampling frequency. The addition of a final branch $G_N(z)$, as shown in Figure 3.8, completes the coverage from 0 to $\frac{f_s}{2}$ with a lowpass function. If, for example, the lowpass filter coefficients of Equation 3.15 are used in the iteration of Equation 3.18, the filter responses of the resulting filters are as shown in Figure 3.9 for a network with $O = 5$. A sampling frequency of 8.0 kHz was assumed.

If the coefficients of the original lowpass filter $g(n)$ are chosen such that

$$\sum_n g(n)g(n - 2l) = 1 \text{ for } l = 0, \text{ and } 0 \text{ otherwise,} \quad (3.19)$$

and

$$\sum_n g(n) = \sqrt{2} \quad (3.20)$$

Figure 3.8: Constant Q Bandpass Filter Network ($O=5$)

Figure 3.9: Daubechies_4 Constant Q Network Responses ($O=5$)

then the impulse responses of the bandpass filters in the network eventually converge to be scaled versions of a prototype function. The prototype function to which the network converges may be determined by forming a piecewise function $f_k(t)$ such that

$$f_k(t) = \frac{1}{\sqrt{2^k}} p_k(n) \quad (3.21)$$

on the interval $\frac{nT_s}{2^k} \leq t < \frac{(n+1)T_s}{2^k}$ for $0 \leq n \leq L_k - 1$. $f_k(t)$ is supported on the interval $0 \leq t \leq [L - 1]T_s$ and for large values of k , $f_k(t)$ converges to a continuous function $\psi(t)$ [57, 75]. $p_k(n)$ is the impulse response of the filter $P_k(z)$.

The convergence of the network so that the bandpass filter impulse functions are merely scaled versions of a prototype function is evident in Figure 3.10, where the functions $p_k(nT_s)$ resulting from the coefficients of Equation 3.15 are plotted for $1 \leq k \leq 5$. It can be shown that at higher values of k , the bandpass impulse functions satisfy the following equation:

$$p_k(n) \approx \frac{1}{\sqrt{2^k}} \psi\left(\frac{nT_s}{2^k}\right). \quad (3.22)$$

where $\psi(t)$ is the Daubechies_4 mother wavelet function shown in Figure 3.6.

A similar convergence also occurs for the lowpass branch of Figure 3.8. If $g_k(n)$ is substituted for $p_k(n)$ in Equation 3.21, the resulting function also converges to a continuous function as k increases. The function, $\phi(t)$, to which the piecewise function $f_k(t)$ converges in this case is known as the *scaling function* associated with the mother wavelet. The scaling function associated with the Daubechies_4 mother wavelet function is illustrated in Figure 3.11.

It is the iterative technique outlined in Equation 3.18 which is used to construct the mother wavelet functions used in most DWT implementations. An example of a second set of transfer functions is illustrated in Figure 3.12. These functions were generated using the twelve tap Daubechies lowpass filter which is defined by [76]

$$\begin{aligned} g(n) = & [0.111540743, 0.494623890, 0.751133908, 0.315250352, \\ & -0.226264694, -0.129766868, 0.097501606, 0.027522866, \\ & -0.031582039, 0.000553842, 0.004777258, -0.001077301]. \end{aligned} \quad (3.23)$$

Figure 3.10: Filter Impulse Functions For Daubechies-4 Constant Q Network

Figure 3.11: Scaling Function For Daubechies_4 Mother Wavelet

Figure 3.12: Daubechies_12 Network Responses ($O=6$)

It may be shown that for any set of lowpass coefficients satisfying Equations 3.19 and 3.20, the wavelet function $\psi(t)$ and scaling function $\phi(t)$ produced will be orthogonal [75]. It may also be shown that $\psi(t)$ is orthogonal to all scaled and shifted images of itself which occur along the dyadic grid. The same is true of $\phi(t)$.

The convergence of the filter bank shown in Figure 3.8 to a set of scaled wavelet functions provides a technique for the implementation of the DWT. As discussed in Section 3.2, the DWT function of Equation 3.13, at a given value of j , is the discrete time convolution of the input signal $s(nT_s)$ with a filtering function defined by $\psi\left(\frac{-nT_s}{2^j}\right)$, followed by a 2^j decimation. Equation 3.13 may therefore be implemented by the network shown in Figure 3.13, where the filter coefficients are simply the time reversed values of the filters used in Figure 3.8.

$$\tilde{g}_j(n) = g_j(L_j - 1 - n) \text{ for } 1 \leq j \leq O \text{ and} \quad (3.24)$$

$$\tilde{h}_j(n) = h_j(L_j - 1 - n) \text{ for } 1 \leq j \leq O. \quad (3.25)$$

This time reversal accounts for the time reversal of $\psi\left(\frac{-nT_s}{2^j}\right)$ in the convolution of Equation 3.13.

The addition of the final lowpass branch to Figure 3.13 represents the scaling function, and allows the representation of a DC component in the DWT, which Equation 3.13 does not allow. Most implementations of the DWT do include this final scaling function. The DWT implemented by the network of Figure 3.13 is therefore given by

$$\begin{aligned} \tilde{S}(i, j) &= \sum_{k=-\infty}^{\infty} s(kT_s) \frac{1}{2^{j/2}} \psi^* \left(\frac{kT_s - i2^j T_s}{2^j} \right) \text{ for } 1 \leq j \leq O \text{ and} \\ \tilde{S}(i, O+1) &= \sum_{k=-\infty}^{\infty} s(kT_s) \frac{1}{2^{N/2}} \phi^* \left(\frac{kT_s - i2^N T_s}{2^N} \right). \end{aligned} \quad (3.26)$$

Although Figure 3.13 does represent a practical implementation of the DWT, it is not computationally efficient since the filtering is all done at the highest sampling rate, and the decimation takes place at the end. It may be shown that a filtering operation using an arbitrary FIR function defined by $\Upsilon(z^{2^j})$, followed by a decimation of 2^j is equivalent to a predecimation by 2^j , followed by a filtering function defined by $\Upsilon(z)$. This network equivalence is illustrated in Figure 3.14. Applying this equivalence to

Figure 3.13: DWT Implementation ($O = 5$)

Figure 3.14: Equivalent Networks

Figure 3.15: Fast Wavelet Transform ($O = 5$)

the network shown in Figure 3.13 yields the more efficient DWT implementation of Figure 3.15, which is commonly known as the Fast Wavelet Transform (FWT).

As can be seen the basic highpass and lowpass filtering functions $\tilde{H}(z)$ and $\tilde{G}(z)$ are used at each stage of the FWT operation. Unlike the network shown in Figure 3.13 the filters do not increase in size at each stage. Assuming that each filter shown in 3.15 has L taps, each output sample requires on the order of L operations for every input sample. Due to the decimations however fewer samples are presented to the filters which are deeper in the chain. The total number of operations for a given number N of input samples is therefore upper bounded by $2LN$ since

$$\text{Total Operations} = LN + \frac{LN}{2} + \frac{LN}{4} + \dots < 2LN. \quad (3.27)$$

It is interesting to note that this is independent of the number of stages contained in the DWT. The fact that the DWT calculation is directly proportional to the number of input samples N makes its calculation more efficient than the calculation of the FFT which requires on the order of $N \log_2 N$ operations.

3.4 Practical FWT Implementation Issues

Although Figure 3.15 does represent a computationally efficient implementation of a Discrete Wavelet Transform, and is used in practice, some implementation issues must be considered before any interpretation of the results can be made. First of all, for a wavelet order of O in Figure 3.15 it is clear that the minimum number of points required at the input of the filter bank is 2^O since this results in at least one point at the output of the final two filtering stages once all the decimations have been performed. The minimum frame length which may be evaluated by a wavelet transform of order O is therefore given by $N = 2^O$. Higher frame lengths may be evaluated, but some order of two relationship must be maintained so that the decimation is performed properly. In general, for a frame length of $N = B2^{O'}$, an FWT order of O' or less is required to correctly process the frame. For an order O' transform, the number of points at the output of the final filtering stage is given by B [77]. Generally the output of the FWT network is packed into a single $N \times 1$ vector

as shown in Figure 3.16.

Most practical implementations of the DWT employ a circular, rather than linear, convolution for the filters shown in Figures 3.15 and 3.16. This involves a periodic extension of the input signal past its boundaries, as well as a periodic extension of the filter impulse responses. The extension is done in order to maintain the orthogonality of the transform thereby ensuring that it is invertible by a conjugate network normally used to define the Inverse Wavelet Transform (IWT) operation [78]. Direct application of the network shown in Figure 3.15 on a finite length signal will result in small errors at the signal boundaries if the signal is reconstructed using an IWT [77]. Periodic extension of the input corrects this problem allowing for perfect reconstruction of the signal over its complete span.

The initial point at which the decimation begins for the filters in an FWT network must also be considered if the FWT is to be properly inverted. In the *MATLAB* implementation of the FWT for example, if the length of the filters $G(z)$ and $H(z)$ is assumed to be L , all decimations begin following a delay of $L - 1$ points which allows the filter transient response to die out for the periodic input [76]. A second common implementation of the FWT, which is found in the book *Numerical Recipes*, uses a delay of $L/2 + 1$ points [79]. The implication of this difference is illustrated in Table 3.1, where the FWT of a simple 32 point impulse function is analyzed using an FWT of order 5, initialized with the twelve tap Daubechies lowpass filter defined by Equation 3.23. As can be seen the results are slightly different. Each FWT will lead to perfect reconstruction of the original delta function but the IWT must account for the initial decimation point. Therefore, the correct IWT routine must be used, and the *MATLAB* IWT routine will not give proper results if it is used with the *Numerical Recipes* generated FWT.

The differences in delays at each level of the network shown in Figure 3.15 result in the requirement to time align wavelet coefficients properly if they are to be used in a scaleogram plot. For an input signal of length $N = B2^O$ points, processed with an FWT of order O , the number of points output from each level of the network shown in Figure 3.15 is

$$N_j = B2^{O-j} \text{ for } 1 \leq j \leq O \text{ and} \quad (3.28)$$

Figure 3.16: FWT Coefficient Packing: $N=32$, $O=5$

Coefficient Number	MATLAB FWT Coefficient Value	Numerical Recipes Coefficient Value
0	0.199223	0.288909
1	0.050777	-0.038909
2	0.183146	0.235844
3	-0.132899	0.093134
4	0.091585	-0.057454
5	-0.049134	-0.000330
6	0.027281	0.135680
7	-0.297245	0.149617
8	-0.000653	-0.090900
9	-0.030085	0.026235
10	-0.27781	-0.003283
11	-0.554668	-0.000653
12	0.224255	-0.030085
13	-0.090900	-0.277810
14	0.026235	-0.554668
15	-0.003283	0.224255
16	-0.001077	0.027523
17	0.000000	0.000554
18	0.000000	-0.001077
19	0.000000	0.000000
20	0.000000	0.000000
21	0.000000	0.000000
22	0.000000	0.000000
23	0.000000	0.000000
24	0.000000	0.000000
25	0.000000	0.000000
26	0.000000	0.000000
27	0.494624	0.000000
28	0.315250	0.000000
29	-0.129767	0.494624
30	0.027523	0.315250
31	0.000554	-0.129767

Table 3.1: Differences Between Wavelet Coefficients

$$N_{0+1} = B \text{ if the scaling function output is included.} \quad (3.29)$$

Assuming that MATLAB decimation convention is used, the last $N_j - 1$ points from each row are used to fill the corresponding scaleogram row first, followed by the remaining points [80]. This ensures proper time alignment of the transient events in each row of the scaleogram if it is displayed as in Figures 3.2 and 3.3. In the case of a 32 point signal frame analyzed using an order 5 FWT, the packing of the scaleogram is as shown in Figure 3.16.

The default ordinate axis for a wavelet based scalogram is the scale parameter. In order to convert this scale axis to frequency, the center frequency of the equivalent bandpass digital filters composing each branch of the network must be determined. Using the transfer functions shown in the form of Figure 3.13, the center frequency may be determined by normalizing the square of the magnitude function and then treating the result as a probability density function of frequency over the range 0 to $f_s/2$. The mean frequency determines the center frequency of the filter and may therefore be determined from the following equation:

$$f_{center} = \frac{\int_0^{\pi f_s} \omega |H(e^{-j\omega T_s})|^2 d\omega}{2\pi \int_0^{\pi f_s} |H(e^{-j\omega T_s})|^2 d\omega}. \quad (3.30)$$

Chapter 4

Gaussian Mixture Modelling

4.1 Introduction

As mentioned in Section 2.6, pattern recognition techniques may be divided into two main categories: parametric and non-parametric. This chapter discusses the Gaussian Mixture Model (GMM), which is a parametric modelling technique recently applied to speaker identification [1]. Each speaker is modelled using a unique GMM, defined as the weighted sum of component Gaussian probability density functions (pdfs). The weighted sum has the advantage that it allows a more accurate modelling of feature vectors generated from speech data than does a single pdf. As long as the underlying statistics are not multi-modal, the GMM will generally represent the data fairly accurately [1].

Each component density in the GMM is used to model the distribution of feature vectors corresponding to an *acoustic class*. Each acoustic class is representative of a group of phonetically similar events such as vowels, nasals, or fricatives. Since the number of acoustic classes used in the GMM for each speaker is significantly less than the number of phonemes in the English language, there is not a one to one correspondence between phonemes and acoustic classes. Instead, each acoustic class corresponds to several phonemes. The incorporation of acoustic classes into a speaker identification system allows the model to account for text variation in the speech data. This allows the speaker specific characteristics to be compared over

similar acoustic classes as they occur in unconstrained speech, thereby making the model more suitable for text independent applications.

The recognition or identification phase of the GMM involves the calculation of a simple set of likelihood functions using the test data and the GMM for each speaker. The model which results in the largest likelihood function indicates the most likely speaker for the test data. For the closed set speaker identification problem implemented in Chapter 5 it is not necessary to threshold the likelihood functions and allow for the possibility that the speaker is not in the database. In general however such thresholding is necessary but it requires a very accurate model of the speaker in order to obtain good identification rates.

During the training phase of the GMM, parameters are estimated from the training data using an iterative procedure which refines each parameter estimate until a given convergence threshold is reached. It is important to realize that the training of the GMM does not involve a supervised transcription of the acoustic classes contained in the training data. Since the GMM models the occurrence of an acoustic class using a set of probabilities, the parameters related to this assumption are developed during the training in a manner similar to the unsupervised clustering of data in a VQ algorithm. As will be seen, the only information which is required during the GMM training phase is the name of the speaker corresponding to the training data. The development of the GMM as well as the discussion of the identification and training aspects discussed in this chapter have been taken largely from reference [1].

4.2 Model Development

For a closed set speaker identification application, it is assumed that a set of feature vectors characterized by the K -dimensional random vector $\vec{X}$ are available for both training and identification phases of operation. These feature vectors are extracted from a series of consecutive frames of speech which are created through the use of a sliding window as in Equations 2.19 and 2.21. For speaker identification purposes, K generally ranges from 10 to 30, depending on the exact parameters extracted from each frame of speech. Silence intervals are removed using a preprocessor so that the

parameter vectors reflect only active speech intervals. From each frame of speech, a feature vector consisting of time or frequency domain parameters is extracted and used in the pattern analysis as shown in Figure 2.24.

The development of the GMM model begins with a modelling of the various acoustic classes present in a sample set of speech from a given speaker. Each acoustic class is assumed to be representative of a small set of acoustically similar phonemes. It is assumed that the distribution of the random feature vector $\vec{X}$ for a given speaker and acoustic class may be described in terms of a multivariate Gaussian pdf:

$$p_{\vec{X}|I}(\vec{x}|i, \lambda_j) = \frac{1}{(2\pi)^{\frac{K}{2}} |\Sigma_{ij}|^{\frac{1}{2}}} e^{-\frac{1}{2}(\vec{x}-\vec{\mu}_{ij})^H |\Sigma_{ij}|^{-1} (\vec{x}-\vec{\mu}_{ij})} \quad (4.1)$$

where $p_{\vec{X}|I}(\vec{x}|i, \lambda_j)$ is the probability density for the random feature vector

$\vec{X}$ assuming acoustic class i and speaker model λ_j .

$\vec{x}$ is the specific occurrence of the random vector $\vec{X}$, which is represented as a K -dimensional column vector.

H denotes a Hermitian transpose operation.

$\vec{\mu}_{ij}$ is a column vector representing the mean of the random feature vector $\vec{X}$ for acoustic class i and speaker model λ_j .

Σ_{ij} is the $K \times K$ covariance matrix for the random vector $\vec{X}$ assuming acoustic class i and speaker model λ_j .

The model for the generation of feature vectors, assuming speaker model λ_j , is as illustrated in Figure 4.1. At each vector observation period, one of M acoustic classes is chosen according to a set of defined probabilities $\{P_i\}_{i=1}^M$. The parameter M is known as the GMM model order, and must be chosen carefully. A value of M which is too low will not allow sufficient degrees of freedom for the modelling of both speech and speaker variations. Too high a value of M , results in the requirement for a large amount of training data. If, relative to the model order, an inadequate amount of training data is available for each speaker, then speaker identification performance will suffer. A high model order also results in an excessive amount of computation during the training and identification phases. Empirical measurements have indicated that a model order M ranging between 10 and 20 allows adequate modelling for the

Figure 4.1: GMM Vector Generation Model For Speaker λ_j

purposes of speaker identification [1]. The performance increase for a model order above 30 in speaker identification applications is negligible.

The K -dimensional random vector $\vec{X}$ is modelled as being generated according to the distribution of Equation 4.1. The probability density of vector $\vec{X}$ for a given speaker is

$$p_{\vec{X}}(\vec{x}|\lambda_j) = \sum_{i=1}^M P_i p_{\vec{X}|I}(\vec{x}|i, \lambda_j), \quad (4.2)$$

which is the weighted summation of multivariate Gaussian probability density functions, where each distribution is weighted by the probability of occurrence of the corresponding acoustic class. Equation 4.2 serves as the GMM for speaker identification applications.

The application of the GMM to the modelling of feature vectors generated by a set of speakers in a database implies that each speaker in the database must have a model specifying the acoustic class probabilities as well as the mean and covariance vectors. The complete model for speaker j may therefore be notationally represented by

$$\lambda_j = \{P_i, \vec{\mu}_{ij}, \Sigma_{ij}\} \text{ for } i = 1 \dots M \text{ and } j = 1 \dots J, \quad (4.3)$$

where J is the total number of speakers in the identification system database. Although the complete specification of a speaker model requires the use of a set of M full $K \times K$ covariance matrices $\{\Sigma_{ij}\}_{i=1}^M$, estimation of these matrices from a set of feature vectors requires a great deal of processing. Fortunately, the selected parameter elements in the feature vector often show a high degree of statistical independence, so the covariance matrix may be replaced with a simpler diagonal matrix. This significantly reduces the number of parameters which must be estimated during the algorithm training phase and also simplifies the calculations during the identification phase since Equation 4.2 simplifies to

$$p_{\vec{X}}(\vec{x}|\lambda_j) = \sum_{i=1}^M \frac{P_i}{(2\pi)^{\frac{K}{2}} \sqrt{\prod_{k=1}^K \sigma_{ijk}^2}} e^{-\frac{1}{2} \sum_{k=1}^K \frac{(x_k - \mu_{ijk})^2}{\sigma_{ijk}^2}}, \quad (4.4)$$

where x_k is the k th element of the feature vector $\vec{x}$.

μ_{ijk} is the k th element of the mean feature vector $\vec{\mu}_{ij}$, and

σ_{ijk}^2 is the k th diagonal element of the $K \times K$ covariance matrix Σ_{ij} .

4.3 Model Identification Phase

The identification phase of the GMM involves a Maximum Likelihood detection problem similar to that discussed in Section 2.6. It is assumed that a set of T_R feature vectors $\{\vec{x}_t\}_{t=1}^{T_R}$ is available from which a decision must be made as to which speaker model best fits the parameter vectors. In a closed set speaker identification problem there are J speaker models available.

If Equation 2.38 is extended to a set of feature vectors, and statistical independence between vectors is assumed, the Maximum Likelihood detection problem may be rewritten so that for a given set of feature vectors, $\{\vec{x}_t\}_{t=1}^{T_R}$, speaker model λ_j is chosen as the most likely if

$$\mathcal{L}(\lambda_j) = \prod_{t=1}^{T_R} p_{\vec{x}}(\vec{x}_t|\lambda_j) > \prod_{t=1}^{T_R} p_{\vec{x}}(\vec{x}_t|\lambda_i) \text{ for } j \neq i, \text{ and } i = 1 \dots J, \quad (4.5)$$

where $\vec{x}_t$ is the k th feature vector in the set $\{\vec{x}_t\}_{t=1}^{T_R}$.

Since evaluation of $\mathcal{L}(\lambda_i)$ for $i = 1 \dots S$ involves a fixed set of feature vectors $\{\vec{x}_t\}$ and a variable set of speaker model, $\mathcal{L}(\lambda_i)$ is a function of the speaker models and is known as the *likelihood function* for the speaker classification. Often, for component densities of the type shown in Equation 4.4, the logarithm of the likelihood function is easier to evaluate since numerical round off problems are avoided. In this case, speaker j is chosen if

$$\log \mathcal{L}(\lambda_j) = \sum_{t=1}^{T_R} \log p_{\vec{x}}(\vec{x}_t|\lambda_j) > \sum_{t=1}^{T_R} \log p_{\vec{x}}(\vec{x}_t|\lambda_i) \text{ for } j \neq i, \text{ and } i = 1 \dots J. \quad (4.6)$$

Since the order of vectors in Equations 4.5 and 4.6 are unimportant, both the likelihood and log likelihood functions are suitable for text independent speaker identification applications.

4.4 Model Training Phase

4.4.1 Maximum Likelihood Estimation

Training of the GMM for each speaker involves estimating the parameters listed in Equation 4.3 for each of the S speakers contained in the database. In general, an estimation problem is much more complicated than a simple detection problem [44].

For the model training, it is assumed that a set of T_T feature vectors $\{\vec{x}_t\}_{t=1}^{T_T}$, extracted from speech known to have originated from the appropriate speaker, is available for each speaker model in the database. The set of parameters which maximize the likelihood function over the feature vector training set is used as the speaker model. These parameters must satisfy the Likelihood Equation which states [44]:

$$\frac{d\mathcal{L}(\lambda)}{d\lambda} = 0, \quad (4.7)$$

where the derivative notation is used to represent the gradient of $\mathcal{L}(\lambda)$ with respect to the vectors contained in the model λ given by Equation 4.3. It can be shown that the solution to Equation 4.7 is both asymptotically consistent as well as efficient. These important statistical properties ensure that given a sufficiently large training set, the solution of Equation 4.7 will converge to the true parameter set for the speaker [44].

Unfortunately, attempting to solve Equation 4.7 directly does not yield a closed form solution. Instead, the maximum likelihood parameters must be solved using an iterative algorithm in which an initial estimate is made of the model parameters, and then iteratively refined until a convergence threshold is met. The objective of the iteration is, given a speaker model λ , find a new model $\bar{\lambda}$ such that

$$\mathcal{L}(\bar{\lambda}) \geq \mathcal{L}(\lambda) \quad (4.8)$$

and keep iterating until a convergence is met. This type of estimation procedure is a special case of a more general statistical estimation algorithm known as Expectation Maximization.

4.4.2 Auxiliary Function

In order to develop the equations for the iteration described in Equation 4.8, the likelihood function of Equation 4.5 must first be expressed in terms of a joint pdf between the observed feature vector training set, and the acoustic classes which produced the training set. It is assumed that there are M acoustic classes contained in the GMM, and that the set of training vectors may be modelled by a set of T_T identical, independent random vectors $\{\tilde{\mathbf{x}}_t\}_{t=1}^{T_T}$. It is also assumed that the acoustic classes transitioned during the observation of the training set may be described in terms of the set of discrete random variables $\{I_t\}_{t=1}^{T_T}$. For each value of t , I_t may assume one of the M acoustic classes used in the model. The likelihood function of Equation 4.5 for speaker model λ may be specified in terms of the joint pdf of the training vectors and acoustic classes as follows:

$$\begin{aligned}
 \mathcal{L}(\lambda) &= \prod_{t=1}^{T_T} p_{\tilde{\mathbf{x}}_t}(\tilde{\mathbf{x}}_t|\lambda) \\
 &= p_{\tilde{\mathbf{x}}_1, \tilde{\mathbf{x}}_2, \dots, \tilde{\mathbf{x}}_{T_T}}(\tilde{\mathbf{x}}_1, \tilde{\mathbf{x}}_2, \dots, \tilde{\mathbf{x}}_{T_T}|\lambda) \\
 &= \sum_{i_1, i_2, \dots, i_{T_T}} p_{\tilde{\mathbf{x}}_1, \tilde{\mathbf{x}}_2, \dots, \tilde{\mathbf{x}}_{T_T}, I_1, I_2, \dots, I_{T_T}}(\tilde{\mathbf{x}}_1, \tilde{\mathbf{x}}_2, \dots, \tilde{\mathbf{x}}_{T_T}, i_1, i_2, \dots, i_{T_T}|\lambda) \\
 &= \sum_{i_1, i_2, \dots, i_{T_T}} \prod_{t=1}^{T_T} P_{i_t} p_{\tilde{\mathbf{x}}|I}(\tilde{\mathbf{x}}|i, \lambda),
 \end{aligned} \tag{4.9}$$

where $p_{\tilde{\mathbf{x}}_1, \tilde{\mathbf{x}}_2, \dots, \tilde{\mathbf{x}}_{T_T}}(\tilde{\mathbf{x}}_1, \tilde{\mathbf{x}}_2, \dots, \tilde{\mathbf{x}}_{T_T}|\lambda)$ is the joint pdf of the T_T feature vectors,

$p_{\tilde{\mathbf{x}}_1, \tilde{\mathbf{x}}_2, \dots, \tilde{\mathbf{x}}_{T_T}, I_1, I_2, \dots, I_{T_T}}(\tilde{\mathbf{x}}_1, \tilde{\mathbf{x}}_2, \dots, \tilde{\mathbf{x}}_{T_T}, i_1, i_2, \dots, i_{T_T}|\lambda)$ is the joint pdf of the feature vectors and acoustic classes, and

P_{i_t} is the discrete probability that the acoustic state for the t th observed feature vector is equal to i_t .

The iteration of Equation 4.8 for a speaker model λ , given the form of Equation 4.9, may be accomplished via the definition of an auxiliary function Q which is given by

$$Q(\lambda, \bar{\lambda}) = \sum_{t=1}^{T_T} \sum_{i=1}^M \ln \left[\frac{\bar{P}_i}{(2\pi)^{\frac{K}{2}} |\bar{\Sigma}_i|^{\frac{1}{2}}} e^{-\frac{1}{2}(\tilde{\mathbf{x}}_t - \bar{\boldsymbol{\mu}}_i)^H |\bar{\Sigma}_i|^{-1} (\tilde{\mathbf{x}}_t - \bar{\boldsymbol{\mu}}_i)} \right] \gamma_t(i). \tag{4.10}$$

where

$$\gamma_t(i) = P(i_t = i | \vec{x}_t, \lambda) \prod_{k=1}^{T_T} p_{\vec{x}}(\vec{x}_k | \lambda) \quad (4.11)$$

and $\lambda = \{P_i, \vec{\mu}_i, \Sigma_i\}$ for $i = 1 \dots M$ is the speaker model before the iteration of Equation 4.8.

$\bar{\lambda} = \{\bar{P}_i, \bar{\vec{\mu}}_i, \bar{\Sigma}_i\}$ for $i = 1 \dots M$ is the speaker model after the iteration of Equation 4.8.

$P(i_t = i | \vec{x}_t, \lambda)$ is the probability that the acoustic state corresponding to feature vector $\vec{x}_t$ is equal to i , given the occurrence of that feature vector.

$\ln$ denotes the natural logarithm operation.

For the density function given in Equation 4.1 $P(i_t = i | \vec{x}_t, \lambda)$, which is called the *a posteriori* probability for state i_t , may be written as

$$P(i_t = i | \vec{x}_t, \lambda) = \frac{\frac{P_i}{(2\pi)^{\frac{K}{2}} |\Sigma_i|^{\frac{1}{2}}} e^{-\frac{1}{2}(\vec{x}_t - \vec{\mu}_i)^H |\Sigma_i|^{-1} (\vec{x}_t - \vec{\mu}_i)}}{\sum_{k=1}^M \frac{P_k}{(2\pi)^{\frac{K}{2}} |\Sigma_k|^{\frac{1}{2}}} e^{-\frac{1}{2}(\vec{x}_t - \vec{\mu}_k)^H |\Sigma_k|^{-1} (\vec{x}_t - \vec{\mu}_k)}}. \quad (4.12)$$

It should be noted that $P(i_t = i | \vec{x}_t, \lambda)$ is a function of the speaker model parameters which exist before the iteration of Equation 4.8.

4.4.3 Model Parameter Iteration

The auxiliary function of Equation 4.10 has the property that finding a set of model parameters $\bar{\lambda}$ which maximizes $Q(\lambda, \bar{\lambda})$ results in an increase in the likelihood function, as required by Equation 4.8. It may therefore be used to determine formulas for the iteration of the model parameters given in Equation 4.3.

To find an expression for the new set of the mixture weights $\bar{P}_i$, given the old model λ , $Q(\lambda, \bar{\lambda})$ must be maximized subject to the constraint that $\sum_{k=1}^M \bar{P}_i = 1$. This constrained maximization problem may be solved using the Lagrange Multiplier

technique [40]. The solution is

$$\bar{P}_i = \frac{\sum_{t=1}^{T_T} \gamma_t(i)}{\sum_{t=1}^{T_T} \sum_{k=1}^M \gamma_t(k)}, \quad (4.13)$$

which may be simplified to

$$\bar{P}_i = \frac{1}{T_T} \sum_{t=1}^{T_T} P(i_t = i | \vec{x}_t, \lambda), \quad (4.14)$$

where $P(i_t = i | \vec{x}_t, \lambda)$ is as defined in Equation 4.12.

The iteration equation for the new set of mean vectors $\tilde{\mu}_i$ for $i = 1 \dots M$ may be determined by taking the gradient of Equation 4.10 with respect to $\tilde{\mu}_i$ and setting the result equal to 0. The resulting solution for the new set of mean vectors is given by

$$\tilde{\mu}_i = \frac{\sum_{t=1}^{T_T} \gamma_t(i) \vec{x}_t}{\sum_{t=1}^{T_T} \gamma_t(k)} \text{ for } i = 1 \dots M, \quad (4.15)$$

which reduces to

$$\tilde{\mu}_i = \frac{\sum_{t=1}^{T_T} P(i_t = i | \vec{x}_t, \lambda) \vec{x}_t}{\sum_{t=1}^{T_T} P(i_t = i | \vec{x}_t, \lambda)} \text{ for } i = 1 \dots M. \quad (4.16)$$

Finally, the new covariance matrices $\tilde{\Sigma}_i$ for $i = 1 \dots M$ may be determined by taking the gradient of Equation 4.10 with respect to Σ_i and setting the result equal to 0. The solution for the new covariance matrices is given by

$$\tilde{\Sigma}_i = \frac{\sum_{t=1}^{T_T} \gamma_t(i) (\vec{x}_t - \tilde{\mu}_i) (\vec{x}_t - \tilde{\mu}_i)^H}{\sum_{t=1}^{T_T} \gamma_t(k)} \text{ for } i = 1 \dots M \quad (4.17)$$

where $\tilde{\mu}_i$ is given by Equation 4.16. Equation 4.17 reduces to

$$\bar{\Sigma}_i = \frac{\sum_{t=1}^{T_T} P(i_t = i | \tilde{x}_t, \lambda) \tilde{x}_t \tilde{x}_t^H}{\sum_{t=1}^{T_T} P(i_t = i | \tilde{x}_t, \lambda)} - \tilde{\mu}_i \tilde{\mu}_i^H \text{ for } i = 1 \dots M. \quad (4.18)$$

If the covariance matrix Σ_i is assumed to be diagonal, Equation 4.18 may be further simplified such that

$$\bar{\sigma}_{ik}^2 = \frac{\sum_{t=1}^{T_T} P(i_t = i | \tilde{x}_t, \lambda) x_{tk}^2}{\sum_{t=1}^{T_T} P(i_t = i | \tilde{x}_t, \lambda)} - \bar{\mu}_i^2 \text{ for } i = 1 \dots M \text{ and } k = 1 \dots K, \quad (4.19)$$

where $\bar{\sigma}_{ik}^2$ is the k th diagonal element of the covariance matrix Σ_i ,

x_{tk} is the k th element of the feature vector $\tilde{x}_t$, and

$\bar{\mu}_{ik}$ is the k th element of the mean feature vector $\tilde{\mu}_i$.

Equations 4.14, 4.16, and 4.18 (or 4.19), along with Equation 4.12 define the complete set of training equations for a speaker model, given a set of T_T feature vectors $\{\tilde{x}_t\}_{t=1}^{T_T}$. As illustrated in Figure 4.2, the training starts out with an initial estimate of the model parameters made from the training data. This estimate is then refined by iterating the four equations. The same set of training vectors is used for each iteration of the equations. If the likelihood function of Equation 4.5 is defined over the set of training vectors, then each training iteration increases the value of the likelihood function until, as shown in Figure 4.3 it effectively converges to a certain value. Once the likelihood function converges to the point that each iteration results in a small percentage change in the likelihood function, the iteration of Figure 4.2 may be terminated, and the GMM for the speaker may be considered to be trained.

4.4.4 Initial Estimate

As may be seen from Figure 4.2, the GMM training algorithm requires that an initial estimate of the model parameters be made from the set of training vectors. Several

Figure 4.2: GMM Training Procedure

Figure 4.3: Typical Likelihood Function Convergence

initialization strategies are possible. One technique involves the use of an HMM phoneme recognizer to segment the phoneme groups corresponding to various acoustic classes contained in the model [1]. Once these classes have been tagged in the training data, the P_i values can be determined as well as the class mean vectors $\vec{\mu}_i$. Global variances may then be used for the initial estimates of the diagonal elements of Σ_i .

A much simpler technique, which will be employed in Chapter 5 involves the grouping of vectors around randomly chosen feature vectors contained in the training set. Assuming that the GMM contains M acoustic classes, this initialization technique involves clustering feature vectors around M randomly chosen feature vectors. The training set is first analyzed to determine the global mean and variance for the entire set of training feature vectors. The global mean vector is defined as

$$\vec{m} = \frac{1}{T_T} \sum_{t=1}^{T_T} \vec{x}_t. \quad (4.20)$$

Assuming that the feature vectors are of dimension K , a $K \times K$ dimension global variance matrix $\mathbf{G}$ may be defined as a diagonal matrix with elements equal to

$$\sigma_k^2 = \frac{\sum_{t=1}^{T_T} (x_{tk} * x_{tk}) - T_T m_k^2}{T_T - 1} \text{ for } k = 1 \dots K. \quad (4.21)$$

where σ_k^2 is the k th diagonal element of the global variance matrix $\mathbf{G}$,

x_{tk} is the k th element of the feature vector $\vec{x}_t$, and

m_k is the k th element of the mean feature vector $\vec{m}$ defined by

Equation 4.20.

Once $\vec{m}$ and $\mathbf{G}$ are determined, a set of M feature vectors $\{\vec{r}_i\}_{i=1}^M$ are selected at random from the training set to be representative of each acoustic class. M clusters of parameter vectors are then created by grouping the feature vectors nearest to the representative vector for each class. A given feature vector $\vec{x}_t$ is put in acoustic class j if

$$d(\vec{x}_t, \vec{r}_j) < d(\vec{x}_t, \vec{r}_i) \text{ for } i = 1 \dots M, \quad (4.22)$$

where $d(\vec{x}_t, \vec{r}_j)$ is the Mahalanobis distance defined by Equation 2.39 and $\mathbf{W}$ in Equation 2.39 is replaced by $\mathbf{G}$. The mean vectors $\vec{\mu}_i$ and variance vectors Σ_i are then

determined for each acoustic class using the same formulas as in Equations 4.20 and 4.21 respectively, where T_T is replaced with the number of vectors assigned to each cluster. Finally, the acoustic class probabilities $\{P_i\}_{i=1}^M$ are estimated using the percentage of vectors assigned to each acoustic class, out of the total number of feature vectors available for training.

It can be shown that the GMM training algorithm illustrated in Figure 4.2 is guaranteed to result in the convergence of the likelihood function to a local maximum, regardless of the initial estimate. The likelihood function for a typical GMM may however have several local maxima and different initial estimates can lead to the convergence of the likelihood function to different local maxima, which of course implies that the resulting estimates for the speaker models are different. Although it is unclear [1] exactly how speaker identification performance is affected by the existence of several local maxima it does appear that speaker identification performance is roughly the same regardless of the initialization scheme employed.

4.4.5 Practical Considerations

One of the most important considerations for GMM based speaker recognition is the availability of a sufficient amount of training data. Generally it is assumed that the training data has a relatively high SNR, and contains enough variation to allow a text independent speaker model to be developed. Unfortunately, the amount of training data required for a given problem can only be determined empirically. Tests have revealed that in general a minimum of sixty seconds of active speech is required in order to train a GMM. This figure however does increase for higher order GMMs.

One symptom of an inadequate amount of training data is the occurrence of variances in Equation 4.19 which tend to zero during the training. This occurs as a result of a singularity in the likelihood function which arises when there is not enough training data for the model order. The same effect also can occur if it becomes necessary to use noisy or corrupted data during the training. A simple technique for avoiding the occurrence of spurious singularities during the model training is the application of a variance limiting scheme in Equation 4.19. If σ_{ik}^2 is evaluated to be less than a value of σ_{min}^2 , it is reset to be equal to σ_{min}^2 . This constraint on the

iterative algorithm has been shown to result in more robust parameter estimates than the unconstrained version [1]. Generally the exact value of σ_{min}^2 must be empirically determined for a particular data set and model order.

The effects of *intersession variability* must also be considered during the training phase of GMM development. This term is used to describe changes in the feature vector statistics collected for a particular speaker which occur if the training set is collected over different sessions, with time gaps between each session. These changes may be the result of subtle physiological changes in the vocal tract or may be caused by variations in the recording equipment used in the different sessions.

To minimize the effects of intersession variability, a mean feature vector is generally defined over each of the training sessions using Equation 4.20. This mean vector is subtracted from each feature vector in a given training session and the difference is used to train the GMM for the speaker. In other words a new set of feature vectors $\tilde{z}$ is used for GMM training where

$$\tilde{z}_t = \tilde{x}_t - \tilde{m}. \quad (4.23)$$

$\tilde{x}_t$ is a feature vector collected during a given training session and $\tilde{m}$ is the mean feature vector for the training session.

Chapter 5

Experimental Results

5.1 Introduction

In this chapter, a comparison is made between the performance of five different types of feature vectors, applied to a closed set, text independent speaker identification problem. In the first two cases, MEL spaced cepstral coefficients are used to make up the feature vectors. In the three other cases, the feature vectors are derived from a wavelet transform. All software discussed in this section is written in C and runs under UNIX on a Sparc IPX workstation.

The closed set identification problem implemented uses speech data obtained from the SPIDRE speech corpus [5]. This corpus is composed of samples of conversational speech between two speakers recorded over the telephone. Each speaker is recorded on a different channel, which eliminates the requirement to segment speakers during the training and testing phases of the experiment. The sampling frequency for all SPIDRE speech files is 8.0 kHz.

The identification system implemented is illustrated in Figure 5.1. Each sample set of speech is segmented into overlapping frames as shown in Figure 5.2. Frame widths vary depending on the type of feature vector extracted, but a frame overlapping factor of fifty percent is always used. This overlapping factor was chosen to be identical to the value used in reference [1]. A speech activity detection algorithm determines if a given frame contains active speech. Inactive frames are ignored for both training

Figure 5.1: Speaker Identification System

Figure 5.2: Speech Frame Processing

and identification phases. The vector of speech samples extracted from each active frame is processed to create a feature vector which characterizes the frame. These feature vectors are collected and processed in either a training or identification mode as appropriate.

Training is accomplished using the procedure outlined in Section 4.4 for all 45 SPIDRE target speakers. Identical training files are used for both the cepstral and wavelet based feature vector sets, with sixty seconds of active speech being used to form the training vectors each case.

The identification phase of the experiment is performed using samples of speech different from the training set. Identical data files are used for the testing of all feature vector sets. Active speech frames are used to form the likelihood function of Equation 4.5 for each speaker model in the database. For a given sample of speech, 45 different likelihood functions are formed for every ten second interval of active speech. The speaker model yielding the highest value of Equation 4.5, averaged over a minute and a half of speech, is determined to be the most likely speaker.

5.2 Speech Activity Detection

As mentioned in Section 5.1, only active frames of speech are used for both the training and identification phases of the speaker identification system. The preprocessor Speech Activity Detection (SAD) algorithm used to extract the active speech frames is almost identical to the one used in reference [1], and is presented in the flow chart of Figure 5.3. The interaction of the various parameters is illustrated in Figure 5.4 and actual values used for the SAD parameters are given in Table 5.1. These values are based on the SAD parameters used in reference [1].

For each frame of speech data, the frame energy is evaluated as

$$FE(i) = \sum_{j=S_i}^{E_i} s^2(jT_s), \quad (5.1)$$

where $FE(i)$ is frame energy for the i th frame,
 S_i is the starting point of the i th frame, and
 E_i is the end point of the i th frame.

Figure 5.3: Speech Activity Detection Algorithm

Figure 5.4: Speech Activity Detection Algorithm Illustration

The frame energy is applied to a single pole filter and is used, along with the energy of the previous frame in order to provide a smoothed energy estimate as follows:

$$SE(i) = FE(i) + SAD_C_1 * FE(i - 1), \quad (5.2)$$

where $SE(i)$ is the smoothed energy estimate for the i th frame, and SAD_C_1 is the pole filter coefficient.

The smoothed energy parameter is used to adaptively track the noise floor of the signal. At each frame, the noise floor is taken to be the lesser of either the smoothed energy or the previous noise floor estimate raised by a small amount. This procedure ensures that the SAD algorithm automatically estimates the noise floor value and can accurately follow it if it rises over time. A speech to silence transition may be detected if the noise floor is evaluated to be greater than twice the value of the frame energy. In this event, the noise floor is set to be equal to half of the frame energy. This allows the adaptive noise floor estimate to be stepped down following an interval of speech.

The frame SNR is calculated as

$$SNR(i) = 10.0 \log_{10} \left(\frac{FE(i)}{NF(i)} \right) \quad (5.3)$$

where $SNR(i)$ is signal to noise ratio estimate for the i th frame,
 $FE(i)$ is the frame energy for the i th frame, and
 $NF(i)$ is the noise floor estimate for the i th frame.

SAD Parameter Name	Parameter Value
SAD_C_1	-0.8
SNR_THRESH	20.0 dB
SC_THRESH	10
NUM_BACK	10
SC_MAX	20
MIN_FE	0.00001

Table 5.1: Speech Activity Detection Algorithm Parameters

For every frame the SNR is compared to a threshold SNR_THRESH and a counter SC is incremented if the SNR exceeds the threshold. During silence to speech transitions, the SC counter may be observed to increase as in Figure 5.4. During speech to silence transitions the counter decreases. If the SC counter exceeds a value of SC_THRESH , then a silence to speech transition is judged to have occurred and the current frame, as well as the previous NUM_BACK frames are labelled as active speech frames. The SC counter is not allowed to exceed a value of SC_MAX in order to ensure that the algorithm is capable of detecting short gaps in the speech, since the SC counter must be decremented to a value less than SC_THRESH in order for the silence interval to be detected. As can be seen from Figure 5.4, the algorithm is still rather slow to detect speech to silence transitions and continues to tag active frames until SC goes below SC_THRESH , with a delay of approximately 250 msec.

One slight modification made in this thesis to the SAD algorithm described in reference [1] is the addition of a lower limit on the minimum value of $FE(i)$. During silence intervals for high SNR signals, it was observed that the noise floor estimate can frequently be evaluated to be equal to zero. The algorithm then fails to track any rises in noise floor since $1.01 \times 0 = 0$ in the flow chart of Figure 5.3. Furthermore, all subsequent SNR values are calculated as infinite. The placement of a lower limit MIN_FE on the frame energy estimate corrects this problem. The value of MIN_FE shown in Table 5.1 was chosen such that it was certain to be less than one LSB value in the ADC.

5.3 Cepstral Feature Vector Sets

Two cepstral based feature vector sets are used in this thesis. Both are derived from MEL spaced cepstral coefficients extracted from active speech frames as shown in Figure 5.5 [1]. A frame length of 20.0 msec is used, corresponding to 160 samples per frame. Each speech frame is then zero padded to a length of 256 points and an FFT is performed on the time domain data. The energy spectrum, which is defined as the magnitude squared of the FFT is then applied to a pre-emphasis transfer function designed to compensate for the high frequency roll-off of the glottal excitation, as

discussed in Section 2.5. The transfer function chosen is as specified in Equation 2.18 with a value of $\alpha = 0.7$. This value of α was empirically determined to provide a slightly better speaker recognition performance than a higher value of 0.9, which is more commonly used in speech processing. This effect is probably due to the data collection technique used in the SPIDRE speech corpus. The pre-emphasized energy spectrum is then applied to a bank of twenty one triangular filters having the frequency domain responses illustrated in Figure 5.6. The filters are chosen to have bandwidths of 100 Hz with equal spacing in the frequency range 0 to 1.0 kHz. In the range 1.0 to 4.0 kHz, the linear bandwidths and spacings are again 100 Hz, but these are subsequently converted to MEL frequencies according to Equation 2.24. The log energy in each band is defined as

$$mfb(i) = 10.0 \log_{10} \left[\frac{\sum_{k=L_i}^{U_i} |S(k)H(e^{j\frac{2\pi k}{N}})\Lambda_i(e^{j\frac{2\pi k}{N}})|^2}{\sum_{k=L_i}^{U_i} \Lambda_i(e^{j\frac{2\pi k}{N}})} \right] \text{ for } i = 0 \dots N_F - 1. \quad (5.4)$$

where $mfb(i)$ is the log energy contained in the i th band after pre-emphasis.

$S(k)$ is k th frequency bin of the FFT.

$\Lambda_i(k)$ is the value of the i th triangular filter in the k th frequency bin.

$H(z)$ is the pre-emphasis function given by Equation 2.18.

L_i is first non zero frequency bin of the triangular filter.

U_i is last non zero frequency bin of the i th triangular filter.

N is the number of points in the FFT (256).

N_F is the number of filters in the triangular filter bank (21).

The denominator in Equation 5.4 is for the normalization of the various filter widths since, as shown in Figure 5.6, the filter bandwidths increase with frequency.

The values of $\{mfb(i)\}_{i=0}^{N_F-1}$ are called the mel-spaced filter bank coefficients and are used to form the cepstral coefficients for the feature vector. These cepstral coefficients are derived from the Discrete Cosine Transfer (DCT) of the mfb vector. This operation is defined as

$$mcc(i) = g(i) \sum_{k=0}^{N_F-1} mfb(k) \cos \left[\frac{(2k+1)\pi i}{2N_F} \right] \text{ for } i = 0 \dots N_F - 1. \quad (5.5)$$

Figure 5.5: Cepstral Feature Vector Extraction

Figure 5.6: Triangular Filter Bank

where $mcc(i)$ is i th cepstral coefficient.

$mf b(k)$ is k th mel spaced filter bank energy parameter, and

$g(i) = \frac{1}{\sqrt{2}}$ for $i = 0$ and 1.0 for $i = 1 \dots N_F - 1$.

The DCT is used to approximate the Inverse Fourier Transform operation of Figure 2.22. This approximation is valid since the $mf b(i)$ values are all real. The value of $mcc(0)$ represents the average log energy in the active speech frame and is discarded from the feature vector as a form of amplitude normalization [1]. The twenty remaining elements of the DCT are used to form the $\vec{C}_1$ cepstral feature vector, which is used in the experiments of Section 5.5. This feature vector is identical to the type used in reference [1]. A second cepstral based feature vector, which is designated as $\vec{C}_2$, is formed in an identical manner except that only the first seven filters in the triangular filter bank of Figure 5.6 are used. The performance of this six dimensional feature vector is compared with the performance of similarly dimensioned wavelet derived feature vector in Section 5.5, where the importance of high resolution spectral analysis of the 0 - 1.0 kHz frequency band becomes evident.

5.4 Wavelet Based Feature Vector Sets

The wavelet derived feature vector sets used in this thesis are chosen to be of dimensions 23, 12, and 6. It was felt that this selection of wavelet dimensions would provide a good indication as to whether or not wavelet derived feature vectors would offer any advantages over the twenty dimensional cepstral based feature vector used in reference [1].

The wavelet based feature vector sets are extracted from active speech frames as shown in Figure 5.7. A frame length of 16.0 msec is used, resulting in a total of 128 points per frame. This is slightly less than in the case of the cepstral feature vector extraction, in which 160 points per frame are used, but as discussed in Section 3.4 an order of two relationship must be maintained so that the decimation operations of the FWT are performed correctly. Keeping the frame length an exact power of two allows for the maximum flexibility in the choice of FWT orders. A value of 128 is the nearest power of two value to 160. Active frames are processed by a sixth order FWT

initialized using twelve tap Daubechies lowpass filter defined by Equation 3.23. The network filter response is as shown in Figure 3.12. All wavelet based feature vectors used in Section 5.5 are calculated from the FWT.

In the case of the first wavelet based feature vector ($\vec{W}_1$), the output of the FWT is used to form a time aligned magnitude squared scaleogram matrix using the technique of Section 3.4. Each row of the scaleogram represents the magnitude squared output of one of the filters in the FWT network. The elements of each row of the scaleogram are multiplied with a pre-emphasis coefficient calculated from Equation 2.18. A log magnitude matrix is then calculated as follows:

$$\tilde{S}_{LOG-ik} = 10.0 \log_{10} \left[\left| \tilde{S}_{ik} H(e^{j\frac{\pi i}{O+1}}) \right|^2 \right] \quad (5.6)$$

for $0 \leq i \leq O$ and $0 \leq k \leq \frac{N}{2} - 1$,

where $\tilde{S}_{LOG-ik}$ is the element in row i and column k of the log magnitude scaleogram matrix,

$\tilde{S}_{ik}$ is the element in row i and column k of the time aligned scaleogram matrix,

$H(z)$ is the pre-emphasis function given by Equation 2.18.

O is the order of the FWT (6), and

N is the number of points per frame (128).

Following a technique similar to the one applied to spectrograms in reference [81], a two dimensional DCT is applied to the log magnitude matrix as follows:

$$wcc2_{ij} = g(i)g(j) \sum_{k=0}^O \sum_{l=0}^{\frac{N}{2}-1} \tilde{S}_{LOG-kl} \cos \left[\frac{(2k+1)\pi i}{2O+2} \right] \cos \left[\frac{(2l+1)\pi j}{N-2} \right] \quad (5.7)$$

for $0 \leq i \leq O$ and $0 \leq j \leq \frac{N}{2} - 1$,

where $wcc2_{ij}$ is the element in row i and column j of the two dimensional DCT.

$\tilde{S}_{LOG-ik}$ is the element in row i and column k of the log magnitude scaleogram matrix, and

$g(i) = \frac{1}{\sqrt{2}}$ for $i = 0$ and 1.0 for $i \neq 0$.

Figure 5.7: Wavelet Feature Vector Extraction

The $wcc2_{ij}$ matrix is a two dimensional extension of the $mcc(i)$ vector developed in Section 5.3. The DCT elements are arranged in a single dimensional vector as

$$w_1(iJ' + j) = \frac{wcc2_{ij}}{wcc2_{00}} \text{ for } 0 \leq i \leq I' \text{ and } 0 \leq j \leq J' \text{ with } (i, j) \neq (0, 0), \quad (5.8)$$

where $w_1(i)$ is the i th element of the $\vec{W}_1$ feature vector.

I' is the number of rows chosen from the two dimensional DCT, and

J' is the number of columns chosen from the two dimensional DCT.

The value of $wcc2_{00}$ is used as a normalization coefficient. These same feature vectors, calculated from a spectrogram, were used for speaker identification purposes in reference [81] where it was discovered that for a frame size of 256 points, better than 90% speaker identification accuracy could be obtained using only the low frequency DCT components, with $I' = 10$ and $J' = 15$ in Equation 5.8. Even using values of $I' = 4$ and $J' = 6$ obtained an identification accuracy of approximately 60% on a speaker database of 50 speakers. These values of I' and J' are chosen to create the twenty three dimensional $\vec{W}_1$ feature vector set in Section 5.5 since they result in a feature vector of approximately the same dimension as that of the $\vec{C}_1$ feature vector developed in Section 5.3.

The second wavelet based feature vector set used in this thesis is designated as $\vec{W}_2$ and is based on a wavelet derived version of the cepstral feature vectors developed in Section 5.3. The development begins with a wavelet filter bank energy vector, which is analogous to the filter bank energy vector (mfb) used in Section 5.3. The log energy contained in each band of the wavelet transform may be calculated from the FWT as follows:

$$\begin{aligned} wfb(O + 1 - k) &= 10.0 \log_{10} \left[\left| H(e^{j\frac{\pi k}{O+1}}) \right|^2 \sum_{i=0}^{\frac{N}{2^k}} \left| \tilde{S}(i, k) \right|^2 \right] \text{ for } 1 \leq k \leq O \text{ and} \\ wfb(0) &= 10.0 \log_{10} \left[\left| H(1) \right|^2 \sum_{i=0}^{\frac{N}{2^O}} \left| \tilde{S}(i, O + 1) \right|^2 \right]. \end{aligned} \quad (5.9)$$

where $wfb(k)$ is the k th element of the wavelet band energy vector, which represents log energy in the j th wavelet band after pre-emphasis.

- $\tilde{S}(i, j)$ is the FWT given by Equation 3.26,
 $H(z)$ is the pre-emphasis function given by Equation 2.18,
 N is the number of points per frame, and
 O is the order of the FWT.

The elements of $\{wfb(i)\}_{i=0}^O$ are used to form a feature vector in which the energies of lower frequency bands are at the lower values of i . The elements of the $\vec{W}_2$ feature vector set are formed from the DCT of the wfb vector using Equation 5.5 and replacing $mfb(k)$ with $wfb(k)$. As in the case of the $\vec{C}_1$ feature vector set, the zero frequency element of the DCT is dropped as a form of amplitude normalization, following the technique used in reference [1]. The final $\vec{W}_2$ vector therefore has six elements.

The final wavelet based feature vector, designated $\vec{W}_3$, is derived from the $\vec{W}_2$ feature vector, and $\Delta\vec{W}_2$ vector which is defined by the differences between the $\vec{W}_2$ feature vector elements calculated for a given frame and those calculated for the previously active frame. The elements of the $\vec{W}_3$ feature vector set are therefore defined as

$$\begin{aligned}
 w_3(i) &= w_2(i) \text{ for } 0 \leq i \leq O - 1 \text{ and} \\
 w_3(i) &= w_2(i) - w'_2(i) \text{ for } O \leq i \leq 2O - 2,
 \end{aligned} \tag{5.10}$$

where $w_3(i)$ is the i th element of the $\vec{W}_3$ feature vector calculated for the most recent frame,

$w_2(i)$ is the i th element of the $\vec{W}_2$ feature vector calculated for the most recent frame,

$w'_2(i)$ is the i th element of the $\vec{W}_2$ feature vector calculated for the previously active frame, and

O is the order of the FWT.

For the wavelet filter bank of Figure 3.12, the result is a twelve dimensional feature vector.

5.5 Results

5.5.1 Training Set Selection

The SPIDRE speaker corpus is composed of audio data files containing conversational speech recorded over the telephone on a chat line set up by Texas Instruments Inc. Each data file contains two audio channels, with each speaker involved in the conversation being recorded on a different channel. Automatic segmentation of the speakers' voices is therefore not necessary. The two CD-ROM set contains approximately 280 conversation files, with each conversation lasting about 5 minutes.

The conversation files are sub-divided into target and non-target directories. The target directories contain conversations from the designated target speakers contained in the corpus. To qualify as a target speaker, a given speaker needed to have participated in four calls, exactly three of which were from different handsets [5]. There are 45 designated target speakers in SPIDRE. Non-target files are those conversations which do not contain any of the designated target speakers and are therefore not used in this thesis.

Of the 180 target speaker files contained in the speech corpus, a few were found to be either corrupted or had headers indicating that the target speakers were in fact not contained in the files. These files, which are listed in Table 5.2, were eliminated from both training and identification phases of the experiments performed in this thesis. It was also determined manually that the files listed in Table 5.3 contained errors in their headers which reversed the channel information for the two speakers present. These mistakes were corrected, using the true channel information for the target speaker and the files were used in either a training mode or a test mode as appropriate. It should be noted that these cases of reversed header information all involved a conversation between a man and a woman. It was therefore very easy to determine that mistakes had been made in the header information. It is suspected that other SPIDRE files containing male/male or female/female conversations may also be similarly mislabelled but since the distinction in these cases was not clear, the benefit of the doubt was given to the SPIDRE header and only the channel information for the files listed in Table 5.3 were corrected.

File Name	Reason For Elimination
.../target/sp1007/audio/sw2254.wav	File Is Corrupt
.../target/sp1171/audio/sw2945.wav	Target Speaker Not In Header
.../target/sp1181/audio/sw3169.wav	Target Speaker Not In Header

Table 5.2: Eliminated SPIDRE Target Files

File Name	Target Speaker	Corrected Channel
.../target/sp1028/audio/sw2303.wav	1028	1
.../target/sp1124/audio/sw2631.wav	1124	1
.../target/sp1022/audio/sw3018.wav	1022	2

Table 5.3: SPIDRE Target Files With Reversed Channel Information

The four conversations for each target speaker are contained in the SPIDRE sub-directories *.../target/spXXXX/audio*, where *XXXX* is the four digit identity of the respective target speaker. A single file from each target speaker sub directory was selected as the training file for that speaker. Training files were manually selected on the basis of speaker activity, clarity, and low crosstalk. An attempt was originally made to automatically select the training files for each speaker solely on the basis of frame activity. The SAD algorithm was run on all four target files for each speaker and the file with the most number of active frames was selected as the training file based on the assumption that these would be the files in which the target speakers were doing the greatest amount of talking. Unfortunately this attempt to select training files failed due to the high levels of cross talk in many of the SPIDRE target files. In many of the SPIDRE files, Speaker B may be clearly heard on Speaker A's channel and vice versa. If these files are used to train a model for either speaker, performance is degraded since the characteristics of both speakers are combined in the single speaker model. It was also observed that there was a significant variation in the crosstalk ratings assigned by the transcribers to the target files and that often files containing a significant amount of crosstalk were in fact labelled as having very little. These inconsistencies made it necessary to select the training files manually. The training files selected for each speaker are listed in Tables 5.4 and 5.5. All target speakers were speaking English and the dialects indicated in the tables are from the United States.

For each training file, the first sixty seconds of active speech from the speaker was used to train the model using the procedure of Figure 4.2. Training time on a SUN IPX is approximately ten minutes per speaker for both types of feature vectors. For the models created from the cepstral feature vector set, the 20 msec frame width leads to a value of $T_T = 3000$ in the training equations of Section 4.4. For the wavelet based feature vector, the slightly smaller frame width leads to a value of $T_T = 3750$. The values of σ_{min}^2 are also slightly different depending on whether the training is being performed on the cepstral feature vector set or the wavelet based feature vector set. A value of $\sigma_{min}^2 = 0.01$ was used for the cepstral feature vector following the value used reference [1]. The value of $\sigma_{min}^2 = 0.0001$ for the wavelet based feature vector set

Target Speaker ID	Sex	Dialect	Training File	Channel
1007	Female	Mid Western	sw2885.wav	1
1028	Male	New York City	sw2303.wav	1
1096	Female	South Midland	sw2640.wav	1
1112	Female	Southern	sw4697.wav	1
1124	Male	North Midland	sw2254.wav	1
1130	Female	Northern	sw2727.wav	1
1165	Male	Northern	sw4725.wav	1
1171	Male	South Midland	sw3102.wav	2
1188	Male	North Midland	sw3005.wav	2
1257	Male	New York City	sw2956.wav	1
1282	Female	Northern	sw3713.wav	1
1341	Female	Western	sw4387.wav	1
1353	Male	Mixed	sw3421.wav	1
1359	Male	Southern	sw3752.wav	1
1372	Female	Western	sw3966.wav	1
1415	Female	South Midland	sw3280.wav	1
1436	Male	South Midland	sw3578.wav	1
1440	Female	North Midland	sw3522.wav	1
1446	Male	South Midland	sw4816.wav	1
1497	Male	Western	sw3884.wav	1
1499	Male	Western	sw3975.wav	1
1533	Male	North Midland	sw4666.wav	2
1551	Male	North Midland	sw4166.wav	2

Table 5.4: Target Speaker Training Files (SPIDRE CD-ROM 1)

Target Speaker ID	Sex	Dialect	Training File	Channel
1013	Female	South Midland	sw2053.wav	1
1022	Female	South Midland	sw3018.wav	2
1043	Female	North Midland	sw2950.wav	1
1074	Female	New York City	sw3723.wav	1
1121	Male	New England	sw2958.wav	1
1148	Female	South Midland	sw3704.wav	2
1175	Female	Western	sw2941.wav	2
1181	Male	South Midland	sw2361.wav	1
1199	Male	Southern	sw4698.wav	1
1258	Male	South Midland	sw3692.wav	1
1407	Male	New York City	sw3561.wav	1
1425	Male	Western	sw4165.wav	1
1437	Female	Northern	sw3782.wav	2
1444	Female	Western	sw3641.wav	1
1449	Male	South Midland	sw3560.wav	1
1463	Male	Southern	sw3907.wav	1
1477	Male	North Midland	sw3755.wav	1
1481	Female	North Midland	sw4752.wav	1
1488	Male	Southern	sw3897.wav	1
1528	Male	Western	sw4400.wav	1
1531	Male	Northern	sw4022.wav	2
1575	Male	Northern	sw4361.wav	1

Table 5.5: Target Speaker Training Files (SPIDRE CD-ROM 2)

was determined by an examination of the typical variances obtained during training. A summary of the GMM training parameters used for both the cepstral and wavelet based feature vector sets is shown in Table 5.6. All symbols correspond to those used in Chapter 4.

Parameters Common To Cepstral and Wavelet Based Training		
Model Parameter Name	Parameter Symbol	Value
Acoustic Classes	M	16
Number Of Speakers	J	45
Cepstral Model Training		
Model Parameter Name	Parameter Symbol	Value
Feature Vector Dimension	K	20
Number Of Training Feature Vectors	T_T	3000
Minimum Variance	σ_{min}^2	0.01
Wavelet Based Model Training		
Model Parameter Name	Parameter Symbol	Value
Feature Vector Dimension	K	23
Number Of Training Feature Vectors	T_T	3750
Minimum Variance	σ_{min}^2	0.0001

Table 5.6: GMM Training Parameters

5.5.2 Speaker Identification Performance

Speaker identification performance is tested using the SPIDRE target speaker files which remain following the removal of the three files shown in Table 5.2 and the training files shown in Tables 5.4 and 5.5. This leaves 132 files for testing purposes, and ensures that the same text is not used for training and identification. For each test file, the target speaker and corresponding channel are identified from the header. Active frames for the target speaker are extracted and 45 likelihood functions are

calculated based on the speaker models formed during training and Equation 4.5. These likelihood functions are calculated on ten second intervals of speech. In other words, the parameter T_R in Equation 4.5 is chosen to correspond to a ten second interval of active speech. Ten second decision intervals were found to provide good speaker recognition performance in reference [1]. This decision interval corresponds to 500 cepstral feature vectors and 625 wavelet based feature vectors. The likelihood functions calculated over the ten second intervals are averaged over 90 seconds in order to calculate the average likelihood value for each speaker model. The speaker corresponding to the model having the largest average likelihood function is chosen as the most likely speaker. The percentage of test files where the true speaker is identified as the most likely speaker is used as an indication of the identification accuracy for the feature vectors. The identification accuracies obtained are shown in Table 5.7. Also shown is the percentage of files in which the true speaker appears in a list of the three most likely candidates following the ranking of the average likelihood values.

Feature Vector Set	Feature Vector Dimension	Identification Accuracy (%)	Top Three Identification Accuracy (%)
$\vec{C}_1$	20	69	80
$\vec{C}_2$	6	51	73
$\vec{W}_1$	23	39	58
$\vec{W}_2$	6	43	56
$\vec{W}_3$	12	44	64

Table 5.7: Speaker Identification Performance

The most successful feature vector for speaker identification purposes, of the five tested, is clearly $\vec{C}_1$ which is the same set of MEL spaced cepstral coefficients used in reference [1]. Identification accuracy obtained from these coefficients in this thesis is somewhat less than the ninety percent accuracy obtained in reference [1], but this is most likely a function of the number of speakers used in the database. The database in reference [1] used only sixteen speakers whereas a database of forty five speakers

is used in this thesis. When the training and testing procedure of Section 5.5 was repeated on a subset of the SPIDRE target files containing only twenty three target speakers, an identification accuracy of 86% was obtained using the $\vec{C}_1$ feature vector. It is also possible that some of the SPIDRE test files have significant levels of crosstalk and that this contributed to a lower identification accuracy, but this possibility was not investigated extensively since it was intended that the $\vec{C}_1$ feature vector be used as a baseline for the comparison of the wavelet derived feature vectors and the same files were used for the testing of all feature vectors.

The performance obtained from all the wavelet based feature vectors was disappointing. It was hoped that the two dimensional processing of the wavelet generated scaleogram would enhance speaker identification performance since transient characteristics occurring during plosives or stops could be better exploited using the wavelet transform. A speaker identification accuracy of ninety percent was obtained with the two dimensional processing of an FFT based scaleogram in reference [81], but this involved the use of a 150 dimensional feature vector. Using the values of $I' = 10$ and $J' = 15$ in Equation 5.8 might have provided higher performance, but the computational cost would be prohibitive especially in view of the fact that better performance can be obtained by simply averaging the energy in each frequency band of the FWT as in the case of the $\vec{W}_2$ feature vector. The fact that longer term averaging of energy in a given frequency band appears to enhance speaker identification performance would seem to confirm that the analysis of longer duration sounds such as vowels, and not transients, is more important for speaker identification purposes [50]. Not only did the averaging of the energy in each wavelet band significantly enhance the speaker identification accuracy over that obtained from the $\vec{W}_1$ feature vector, it also reduced the dimension of the required feature vector from twenty three to six, resulting in a considerable savings in computational requirements during training and identification phases. With this reduction, an attempt was made to augment the $\vec{W}_2$ feature vector set using delta parameters. The performance enhancement was marginal indicating that wavelet processing of the speech signal has a more fundamental limitation.

The main limitation of wavelet based processing of speech signals appears to be the half band restriction on the filter bank implemented during the wavelet transform.

As shown in Figures 3.9 and 3.12, the equivalent filter bank is constrained such that each bandpass filter is twice the bandwidth of the preceding bandpass filter and is also at twice the center frequency. This arrangement of filters appears to be too restrictive for the processing of speech signals for speaker identification work since it precludes a high resolution analysis of the frequency content in the range 0 - 1.0 kHz. This theory is confirmed by the fact that the $\vec{C}_2$ feature vector, which is generated using only the first seven filters of the triangular filter bank of Figure 5.6, obtains better speaker identification accuracy than all three wavelet derived feature vectors. Although the $\vec{C}_2$ feature vector uses only information in the frequency band 0 - 800 Hz it provides a higher speaker identification accuracy than the $\vec{W}_2$ feature vectors which uses information in the band 0 - 4.0 kHz. This attests to the importance of flexibility in the selection of frequency bands for the analysis of speech signals. Such flexibility is difficult to achieve with the constraints of the standard wavelet transform. The addition of another level to the wavelet transform operation simply halves the bandwidth of the lowpass filter. In the case of the filter bank used in this thesis, an increase of the FWT order from six to seven would have resulted in the insertion of a 25 Hz lowpass filter in the network of Figure 3.12. It would not have enhanced the resolution of the analysis in the important range 0 - 1.0 kHz.

Different filter coefficients used in the initialization of the FWT might offer some improvement in the identification accuracy obtained from wavelet derived feature vectors, but the restrictive half band structure of the filter bank would remain. Clever frequency conversion and filtering schemes may be used to move critical bands into a higher resolution area of the FWT bandpass filter bank, but in view of the relatively low complexity involved in the generation of the twenty dimensional cepstral based feature vector and its better performance, it appears to have the advantage over wavelet generated feature vectors.

Chapter 6

Conclusions

6.1 Discussion Of Results

As discussed in Section 5, the wavelet based parameters derived in this thesis failed to provide satisfactory performance in the speaker recognition application. The performance they did provide was substantially less than the speaker recognition performance provided by standard cepstral coefficients. The main reason for this poor performance appears to be that the restrictive halfband structure of the wavelet transform filter bank precludes a flexible, high resolution of spectral components in the 0 - 1 kHz frequency range. Proper analysis of this band appears to be important for speaker recognition applications.

6.2 Suggestions for Future Work

The half band restriction of the standard FWT has recently been addressed with the development of the Wavelet Packet Transform (WPT). This generalization of the FWT allows an arbitrary segmentation of the time frequency plane while preserving the numerical efficiency of the FWT [82, 83, 84]. This opens up the possibility of using a WPT to provide higher resolution of the critical frequency bands in the characterization of speech signals which may lead to an improvement in speaker identification performance of wavelet based feature vectors.

Bibliography

- [1] D. A. Reynolds, *A Gaussian Mixture Modeling Approach To Text-Independent Speaker Identification*. PhD thesis, Georgia Institute Of Technology, Atlanta, Georgia, U.S.A, 1992.
- [2] B. S. Atal, "Automatic Recognition Of Speakers From Their Voices," *Proceedings Of The IEEE*, pp. 460–475, April 1976.
- [3] G. R. Doddington, "Speaker Recognition - Identifying People By Their Voices." *Proceedings Of The IEEE*, pp. 1651–1664, November 1985.
- [4] "What Is The LDC?," in *Linguistic Data Consortium Newsletter*, University Of Pennsylvania: Linguistic Data Consortium, 1992.
- [5] "SPIDRE Documentation Files," in *Speaker Identification Research Corpus*. Liguistic Data Consortium, 1994.
- [6] D. O'Shaughnessy, *Speech Communication, Human and Machine*. Reading, Massachusetts: Addison-Wesley Publishing, 1987.
- [7] American Radio Relay League, *The 1987 ARRL Handbook For The Radio Amateur*. Newington, Connecticut: American Radio Relay League, 1987.
- [8] L. R. Rabiner and R. W. Schafer, *Digital Processing Of Speech Signals*. Englewood Cliffs, New Jersey: Prentice-Hall, 1987.
- [9] H. C. Mishkoff, *Understanding Artificial Intelligence*. Dallas, Texas: Texas Instruments Information Publishing Center, 1985.

- [10] R. Dale, E. Hovy, D. Rosner, and O. Stock, *Aspects Of Automated Natural Language Generation - 6th International Workshop On Natural Language Generation*. Berlin, Germany: Springer-Verlag, 1992.
- [11] M. Meteer, *Expressibility And The Problem Of Efficient Text Planning*. London, England: Pinter Publishers, 1992.
- [12] D. L. Waltz, *Semantic Structures, Advances In Natural Language Processing*. Hillsdale, New Jersey: Lawrence Erlbaum Associates, 1989.
- [13] R. D. Peacocke and D. H. Graf, "An Introduction To Speech and Speaker Recognition," *IEEE Computer Magazine*, pp. 26–33, August 1990.
- [14] "phoncode.doc documentation file," in *TIMIT Acoustic-Phonetic Continuous Speech Corpus*, National Institute Of Standards and Technology, 1990.
- [15] P. Ladefoged, *A Course In Phonetics*. San Diego, California: Harcourt Brace Jovanovich, 1982.
- [16] W. R. Zemlin, *Speech and Hearing Science: Anatomy and Physiology*. Englewood Cliffs, New Jersey: Prentice-Hall, 1968.
- [17] M. J. Hunt, "The Speech Signal," in *Speech Analysis and Synthesis and Man-Made Speech Communications For Air Operations*. NATO Advisory Group For Aerospace Research and Development, 1990.
- [18] N. S. Jayant, "Coding Speech At Low Bit Rates," *IEEE Spectrum*, pp. 58–63, August 1986.
- [19] J. L. Fike and G. E. Friend, *Understanding Telephone Electronics*. Dallas, Texas: Texas Instruments Learning Center, 1983.
- [20] L. S. Nored, *Course Notes - Analog and Digital Cellular Networks*. Washington, D.C.: George Washington University Continuing Education, 1995.
- [21] F. G. Stremler, *Introduction To Communications Systems*. Reading, Massachusetts: Addison-Wesley, 1982.

- [22] J. L. Flanagan, M. R. Schroeder, B. S. Atal, R. E. Crochiere, N. S. Jayant, and J. M. Tribolet, "Speech Coding," *IEEE Transactions On Communications*, pp. 710–737, April 1979.
- [23] A. Gersho and V. Cuperman, "Vector Quantization: A Pattern Matching Technique For Speech Coding," *IEEE Communications Magazine*, pp. 15–21, December 1983.
- [24] N. S. Jayant and P. Noll, *Digital Coding Of Waveforms*. Englewood Cliffs, New Jersey.: Prentice Hall, 1984.
- [25] R. Steele, *Mobile Radio Communications*. London, England.: Pentech Press, 1992.
- [26] International Telegraph and Telephone Consultative Committee, "Recommendation G.722: 7 kHz Audio Coding Within 64 kbit/s," in *Blue Book Volume III. Fascicle III.4*, International Telegraph and Telephone Consultative Committee, 1989.
- [27] T. E. Tremain, "The Government Standard Linear Predictive Coding Algorithm: LPC-10," *Speech Technology*, pp. 40–49, April 1982.
- [28] *NATO STANAG 4198: Parameters and Coding Characteristics That Must Be Common To Assure Interoperability Of 2400 bps Linear Predictive Encoded Digital Speech*. North Atlantic Treaty Organization, 1984.
- [29] T. E. Tremain, J. P. Campbell, and V. C. Welch, "A 4.8 kbps Code Excited Linear Predictive Coder," *Proceedings Of The Mobile Satellite Conference*, pp. 491–496, 1988.
- [30] R. M. Grey, "Vector Quantization," *IEEE ASSP Magazine*, pp. 4–29, April 1984.
- [31] J.-H. Chen, R. V. Cox, Y.-C. Lin, N. Jayant, and M. J. Melchner, "A Low-Delay CELP Coder For The CCITT 16 kb/s Speech Coding Standard," *IEEE Journal On Selected Areas In Communications*, pp. 830–849, June 1992.

- [32] L. R. Rabiner, B. S. Atal, and J. Flanagan, "Current Methods Of Digital Signal Processing," in *Speech Analysis and Synthesis and Man-Made Speech Communications For Air Operations*, NATO Advisory Group For Aerospace Research and Development, 1990.
- [33] D. J. Rahikka, T. E. Tremain, V. C. Welch, and J. P. Campbell, "CELP Coding For Land Mobile Radio Applications," *Proceedings Of The IEEE International Conference On Acoustics, Speech and Signal Processing*, pp. 465–468, 1990.
- [34] J. P. Campbell, V. C. Welch, and T. E. Tremain, "An Expandable Error-Protected 4800 bps CELP coder (U.S. Federal Standard 4800 bps Voice Coder)," *Proceedings Of The IEEE International Conference On Acoustics, Speech and Signal Processing*, pp. 735–738, 1989.
- [35] I. A. Gerson and M. A. Jasiuk, "Vector Sum Excited Linear Prediction (VSELP) Speech Coding At 8 kbps," *Proceedings Of The IEEE International Conference On Acoustics, Speech and Signal Processing*, pp. 461–464, 1990.
- [36] H. Fletcher, *Speech and Hearing In Communication*. New York, New York: D. Van Nostrand Company Inc., 1953.
- [37] X. Yang, K. Wang, and S. A. Shamma, "Auditory Representations Of Acoustic Signals," *IEEE Transactions On Information Theory*, pp. 824–839, March 1992.
- [38] R. E. Ziemer and W. H. Tranter, *Principles Of Communications*. Boston, Massachusetts: Houghton Mifflin Company, 1985.
- [39] *Xwaves and ESPS Documentation*. Washington, D.C.: Entropic Research Laboratory Limited, 1993.
- [40] S. Haykin, *Adaptive Filter Theory*. Englewood Cliffs, New Jersey.: Prentice Hall, 1986.
- [41] A. V. Oppenheim and R. W. Schaffer, *Digital Signal Processing*. Englewood Cliffs, New Jersey.: Prentice Hall, 1975.

- [42] *Proceedings Of Speech Research Symposium XII*. The Speech Research Organization, 1992.
- [43] *Proceedings Of The Canadian Speech Processing Workshop*. Consulting and Audit Canada, February, 1991.
- [44] H. L. Van Trees, *Detection, Estimation and Modulation Theory: Part I*. New York, New York: John Wiley and Sons, 1967.
- [45] E. A. Patrick, *Fundamentals Of Pattern Recognition*. Englewood Cliffs. New Jersey: Prentice-Hall Inc., 1972.
- [46] R. P. Lippmann, "An Introduction To Computing With Neural Nets." *IEEE ASSP Magazine*, pp. 4-22, April 1987.
- [47] K. Fukunaga, *Introduction To Statistical Pattern Recognition*. San Diego, California: Academic Press Inc, 1990.
- [48] L. R. Rabiner, "A Tutorial On Hidden Markov Models and Selected Applications In Speech Recognition," *Proceedings Of The IEEE*, pp. 257-285, February 1989.
- [49] D. B. Roe and J. C. Wilpon, "Whither speech recognition: The next 25 years." *IEEE Communications Magazine*, pp. 54-62, November 1993.
- [50] D. O'Shaughnessy, "Speaker Recognition," *IEEE ASSP Magazine*, pp. 4-17. October 1986.
- [51] L. R. Rabiner, A. E. Rosenberg, and S. E. Levinson, "Considerations In Dynamic Time Warping Algorithms For Discrete Word Recognition," *IEEE Transactions On Acoustics, Speech, and Signal Processing*, pp. 575-582, December 1978.
- [52] H. Sakoe and S. Chiba, "Dynamic Programming Algorithm Optimization For Spoken Word Recognition," *IEEE Transactions On Acoustics, Speech, and Signal Processing*, pp. 43-49. February 1978.
- [53] H. Abut (editor). *Vector Quantization*. New York. New York: IEEE Press. 1990.

- [54] F. K. Soong and A. E. Rosenberg, "On The Use Of Instantaneous and Transitional Spectral Information In Speaker Recognition," *IEEE Transactions On Acoustics, Speech, and Signal Processing*, pp. 871-879, June 1988.
- [55] S. Haykin, *Neural Networks: A Comprehensive Foundation*. New York, New York.: Macmillan College Publishing Company Inc., 1994.
- [56] L. R. Rabiner and B. H. Juang, "An Introduction To Hidden Markov Models." *IEEE ASSP Magazine*, pp. 4-16, January 1986.
- [57] O. Rioul and M. Vetterli, "Wavelets and Signal Processing," *IEEE Signal Processing Magazine*, pp. 14-38, October 1991.
- [58] F. J. Harris, "On The Use Of Windows For Harmonic Analysis With The Discrete Fourier Transform," *Proceedings Of The IEEE*, pp. 51-83, January 1978.
- [59] T. Nohara and S. Haykin, "The Wavelet Transform: A Tool For Time/Frequency Signal Processing With Applications In Radar," *Proceedings Of The Canadian Conference On Electrical And Computer Engineering*, September 1990.
- [60] L. G. Weiss, "Wavelets and Wideband Correlation Processing," *IEEE Signal Processing Magazine*, pp. 13-32, January 1994.
- [61] D. P. Casasent, J.-S. Smokelin, and A. Ye, "Wavelet and Gabor Transforms For Detection," *Optical Engineering*, pp. 1893-1897, September 1992.
- [62] B. Vidakovic and P. Muller, "Wavelets for kids," *Unpublished: Available by anonymous FTP at ftp.isds.duke.edu*.
- [63] J. Carey, "Wavelets Are Causing Ripples Everywhere," *Business Week*, February 3 1992.
- [64] A. Graps, "An Introduction To Wavelets," *IEEE Computational Science and Engineering Magazine*, Summer 1995.

- [65] S. Kadambe and G. F. Boudreaux-Bartels, "Application Of The Wavelet Transform For Pitch Detection Of Speech Signals," *IEEE Transactions On Information Theory*, pp. 917–924, March 1992.
- [66] M. V. Wickerhauser, "Acoustic Signal Compression With Wave Packets," *Unpublished: Available by anonymous FTP at pascal.math.yale.edu*.
- [67] R. R. Coifman, Y. Meyer, S. Quake, and M. V. Wickerhauser, "Signal Processing and Compression With Wave Packets," *Unpublished: Available by anonymous FTP at pascal.math.yale.edu*.
- [68] I. Daubechies, "The Wavelet Transform, Time-Frequency Localization and Signal Analysis," *IEEE Transactions On Information Theory*, pp. 961–1005, September 1990.
- [69] R. K. Young, *Wavelet Theory and Its Applications*. Boston, Massachusetts: Kluwer Academic Publishers Inc., 1993.
- [70] P. Flandrin, "Wavelets and Related Time-Scale Transforms," in *Proceedings Of SPIE Conference 1348: Advanced Signal Processing Algorithms, Architectures, and Implementations*, Society Of Photo-Optical Instrumentation Engineers, 1990.
- [71] H. J. Caulfield and H. H. Szu, "Parallel Discrete and Continuous Wavelet Transforms," *Optical Engineering*, pp. 1835–1839, September 1992.
- [72] Y. Sheng, D. Roberge, and H. H. Szu, "Optical Wavelet Transform," *Optical Engineering*, pp. 1835–1839, September 1992.
- [73] Y. Chan, "Wavelet Lecture Notes," *Royal Military College*.
- [74] I. Daubechies, "Orthonormal Bases Of Compactly Supported Wavelets," *Communications On Pure and Applied Mathematics*, vol. XLI, pp. 909–996, 1988.
- [75] M. Vetterli and C. Herley, "Wavelets and Filterbanks: Theory and Design," *IEEE Transactions On Signal Processing*, pp. 2207–2232, September 1992.

- [76] J. C. Kantor, "Wavelet Toolbox Reference," *Unpublished: Available via email at kantor@nd.edu*, 1993.
- [77] J. C. Kantor, "Private Communication," *University Of Notre Dame, South Bend, Indiana, USA*, 1993.
- [78] G. Strang, "Wavelets and Dilation Equations: A Brief Introduction," *SIAM Review*, pp. 614–627, December 1989.
- [79] W. H. Press, B. P. Flannery, S. A. Teukolsky, and W. T. Vetterling, *Numerical Recipes In C: The Art Of Scientific Computing (Second Edition)*. Cambridge, U.K.: Cambridge University Press, 1992.
- [80] M. D. Van Der Laan, "Private Communication," *University Of Groningen, Groningen, Netherlands*, 1993.
- [81] H. W. Strube, D. Helling, A. Krause, and M. R. Schroeder, "Word and Speaker Recognition Based On Entire Words Without Framewise Analysis," *Bibliotheca Phonetica*, vol. 12, pp. 80–114, 1985.
- [82] B. B. Hubbard, *The World According To Wavelets*. Wellesley, Massachusetts: A. K. Peters Inc., 1996.
- [83] M. A. Cody, "The Wavelet Packet Transform - Extending The Wavelet Transform," *Dr. Dobbs's Journal*, April 1994.
- [84] G. Strang and T. Nguyen, *Wavelets and Filter Banks*. Wellesley, Massachusetts: Wellesley-Cambridge Press, 1996.