Investigation of the impact of High Frequency transmitted speech on Speaker Recognition

Jan Pool

March 2002

Study Leaders: Prof. J.A. du Preez

Prof. J.G. Lourens

Thesis presented in partial fulfilment of the requirement for the degree of

Master of Electronic Engineering at the *University of Stellenbosch*

Stellenbosch University http://scholar.sun.ac.za

I, the undersigned, hereby declare that the work contained in this thesis, is my original work and has not previously in its entirety or in part been submitted at any university for a degree.

Signature Date:

Stellenbosch University http://scholar.sun.ac.za

Abstract

Speaker recognition systems have evolved to a point where near perfect performance can be obtained under ideal conditions, even if the system must distinguish between a large number of speakers. Under adverse conditions, such as when high noise levels are present or when the transmission channel deforms the speech, the performance is often less than satisfying.

This project investigated the performance of a popular speaker recognition system, that use Gaussian mixture models, on speech transmitted over a high frequency channel. Initial experiments demonstrated very unsatisfactory results for the base line system.

We investigated a number of robust techniques. We implemented and applied some of them in an attempt to improve the performance of the speaker recognition systems. The techniques we tested showed only slight improvements.

We also investigates the effects of a high frequency channel and single sideband modulation on the speech features of speech processing systems. The effects that can deform the features, and therefore reduce the performance of speech systems, were identified.

One of the effects that can greatly affect the performance of a speech processing system is noise. We investigated some speech enhancement techniques and as a result we developed a new statistical based speech enhancement technique that employs hidden Markov models to represent the clean speech process.

Stellenbosch University http://scholar.sun.ac.za

Opsomming

Sprekerherkenning-stelsels het 'n punt bereik waar naby aan perfekte resultate verwag kan word onder ideale kondisies, selfs al moet die stelsel tussen 'n groot aantal sprekers onderskei. Wanneer nie-ideale kondisies, soos byvoorbeeld hoë ruisvlakke of 'n transmissie kanaal wat die spraak vervorm, teenwoordig is, is die resultate gewoonlik nie bevredigend nie.

Die projek ondersoek die werksverrigting van 'n gewilde sprekerherkenning-stelsel, wat gebruik maak van Gaussiese mengselmodelle, op spraak wat oor 'n hoë frekwensie transmissie kanaal gestuur is. Aanvanklike eksperimente wat gebruik maak van 'n basiese stelsel het nie goeie resultate opgelewer nie.

Ons het 'n aantal robuuste tegnieke ondersoek en 'n paar van hulle geïmplementeer en getoets in 'n poging om die resultate van die sprekerherkenning-stelsel te verbeter. Die tegnieke wat ons getoets het, het net geringe verbetering getoon.

Die studie het ook die effekte wat die hoë-frekwensie kanaal en enkel-syband modulasie op spraak kenmerkvektore, ondersoek. Die effekte wat die spraak kenmerkvektore kan vervorm en dus die werkverrigting van spraak stelsels kan verlaag, is geïdentifiseer.

Een van die effekte wat 'n groot invloed op die werkverrigting van spraakstelsels het, is ruis. Ons het spraak verbeterings metodes ondersoek en dit het gelei tot die ontwikkeling van 'n statisties gebaseerde spraak verbeteringstegniek wat gebruik maak van verskuilde Markov modelle om die skoon spraakproses voor te stel.

Stellenbosch University http://scholar.sun.ac.za

This work is dedicated to my parents, Jan and Gerda, and to Jeanine Engelbrecht.

Stellenbosch University http://scholar.sun.ac.za

Acknowledgements

I would like to thank the following people and entities:

- My supervisor, Prof Johan du Preez, for his constant support and guidance and especially for being such an endless supply of ideas and knowledge.
- My co-supervisor, Prof J.G. Lourens, for his help with the first part of the study, especially the work related to HF channels.
- Jeanine Engelbrecht, for always being there for me and encouraging me all the way, especially when morale and will power was low.
- The guys in our research lab, who helped to create a fun and enjoyable work environment full of valuable knowledge.
- All the people over the world who help developed Linux and all the high quality free software that I used daily. Without them the world would have been a poorer place.
- Everyone who have, over the many years, contributed to our speech processing group's pattern recognition software.
- The NRF for their financial support.

Stellenbosch University http://scholar.sun.ac.za

List of Abbreviations

ACW Adaptive component weighting

ADC Analogue-to-digital convertion

AGC Adaptive gain control

AR Autoregressive

BC Bayes classifier

BSVQ Binary Split vector quantisation

CD Compact disk

CMS Cepstral mean subtraction

CPF Cepstral post filtering

CW Cepstral weighting

DAC Digital-to-analog converter

DSP Digital signal processing

DZC Discarding zero order cepstral coefficient

EM Expectation maximisation

FIT Fast incremental training

FFT Fast Fourier transform

Stellenbosch University http://scholar.sun.ac.za

FIR Finite impulse response

FT Fourier transform

GM Gaussian model

GMM Gaussian mixture model

HF High frequency

HMM Hidden Markov model

HT Hilbert transform

IFFT Inverse fast Fourier transform

KMVQ K-means vector quantisation

LP Linear prediction

LPC linear predictive coding

LPCC Linear prediction coefficient cepstrum

MA Moving average

MAP Maximum a posteriori

MC Markov chain

MFCC Mel-scale frequency cepstral coefficients

ML Maximum likelihood

MLP Multi layer perceptron

MMSE Minimum mean square error

NN Neural networks

NUBSVQ Non-uniform binary split vector quantisation

PDF Probability density function

Stellenbosch University http://scholar.sun.ac.za

PFCMS Pole-filtered cepstral mean subtraction

PLP Perceptual linear prediction

RAS Relative autocorrelation sequence

RAS-MFCC Relative autocorrelation sequence Mel-scale frequency cepstral coefficients

RASTA RelAtive SpecTrAl

RBF Radial basis function

SE Speech enhancement

SNR Signal-to-noise ratio

SQNR Signal-to-quantisation noise ratio

SR Speaker recognition

SSB Single sideband modulation

TDNN Time delay neural network

VQ Vector quantisation

Stellenbosch University http://scholar.sun.ac.za

Notation and operations

Analog signals

$x(t)$ Analog signal.

$F\{\cdot\}$ Fourier transform operation.

$F^{-1}\{\cdot\}$ Inverse Fourier transform operation.

$X(\omega)$ Fourier transform pair of $x(t)$.

$H(\omega)$ Transfer function in the frequency domain.

Digital signals

$x(n)$ Digital signal.

$\eta(n)$ Noise signal.

$Z\{\cdot\}$ Z-transform operation.

$Z^{-1}\{\cdot\}$ Inverse z-transform operation.

$X(z)$ Z-transform pair of $x(n)$.

$H(z)$ Transfer function in the z-domain.

Matrix and Vector

$\mathbf{M}$ Any bold upper case letter and Greek symbols denotes a matrix.

Stellenbosch University http://scholar.sun.ac.za

$\mathbf{v}$ Any bold lower case letter and Greek symbols denotes a vector.

$(\cdot)^T$ Transpose of input.

$(\cdot)^*$ Complex conjugate of input.

$(\cdot)^H$ Hermitian transposition of input.

Statistical

$P(A)$ The probability of event A.

$P(A|B)$ The probability of event A given that event B occurred.

$E[\cdot]$ Expectation of a function.

$f(\mathbf{x}|\lambda)$ The probability density function (PDF) of $\mathbf{x}$, given the model λ.

$r(n)$ Autocorrelation parameter.

Linear prediction and the cepstrum

a_k The k'th autoregressive (AR) parameter or linear prediction (LP) coefficient.

r_k Residue of the k'th AR parameter.

p_k Pole of the k'th AR parameter.

w_k The k'th linear predictor tap weight.

f_M Forward prediction error.

P_M Minimum mean squared prediction error.

$\mathbf{R}$ Correlation matrix of an input tap vector.

$\mathbf{r}$ Cross-correlation between input tap vector and the desired response.

$\mathbf{w}$ Tap weight vector.

$c_x(n)$ Complex cepstrum of sequence $x(n)$.

Stellenbosch University http://scholar.sun.ac.za

Speaker recognition

m_i Gaussian mixture weights.

$\boldsymbol{\mu}$ Gaussian PDF mean vector.

$\boldsymbol{\Sigma}$ Gaussian PDF covariance matrix.

λ Compact notation for Gaussian mixture model (GMM) parameters.

$\beta(\lambda_a|\lambda_b)$ Speaker identification cost function. The cost of selecting λ_a if the observed feature vector belonged to λ_b.

$d(\lambda_a, \lambda_b)$ Distance measure between two speaker models.

Hilbert transform and single sideband modulation

$x(t)$ Transmitted signal.

$\widehat{x}(t)$ Hilbert transformed signal of $x(t)$.

$\widetilde{x}(t)$ Demodulated signal.

$y_+(t)$ Upper sideband.

$y_-(t)$ Lower sideband.

ω_c Carrier frequency.

$\Delta\omega$ Frequency error.

θ Phase error.

Hidden Markov model

S_i The i'th state.

q_n State at time n.

Stellenbosch University http://scholar.sun.ac.za

a_{ij} Transitional probability, the probability of going from the current state, i, to the next state, j.

$\mathbf{A}$ Transition probability matrix.

O_i The i'th observation.

π_i The initial probability of state i.

$\boldsymbol{\pi}$ Initial probability vector.

V_i Observation sequence corresponding to the output of an HMM.

$b_j(k)$ Observation probability of the k'th possible observation of the j'th state.

$\mathbf{B}$ Observation probability matrix.

λ Compact notation for HMM.

Other

$\Re$ Real numbers.

$S(\omega)$ Power spectrum.

$w(n)$ Window function.

$u(t)$ Unit step function.

$sgn(w)$ Signum function.

Stellenbosch University http://scholar.sun.ac.za

Contents

Stellenbosch University http://scholar.sun.ac.za

Stellenbosch University http://scholar.sun.ac.za

Stellenbosch University http://scholar.sun.ac.za

Stellenbosch University http://scholar.sun.ac.za

Stellenbosch University http://scholar.sun.ac.za

Stellenbosch University http://scholar.sun.ac.za

Stellenbosch University http://scholar.sun.ac.za

List of Figures

Stellenbosch University http://scholar.sun.ac.za

Stellenbosch University http://scholar.sun.ac.za

Stellenbosch University http://scholar.sun.ac.za

List of Tables

Stellenbosch University http://scholar.sun.ac.za

Part I

Background

Stellenbosch University http://scholar.sun.ac.za

Chapter 1

Introduction

1.1 Motivation

The field of speech processing attracted much research during the past few decades. The field matured to a point where many applications, such as speech recognition, speaker identification/verification and language recognition, are achieving near perfect recognition results under ideal or well controlled conditions. For example, a less than 0.5% error rate for speaker independent connected digit recognition using the TIDIGITS database [1], about 5% error rate for speaker-independent continuous speech recognition of a 1000 word vocabulary in the ARPA Resource Management task [2], and less than 0.5% recognition error for speaker identification with 630 speakers using the TIMIT database [3]. However, in practical applications these recognisers have to operate under conditions that are very different from the ideal laboratory environments. The speech can be contaminated by additive noise and by the transfer function of the transmission channels (convolutional noise). For example, the transfer functions of telephone channels differ from one another and also change over time. The presence of these effects have become a major obstacle to commercial use of speech recognition techniques.

It is well known that recognition accuracy drops rapidly if noise is added to the speech or if the testing and training conditions mismatch, even if the system is used in better signal-to-noise ratio (SNR) environments than what the system was trained in [4, 5]. Efforts have been

 Stellenbosch University http://scholar.sun.ac.za

made to reduce the mismatch between training and operating conditions. One technique is to train different models for various SNR conditions [5, 6]. While this method demonstrates improvement in recognition results, it is a very impractical solution. A more realistic way to try to improve on the situation is to search for ways to improve the robustness of the techniques used. This includes using models or feature vectors that are more invariant to noise and channel changes, enhancing the corrupted speech, or adapting the speech models.

In addition to work on basic techniques, a significant amount of research is done on more specialised situations and applications. This project started out in this direction. We were faced with the problem of speaker identification on speech transmitted over high frequency (HF) channels. An HF channel introduces some interesting problems, such as fading, Doppler shifts and multi-path signals.

Initial experiments, using standard techniques, produced very unsatisfactory results, even compared to that of telephone channels. Subsequent studies, employing robust techniques, didn't show much improvement in the results. It was therefore decided to study the HF channel in order to determine the causes of such bad performance. Based on the analysis the second part of this study examined possible ways to improve the situation.

1.2 Background

There are four notable levels where attempts can be made to increase the robustness and performance of speech based pattern recognisers:

- The most basic level is the speech itself. Any improvements made on this level will propagate through the system and consequently make the work of the consecutive stages easier. Work done on this level consists of enhancing the quality of speech, such as noise removal.
- The next is the feature level. The choice of the correct feature representation is a difficult one. A feature vector performing well in one situation is not always the best choice in other situations. The most popular features in speech processing systems are the linear

 Stellenbosch University http://scholar.sun.ac.za

prediction coefficient cepstrum and Mel-warped cepstrum. Improvements on this level consists of modifying the basic cepstral features and in some cases using new or hybrid feature vectors.

- The third level considers the models and classifiers used for training and testing. The aim here is to choose a model that represents the pattern to be recognised as close as possible under the given conditions and with the available training data.
- The fourth level consists of processing the likelihood scores. On this level techniques are introduced to process the available scores from the classifier in such a way as to improve the overall performance of the system.

1.2.1 Speech enhancement

In all speech processing systems the presence of background interference causes the quality and intelligibility of speech to degrade, resulting in decreased performance. The purpose of speech enhancement algorithms are to reduce background noise, minimise channel transformations, suppress interfering background speakers, or to improve the quality of the speech. This report will focus on algorithms designed to remove additive background noise, thereby possibly improving the quality and intelligibility of speech signals. Background noise includes broadband noise, such as Gaussian white noise, that is added to the speech signal. By improving the quality of corrupted speech we also improve the performance of speech processing systems, such as speech and speaker recognition.

When evaluating speech enhancement algorithms we need some objective measure, such as signal-to-noise ratios, articulation index or the Itakura measure, to compare or quantify the level of noise present. However, it should be noted that these measures are not always in accordance with the subjective perception of a human listener [7, p.502]. It could be that a speech enhancement system that improves the SNR of a speech signal inadvertently degrades the intelligibility of the spoken words. Depending on the application of the enhanced speech it might be more desirable to rather increase the intelligibility (perceptive quality) of the speech instead of its objective measured quality.

 Stellenbosch University http://scholar.sun.ac.za

Speech enhancement methods can broadly be classified into techniques based on stochastic processing of speech models and techniques based on perceptual aspects of speech. There can further be differentiated between methods that work on a single channel and methods that require multiple channels to function properly. Most applications of interest to us have only one available channel. Examples of this are telephone and radio communication channels. In this light we will put more emphasis on single channel speech enhancement techniques.

1.2.2 Feature vector analysis

One of the most important aspects of any speech processing system is the extraction of feature vectors from the speech signal. Feature selection is the process of mapping the original measurements into a more effective space. Digitised speech usually consists of large amounts of data. While all this information is required to represent the speech waveform, the speech process can be represented by much less. This is due to the fact that the characteristics of speech change slowly compared to that of the speech waveform. The goals of feature extraction is to compress the available information in such a way as to select the best features to represent the speech signal for the application at hand.

1.2.3 Models and classifiers

The heart of a speaker recognition system lies in the choice of model to represent the speakers. The model should exploit some characteristics of the speaker. In this report only probabilistic models will be considered. A probabilistic model uses probability distributions to model speakers and make classification decisions using probabilities.

Probabilistic models can be divided into parametric and non-parametric models. Non-parametric models assume very little about the probability density function. Examples of non-parametric models are nearest neighbour [8] and vector quantisation (VQ) [9, 10] models. Parametric models assume a predefined structure that is characterised by the structure parameters. Examples of parametric models are Gaussian and Gaussian mixture models (GMM) [11, 12]. The advantage

 Stellenbosch University http://scholar.sun.ac.za

of parametric models are that less data are required to specify the model [6]. The drawback is that the restricted structures may not be adequate for the modelling task [6].

The GMM is currently one of the most popular speaker models and has become the standard for both speaker identification and verification. It is also the model used in this project.

1.2.4 Scoring

The recognition system's classifier returns a score, usually a log likelihood score, for each model that participated in the classification process. For an utterance under test the log likelihood scores of each model can be summed (under the assumption that the feature vectors are independent) to produce a total score for the whole utterance. The model with the highest score is then selected as the one with the highest likelihood to represent the given test data.

A few robust scoring methods are discussed in [6], of which the two most significant of these are mentioned here. In both methods the segments evaluated are ranked from the lowest to highest scores. The first method sums only the top N scores of each model and then select the model that produced the highest summed log likelihood score. The idea being that, under adverse conditions, the highly contaminated speech segments will have equally low scores for all the models, while the less contaminated regions will still score high for the correct speaker. We therefore only consider the top N scores under the assumption that the data vectors that produced the highest scores suffered the least from the speech contamination. The second method sums the top N to M ranked scores of each model and then select the model with the highest summed score. In other words the N best scores are selected and from these the top M scores are discarded. The reason is that anomalous events can occur that may score very high for one speaker model, but score low for all other models. This method try to eliminate these events at the price of possibly excluding valid segments. In the experiments provided in [6], the N best method performed consistently better than summing all the available scores. It also outperforms the M to N method in most test cases. This demonstrates that throwing away some of the top scores are, in general, not a good idea.

Stellenbosch University http://scholar.sun.ac.za

1.3 Literature study

1.3.1 Speech enhancement

In this section we present a number of speech enhancement techniques. The first technique, spectral subtraction, removes the additive noise component in the spectral domain. The second technique, noise masking, uses a phenomenon of the human auditory system to perceptually suppress the noise signal. In the last section we will discuss a number of statistical based speech enhancement techniques. In addition to these methods, the interested reader can also investigate wavelet de-noising [13, 14, 15]. Due to time restrictions this technique was not pursued in this study.

1.3.1.1 Spectral subtraction

In spectral subtraction [16] the assumptions are made that the noise and speech are uncorrelated and additive in the time domain and that the noise characteristics change slowly compared to that of the speech. If this is true the power spectrum of the noisy signal is the sum of the speech and noise power spectrum. The noise spectrum can be estimated during silent periods of the speech signal. This estimate is then subtracted from the power spectrum of the noisy signal to obtain an estimate of the power spectrum of the original uncorrupted speech signal.

While fairly simple and effective this method exposes the following problems:

- This method relies on detecting silent periods in the speech signal to estimate the noise power spectrum. Consequently, if there are only a few silent segments in the speech, the algorithm might use an unvoiced speech segment to obtain the noise estimate, which is obviously not desirable.
- By blindly subtracting the estimated power spectrum of the noise from the power spectrum of the speech signal, the resulting spectrum may have negative values. An ad hoc technique is then required to make the negative values positive. In some cases the negative

Stellenbosch University http://scholar.sun.ac.za

values are set to zero and in other cases it is set to a small positive value. This non-linear operation produces a type of noise that is commonly termed musical noise [7, p.510].

1.3.1.2 Noise masking

Noise masking [17] is a psychological phenomenon based on the human perception of speech. A person can not detect an acoustic stimulus whose level is lower than the masking threshold generated by another acoustic stimuli. Stated differently: the human auditory system is incapable of distinguishing two signal components close in the time or frequency domain. When listening to speech corrupted with noise, some of the noise is perceptually suppressed by the masking phenomena.

Noise masking is usually used as a preprocessing step for speech processing, not as a method to enhance the speech quality for listening. Since this technique tries to emulate the human masking system, there wouldn't be much difference to the human ear if the synthetic masking operation succeeded. Examples of the application of noise masking can be found in [17, 18, 19, 20].

When a speech signal is corrupted by high noise levels, noise masking may have no effect. This is due to the fact that even the higher energy bands of the speech could contain less energy than the noise mean [21].

The threshold for noise masking approximately corresponds to the background noise levels and it requires an accurate estimate for proper operation. No information about the speech can then be derived from observations that fall below this level.

1.3.1.3 Statistical modelling and estimation

A more exciting category of speech enhancement is the collection of techniques based on statistical analysis of the speech and noise signals.

One of the first approaches [22] is autoregressive (AR) all-pole modelling and Wiener filtering

UNIVERSITEIT STELLENBOSCH
BIBLIOTEEK

Stellenbosch University http://scholar.sun.ac.za

of the speech. This method is based on iterative *maximum a posteriori* (MAP) estimation of the linear prediction (LP) coefficients and the enhanced speech signal. This method requires knowledge about the noise variance to construct adaptive noncausal Wiener filters. The algorithm can be divided into two iterative steps: Firstly the LP coefficients from the current speech estimate area estimated. Secondly, an adaptive Wiener filter from the LP coefficients and noise variance estimate is constructed. Then the Wiener filter is used to obtain the next clean speech estimate by filtering the noisy speech.

A similarity with the expectation maximisation (EM) algorithm [23] is observed here. The observed noise corrupted speech can be viewed as the incomplete data, while the clean speech is the complete data. The clean speech can be iteratively estimated from the observed noise corrupted speech using the EM algorithm.

The previous method is known as unconstrained MAP estimation of speech. A constrained MAP estimation was introduced [24] in which vocal tract specific constraints were placed on the spectrum of the enhanced speech between MAP iterations. Some of these constraints are based on perceptually important characteristics, such as the stability of the all-pole speech models, relative pole positions of the speech models and the inertia of the vocal tract characteristics (the possible rate of change). The modifications were triggered by the following observations:

- The formants shifted their location and their bandwidth decreased with every iteration.
- Frame-to-frame pole jittering was introduced over time.
- A heuristic convergence criterion must be used for sequential MAP estimation.
- The original technique employs no frame-to-frame constraints.

The first two effects contributed to unnatural sounding speech. The improved enhancement algorithm with the constraints over time (inter-frame) and iterations (intra-frame) attempts to:

- Stabilise the all-pole speech models.
- Ensure that the poles are not too close to the unit circle, so that the pole bandwidth is not too narrow. This produces more natural sounding speech.

Stellenbosch University http://scholar.sun.ac.za

- The vocal tract characteristics do not vary too much from frame to frame in the presence of speech.

The constrained MAP estimation improves aspects that are of importance to human perception. The estimation of the AR parameters can also be replaced by a constrained codebook system [25].

It was also noted that, while speech is non-stationary, a stationary Wiener filter was employed in the above cases. The Wiener filter can be substituted with a Kalman filter [26], which performs better than the Wiener filter. A further improvement suggested a delayed Kalman filter. The disadvantage of the Kalman filter is that it is computationally more expensive than a Wiener filter.

In practice the all-pole enhancement system has notable limitations. The noisy speech samples do not contain enough information to yield accurate estimators for both the model parameters and the speech signal itself. The problem of insufficient data can be eliminated by constructing the model from clean speech signals [27]. It has also been suggested that the clean speech estimator should be dependent on speech characteristics or classes [28]. One such series of methods employ hidden Markov model (HMM) states and classes [28, 29, 30].

A technique using HMMs is discussed in [28, 29, 31]. One HMM is used as a clean speech model and another HMM as a noise model. Each of these two models consists of a number of states. Each state is a mixture Gaussian AR model. After training, a Wiener filter is constructed for each of the possible speech/noise state combinations. The clean speech is then estimated from the noisy speech using either a MAP or a minimum mean square error (MMSE) estimator. In both cases Viterbi decoding is used to determine the speech/noise state combinations. In the MMSE case the noisy speech is filtered by all the Wiener filters. The clean speech estimate is then obtained from a weighted sum of the filtered signals. The weights for the filters are the corresponding probability of the speech/noise state combinations as determined by the Viterbi decoder. The MMSE estimator is said to make a soft decision, because the clean speech estimate is a weighted sum of all the filter outputs. The MAP estimator iteratively estimates the speech/noise state combination and the clean speech estimate. The speech/noise state combi-

 Stellenbosch University http://scholar.sun.ac.za

nation is obtained from the current clean speech estimate using a Viterbi decoder. The noisy speech is then filtered, with the Wiener filter corresponding to the most likely speech/noise state combination, to produce the next clean speech estimate. The MAP estimator is said to make a hard decision, because only the output of the most likely Wiener filter is used at each iteration. The Wiener filters can also be replaced by Kalman filters under a dynamic system framework [32].

1.3.2 Feature vector analysis

Over the years various speech features have been investigated. Among these are intensity [33], pitch [34], short-time spectrum [35], LP cepstrum [36], formants [37], nasal co-articulation [38], spectral correlation [38], harmonic features [39] and cepstral measures [33]. The most popular speech feature in recent years has been the cepstrum and it is suggested that it is superior for speech processing applications [10, 40]. Of these, the LP cepstrum and Mel-warped cepstrum are frequently used. In this study we will mainly focus on the LP cepstrum.

In attempts to increase the robustness of speech processing systems on the feature level, researchers have either attempted to modify the features or to derive new types of features from the LP cepstrum or Mel-warped cepstrum. Usually techniques in this category focuses on minimising the effect of noise, rather than removing the noise itself.

Here follows a brief synopsis of the most important techniques. A more detailed discussion is presented in Chapter 4.

1.3.2.1 Robust features

Perceptual linear prediction (PLP)

Perceptual linear prediction [41] is based on the short-term spectrum of speech that has been modified by psychophysical spectral transformations. The PLP technique attempts to simulate some of the properties of hearing namely:

Stellenbosch University http://scholar.sun.ac.za

- Critical-band resolution curve.
- Equal-loudness curve,
- Intensity-loudness power-law relation.

After these modifications have been applied in the frequency domain, LP coefficients are calculated (more detail in Section 4.1.1) to form a new speech feature. While this feature improves on the robustness of speech recognition system it is still (just like other features that are based on the short-time spectrum) susceptible to changes in the frequency response of the communication channel and noise.

Relative autocorrelation sequence (RAS) mel-scale cepstrum

This technique attempts to remove the effects of additive white (and coloured) noise by filtering the temporal trajectories of short-time, one-sided autocorrelation sequences [42]. Speech corrupted by uncorrelated noise has an additive component in the time, spectral and autocorrelation domain. The assumption is further made that the additive component is constant over a short period of time. The autocorrelation sequences are differentiated over time, since constant factors are removed by differentiation. These new sequences are called the relative autocorrelation sequences (RASs). To form the speech feature these RASs are used (instead of the original speech) to calculate the mel-scale frequency cepstrum. The combination is called the RAS-MFCC. This technique will be discussed in more detail in Section 4.1.2.

1.3.2.2 Intra-frame processing of cepstral features

Discarding the zero'th order cepstral coefficient

The zero'th order cepstral coefficient is related to the energy or gain of the frame of speech of which it was calculated. Noise and channel distortions can modify the gain in a signal significantly, while very little information about the speaker is conveyed by it. Therefore, the zero'th order cepstral coefficient is often discarded to improve speech processing system performance in adverse conditions [11, 43].

 Stellenbosch University http://scholar.sun.ac.za

Cepstral weighting (CW)

In cepstral weighting (CW), also known as liftering, each component of the cepstrum is weighted by a weighting factor. Cepstral weighting can be interpreted as a window function that is being multiplied by the cepstrum. The lower order cepstral coefficients are sensitive to spectral slope, while the higher order coefficients are more sensitive to noise [43, 44]. By defining proper windows, the overall effects of the weighted cepstral components can be emphasised or de-emphasised. The most commonly used window functions are:

- Rectangular window, which weight each component with the same value. This is the simplest window. We can consider all feature vectors as being weighted by an unity rectangular window.
- Ramp or quefrency liftering, which weight each component with its index [45]. This window function would be more of interest in a situation where noise is limited and channel distortion dominant (spectral slope).
- Band-pass liftering, which de-emphasise both low and high order cepstral coefficients by using a raised sinusoidal function [46]. This window would probably be more realistic in practical situations, where both channel distortions and noise are present.

Adaptive component weighting (ACW)

Adaptive component weighting [44, 47, 48] emphasise the formants of the LP derived spectrum by emphasising the narrow-band components and suppressing the wide-band components obtained by LP analysis.

To explain how this is done the all-pole model is written as the following transfer function

$$H(z) = \frac{1}{1 + \sum_{k=1}^{M} a_k z^{-k}} = \sum_{k=1}^{M} \frac{r_k}{1 - p_k z^{-1}}, \tag{1.1}$$

where M is the order of the LP analysis, a_k is the LP coefficient, r_k is the residue of the pole p_k. Each of these poles represent the centre frequency and the bandwidth of the k'th LP component. The narrow bandwidth components represent the formants, while the wider bandwidth components are more representative of the transmission channel and glottal characteristics.

 Stellenbosch University http://scholar.sun.ac.za

It is also known that the sensitivity of the poles to an error in the LP coefficients are proportional to the residues of the poles [49, p.336]. Therefore, the poles with large residues are more affected by distortions. It was also shown in [44] that the wide bandwidth components experience large overall variation with channel distortions. The narrow bandwidth components experience little variation in centre frequency and bandwidth, while the residues experience large variations. It was suggested in [44] that the effects of the residues in the LP spectrum be eliminated or reduced, and the contribution of the wide bandwidth components be suppressed. This is accomplished by normalising each component with its residue by setting all the residues equal to unity. This modification creates a spectrum where each component's peak value is inversely proportional to its bandwidth.

The cepstrum of the ACW spectrum is shown to be the conventional linear prediction coefficient cepstrum (LPCC) minus a cepstral component that can be determined using a simple recursion involving the LP coefficients. It can also be seen that ACW attempts to estimate the cepstrum of the channel and removes it from the LPCC to form the new feature. This channel estimation is adapted on a frame basis. Two other techniques which will be examined shortly, is the cepstral mean subtraction and post-filter cepstral mean subtraction. They also attempt to estimate the cepstrum of the transmission channel, but obtain their estimate by evaluating a number of frames. ACW outperform both these techniques and the unmodified LPCC [44, 48].

Cepstral post filtering (CPF)

The post filtered cepstrum is another feature that operates on the basis of specifying a spectrum that emphasise the spectral peaks (formants) and suppress the spectral valleys. Post filtering was introduced in [50] to enhance noisy speech. The transfer function of the post filter is given by

$$H_{pf}(z) = \frac{A(\frac{z}{\beta})}{A(\frac{z}{\alpha})}, \qquad \text{for } 0 < \beta < \alpha \leq 1, \tag{1.2}$$

where α and β are the liftering parameters. In the cepstral domain the post filter cepstrum simplifies to a weighting (or liftering) of the central components by a factor $\alpha^n - \beta^n$ (for $n > 0$), where n indicate the n'th cepstral component. Therefore, just as in ramp liftering, the lower order cepstral coefficients are de-emphasised. The post filtered cepstrum produce

 Stellenbosch University http://scholar.sun.ac.za

very favourable results in robust conditions [48].

1.3.2.3 Inter-frame processing of cepstral features

Cepstral derivatives (delta cepstrum)

A cepstrum represents the local spectral properties of a given frame of speech. However, it does not characterise the temporal or transitional information in a sequence of speech frames. Such temporal information can be obtained by differentiating the cepstrum. This result is the first order cepstral derivative, which is also known as the delta cepstrum. The delta cepstrum does not perform well on its own, but in conjunction with the cepstrum it produces better results than the cepstrum alone [51]. Since the additive noise and channel distortions are relatively constant over a short period of time, the temporal information introduced by the cepstral derivative is less susceptible to changes in adverse conditions [51, 43].

Cepstral mean subtraction (CMS)

The effect of a transmission channel can, as a first order approximation, be seen as a linear transformation of the speech signal. The frequency response of the channel is multiplied by the frequency response of the speech signal in the spectral domain. In the logarithmic domain, such as the cepstral domain, multiplications becomes additions. If we assume that the clean speech cepstral vector has zero mean, we can compensate for the channel by subtracting the cepstral average, obtained over a number of frames, from the cepstrum. This method is called cepstral mean subtraction (CMS) and it can improve the performance of recognition systems in adverse conditions [43, 52, 53]. It is not recommended to use this feature for clean speech signals. This is because of the assumption that clean speech has a zero cepstral mean, which is not actually true.

Pole-filtered cepstral mean subtraction (PFCMS)

Pole-filtered cepstral mean subtraction attempts to remove the residual effects of speech from the cepstral mean used for channel normalisation [54]. The eigenmodes of speech that are more

Stellenbosch University http://scholar.sun.ac.za

susceptible to convolutional deformations caused by transmission channels, are filtered. The LP poles with narrow bandwidth, that lie close to the unit circle, represents the formants. The formants are less sensitive to noise and channel deformations. On the other hand the LP poles with broad bandwidths model the spectral tilt, sub-glottal variation and channel effects [43, 54]. The formants contain little information about the channel, while the broad bandwidth poles contain more channel effects. So by de-emphasising the formants a better estimation of the channel is obtained. This is done by broadening the bandwidth of the formants (moving them away from the unit circle, while leaving the frequency unchanged). The LP cepstrum is then calculated. The mean of this new feature is used as an improved approximation of the additive effect of the convolution channel. The pole-filtered cepstral mean subtraction features are then obtained by subtracting the mean of the pole-filtered cepstrum from the unfiltered cepstrum. The pole-filtered cepstrum is only used to obtain a better approximation of the additive component contributed by the transmission channel. This feature has been shown to perform better than the standard CMS technique [43, 54].

Relative spectral (RASTA) processing

The rate at which non-linguistic components in speech change often lies outside the typical rate of change of the shape of the vocal tract [55]. RASTA takes advantage of this fact by suppressing the spectral components of the speech that change slower or faster than the typical range of speech changes [56, 55]. The underlying principle of RASTA processing is that the current frame of speech is considered in relation to some weighted average of past speech frames. This can be done by linearly filtering the time trajectories of each element of the parametric representation of the speech. The two issues faced in RASTA is to determine the domain in which filtering will be done and the filter to use. A band-pass filter that filters each frequency channel by a filter with sharp spectral zeros at zero frequency and two higher frequencies are used. The low and high cut-off frequencies determine the band of spectral changes that will be preserved. Two domains in which filtering are performed, were introduced. The first is the log spectral domain. Distortions caused by telecommunication channels and convolutional noise are additive in the log spectral domain. Since these distortions are additive in the log domain it makes sense to do the filtering in this domain. The RASTA processing combined with PLP feature vectors

 Stellenbosch University http://scholar.sun.ac.za

show excellent performance improvement, when the speech is contaminated by convolutional noise and channel distortions, over standard features [55, 56]. However, for additive noise, the performance is not as good if filtering is performed in the log domain. The effects of additive noise are also additive in the power spectrum domain, but signal-dependent in the log domain. The second domain specified in [55] is the lin-log domain, which is linear for small spectral values and logarithmic for larger values. The lin-log domain is more suitable for removing some of the effects of additive noise. The transformation function is given by

$$y = ln(1 + Jx), \tag{1.3}$$

where J is a constant. The problem with this method is that there is an optimal value of J for each signal to noise ratio. Consequently, the problem of determining which J to use in a system is posed. Adaptive ways of modifying J have been suggested in [55].

1.3.3 Models and classifiers

Probably the most popular model used in current speaker recognition systems are GMMs. The application of GMMs in speaker recognition was introduced by Reynolds and Rose [11, 12]. Reynolds also published large population speaker identification experiments on clean and telephone speech [3]. In his experiments he used the TIMIT (clean high quality speech) and NTIMIT (telephone transmitted version of the TIMIT database) databases. For the clean speech the performance results barely dropped as the population size increased to 630 speakers, with an end result of 99.5%. This indicates that GMMs are very well suited for modelling speakers, even for large speakers sets. However, on the NTIMIT database the identification results gradually dropped to 60.7% as the population size increased. This experiment clearly demonstrates how not even the best models are immune against the effects of noise and channel deformations.

Before GMMs were introduced, VQ models were very popular. Examples of its application in speaker recognition can be found in [9, 57, 58, 59]. Other systems that were used for speaker recognition were covariance only models [60, 6], autoregressive models [61, 62], neural networks (NN) [63, 64] (such as the multi layer perceptron (MLP) [65], radial basis function (RBF) networks [66], learning vector quantisation [67] and time delay neural networks (TDNN) [68]),

Stellenbosch University http://scholar.sun.ac.za

as well as hidden Markov models. [69, 70, 71, 72, 73]. However, since GMMs outperform these systems [11], they will not be discussed in more detail. All the speaker identification experiments conducted in this study have employed GMMs.

No studies were found that compared the various speaker models against each other to determine which is the most robust. In most cases it would seem that if one model performs better than another under ideal conditions, then it will also perform better under noisy conditions. This should not be taken as a fact without proper verification. In general, attempts are not made to increase robustness on the model level, so we will not spend any more time on this part of the recognition system.

1.4 Objectives

This study had the following objectives in mind:

- Investigate the success of current speaker recognition technology when applied to speech transmitted over HF channels.
- Investigate existing robust speech processing methods and determine how effective they are when applied to the HF transmitted speech data.
- Determine the effect that the various HF channel deformations have on the LP coefficients and the LP cepstrum.
- Determine the effects that single sideband modulation (SSB) has on the LP coefficients and the LP cepstrum.
- Investigate time domain methods to combat channel noise, in order to increase speaker recognition performance.

Stellenbosch University http://scholar.sun.ac.za

1.5 Contributions

The following contributions to the fields of interest were made:

- Identification of the deformation effects on LP coefficients and the LP cepstrum, when speech is transmitted over HF channels that uses SSB modulation.
  - Unless a large frequency error in modulation and demodulation is present, SSB modulation has only a small effect on the LP coefficients and the LP cepstrum.
  - The HF channel may have a large influence in the deformation of the LP coefficients and the LP cepstrum if the amplitude of the original received signal and that of any multi-path signals is of the same order, or if large Doppler shifts are present.
  - One of the primary contributors to the deformation of the LP coefficients and the LP cepstrum is noise.
- Development of a new statistical based speech enhancement method using HMMs.
  - Using first and higher order HMMs and speech samples to model the speech process.
  - HMM mixture expansion algorithm.

1.6 Outline of the thesis

In the second chapter we describe the speech analysis process in detail, from the analogue signal to the calculation of LP cepstral features.

The third chapter presents a speaker recognition system based on Gaussian mixture models. The creation and application of models are described.

Chapter four introduces a number of robust feature analysis techniques. Two new features are introduced and a number of inter- and intra-frame processing of the cepstral features are presented.

Stellenbosch University http://scholar.sun.ac.za

In chapter five the results of our speaker recognition experiments are presented.

In chapter six the HF channel and SSB modulation are described. The effects that the channel and modulation system have on the LP coefficients and LP cepstrum is derived.

Chapter seven describes our new speech enhancement method based on hidden Markov models.

In chapter eight the results of our speech enhancement experiments are presented.

Chapter nine finishes off with a summary of the project and suggestions for improvements and further research possibilities.

Stellenbosch University http://scholar.sun.ac.za

Chapter 2

Speech signal analysis

The process of speech analysis involves converting the recorded analogue speech signal into feature vectors that can be applied in any speech processing application of the user's choice. This generally involves three steps:

- Analogue-to-digital conversion.
- Preprocessing.
- Feature calculation.

2.1 Analogue-to-digital (ADC) conversion

A digital computer can not operate on the pure acoustic waves (as transmitted through air or other mediums) or its electronic analogue form. The analogue speech signal must first be converted to a digital speech signal that can be discretely manipulated by a digital computer. This is done by an analogue to digital converter (ADC) (see [74, pp.612-641] and [75, pp.21-39] for more details), which is usually a specialised component inside either a digital signal processing (DSP) card or a sound card in a digital computer. Since noise inflict such a huge penalty on recognition systems, the quality of the ADC can contribute significantly to the success of the

Stellenbosch University http://scholar.sun.ac.za

recognition system.

ADC can be subdivided into three steps:

- Sampling.
- Quantisation.
- Coding.

Sampling is the process of taking a sample from the continuous analogue speech signal at regular intervals, and which is denoted the sampling interval. According to the sampling theorem, the analogue signal should be sampled at more than twice the highest operating frequency to avoid aliasing. Depending on the operating environment, a tradeoff between the sampling frequency and the quantity of data can be made. CD quality audio data is sampled at 44.1 kHz, high quality speech usually between 16 and 20 kHz, while low quality telephone transmitted speech is sampled at around 8 kHz (because of the low bandwidth that telephone channels offer).

Quantisation is the process of converting the sampled continuous analogue speech signal into a digital signal by expressing each sample as a finite number of digits. An error is introduced by representing the continuous signal by a finite number of binary values. This error is known as the quantisation error or the quantisation noise. In a digital computer we can represent the quantisation levels in powers of two (2^b, where b is known as the bits per sample of the ADC). For example: an eight bit converter will be able to represent 256 values, while an 16 bit converter can represent 65536 values. The signal-to-quantisation noise ratio (SQNR) is the highest possible SNR of the system and is given by [75, p.37]

$$SQNR = \frac{3}{2} \cdot 2^{2b}, \tag{2.1}$$

or in dB as

$$SQNR_{dB} = 10 \log_{10} SQNR = 1.76 + 6.02b. \tag{2.2}$$

This implies that for each added bit you roughly gain 6 dB SNR.

Stellenbosch University http://scholar.sun.ac.za

Coding is the process of assigning an unique binary number to each quantisation level. Examples of coding techniques are the uniform quantiser and pulse code modulation (see chapter 7 in [7, p.409-500] for a comprehensive covering of this topic).

2.2 Preprocessing

The preprocessing stage is where the raw digital signal is further processed in order to improve the performance of the recognition system, or to prepare the speech for the feature calculation stage. A number of preprocessing methods follow:

2.2.1 Speech enhancement

In practical applications the environment in which the speech is recorded can be contaminated by background noise or interfering speakers. The transmission channel also deforms the speech. The system implementer can either choose to ignore these effects at this stage and combat them on the feature level, or employ some kind of speech enhancement system. A short introduction to a few speech enhancement systems were provided in Section 1.2.1 and Section 1.3.1.

In a practical system it would be to the advantage of the implementer to employ some kind of speech enhancement system.

2.2.2 Speech framing

In practical applications it is more convenient to work with short segments (frames) of the speech signal. Conventional analysis techniques typically require the signal to be at least wide sense stationary. It is of course not true in long continuous speech signals, but it is assumed to be true for short portions of the speech signal. The choice of the frame length should be made so that this assumption is reasonable. This requirement is in contrast with that of accurate spectral estimation that benefits from longer sequences. A longer frame length will produce a

Stellenbosch University http://scholar.sun.ac.za

better spectral picture, while a shorter frame length will resolve events in the signal better in time. This trade-off is known as the spectral-temporal resolution trade-off [7, p.20].

Typical frame lengths vary between 15 ms and 40 ms in duration. The frames are usually overlapped by 50% to produce a smooth transition between frames.

2.2.3 Windowing

A window function, $w(n)$, is a real, finite length sequence (the length of the operating frame) that is multiplied by the signal (or frame) [7, p.16].

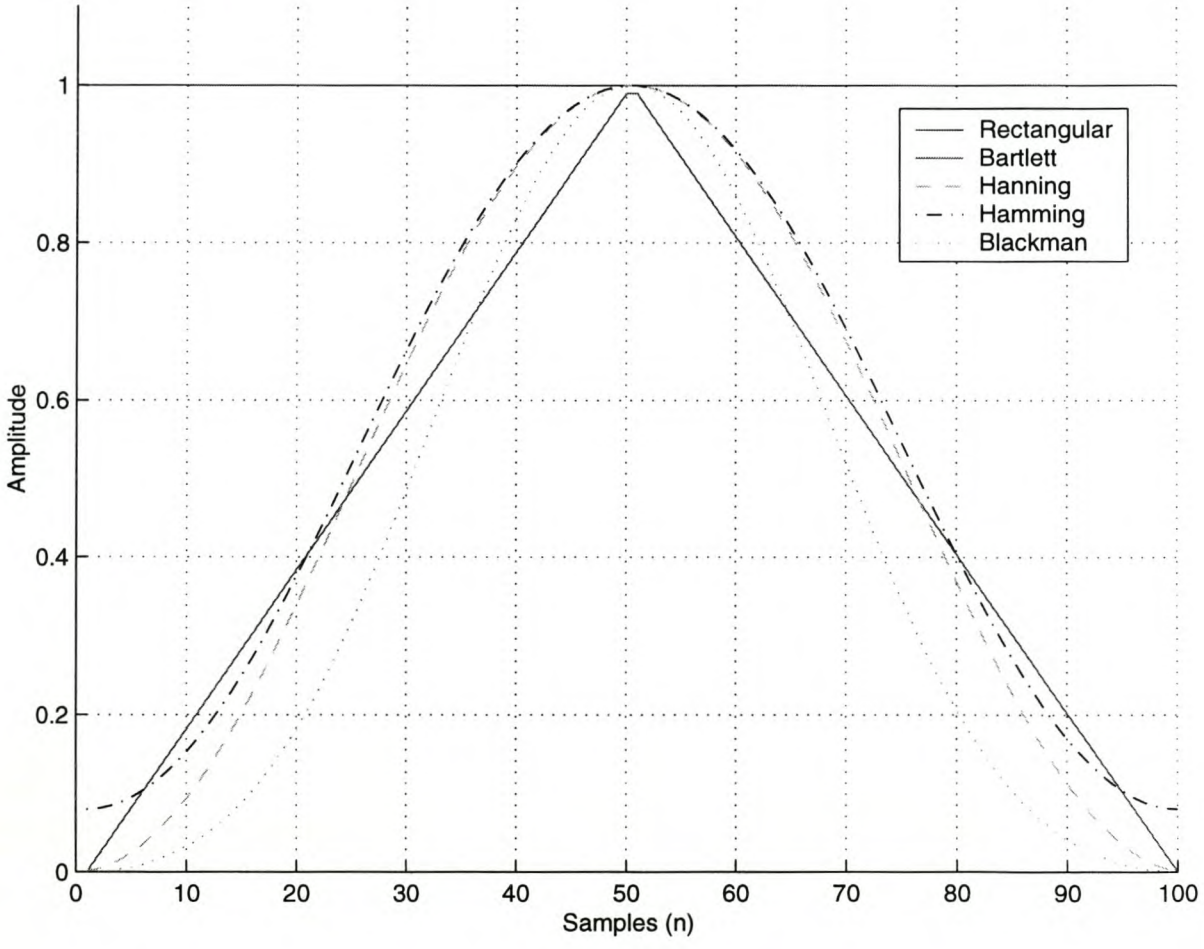

Figure 2.1: *Time domain response of some common window functions.*

A few commonly used windows are shown in Figure 2.1. These windows are defined for a sequence $w(n)$, with length N, as follow:

Stellenbosch University http://scholar.sun.ac.za

Rectangular:

$$w(n) = \begin{cases} 1, & 0 \leq n \leq N \\ 0, & \text{otherwise} \end{cases}. \tag{2.3}$$

Bartlett (triangular):

$$w(n) = \begin{cases} \frac{2n}{N}, & 0 \leq n \leq \frac{N}{2} \\ 2 - \frac{2n}{N}, & \frac{N}{2} \leq n \leq N \\ 0, & \text{otherwise} \end{cases}. \tag{2.4}$$

Hanning:

$$w(n) = \begin{cases} 0.5 - 0.5\cos(\frac{2\pi n}{N}), & 0 \leq n \leq N \\ 0, & \text{otherwise} \end{cases}. \tag{2.5}$$

Hamming:

$$w(n) = \begin{cases} 0.54 - 0.46\cos(\frac{2\pi n}{N}), & 0 \leq n \leq N \\ 0, & \text{otherwise} \end{cases}. \tag{2.6}$$

Blackman:

$$w(n) = \begin{cases} 0.42 - 0.5\cos(\frac{2\pi n}{N}) + 0.08\cos(\frac{4\pi n}{M}), & 0 \leq n \leq N \\ 0, & \text{otherwise} \end{cases}. \tag{2.7}$$

The frequency response of these windows are shown in Figure 2.2. The first point to note is that the rectangular window function has very high side lobes compared to that of the other windows. Secondly, for the rectangular window function the first main lobe is narrower than that of the other windows. A narrower main lobe will be able to resolve sharper details in the speech spectrum. Smaller sidelobes mean that fewer components from the rest of the spectrum will leak into the current frequency component. The attenuation of the first sidelobes is very important to obtain an accurate frequency response. The cost of this is the loss of resolution for frequency components close to each other.

Stellenbosch University http://scholar.sun.ac.za

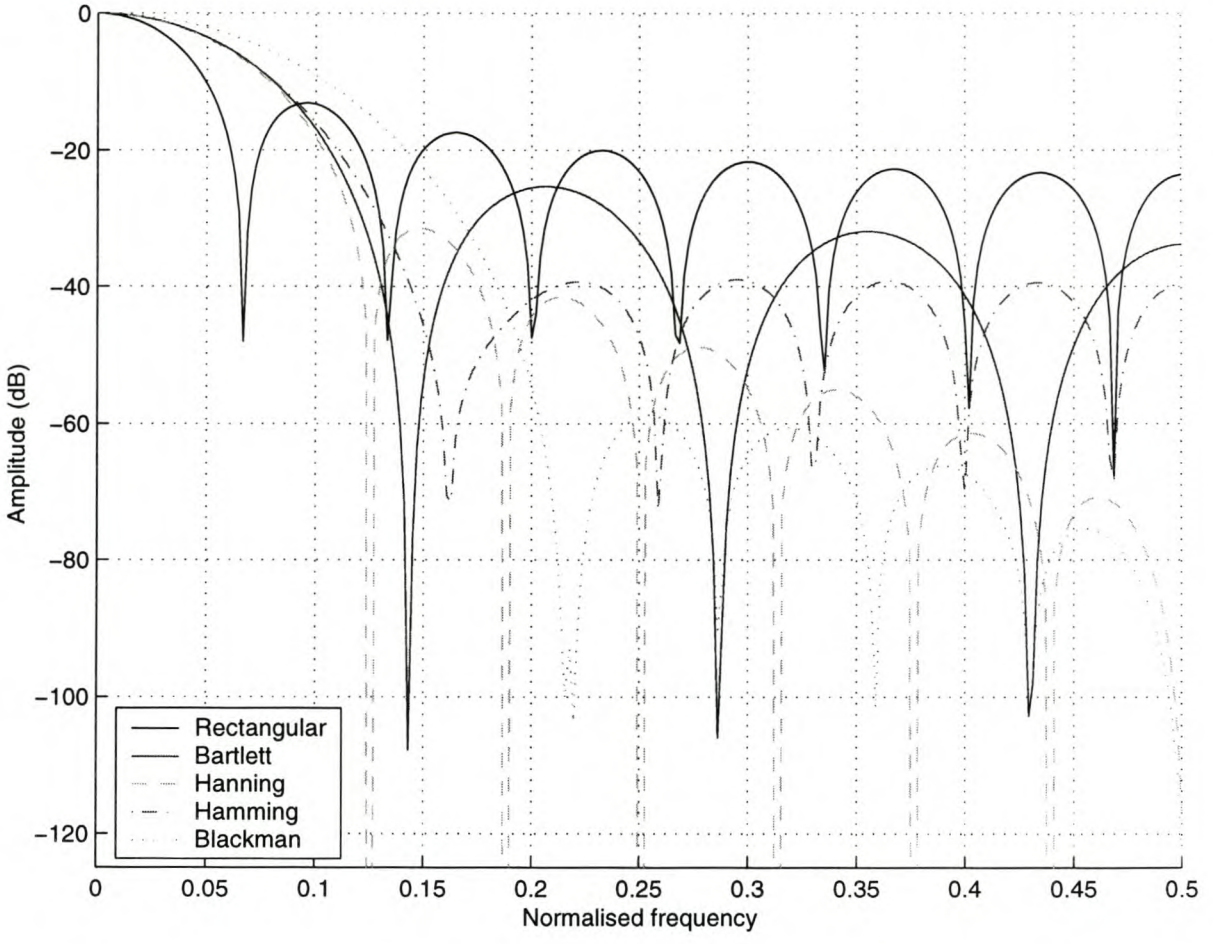

Figure 2.2: *Frequency domain response of some common window functions.*

Even if a window is not explicitly applied to the frame, a rectangular window is implicitly applied. This is due to the finite length of a frame. The smoother windows are usually used because of their preferable side-lobe characteristics. A popular choice for speech processing applications is the Hamming window.

It is stated as a general rule for LP analysis, that a window should be used if the frame length is larger than 15 ms or consists of several pitch periods [36, p.158].

2.2.4 Pre-emphasis

A pre-emphasis filter is frequently applied to the speech signal before LP analysis.

Stellenbosch University http://scholar.sun.ac.za

It is typically a simple first order high pass filter that increases the relative energy of the higher frequency spectrum [7]. The transfer function of such a filter is

$$H(z) = 1 - \alpha z^{-1}, \tag{2.8}$$

where α is a constant and $0.9 \leq \alpha \leq 1.0$. The frequency response and phase of a pre-emphasis filter with $\alpha = 0.97$ is shown in Figure 2.3.

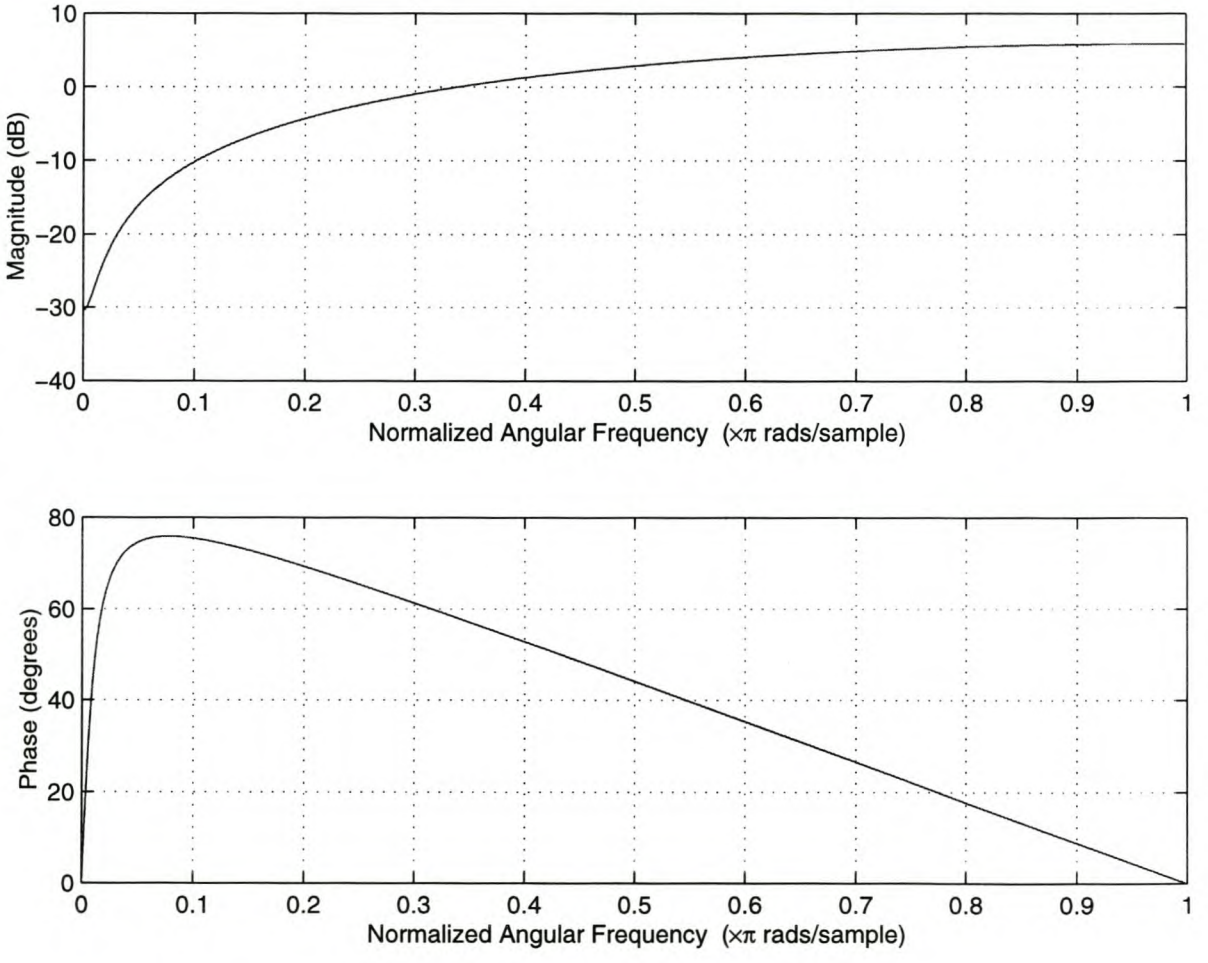

Figure 2.3: *The frequency and phase response of a pre-emphasis filter with* $\alpha = 0.97$.

There are several reasons for using a pre-emphasis filter. The first of these is that it is usually more desirable to only include the characteristics of the vocal tract for LP analysis. By applying a pre-emphasis filter the glottal waveform and lip radiation characteristics are eliminated [36, pp.158].

The second reason is to prevent numerical instability [7, p.330]. When calculating the high order LP coefficients, an ill conditioned autocorrelation matrix can result if the speech signal is

Stellenbosch University http://scholar.sun.ac.za

dominated by low frequency components. Furthermore, the problem increases if the dynamic range of the spectrum increases. If the spectrum has a tilt that causes the wide dynamic range, then a first order inverse filter should be able to whiten the spectrum. The pre-emphasis filter is one such a filter. An optimum value for α that will maximise the spectral flatness of the output speech (and thus minimising the effect of ill-conditioning) is given by [36, p.216]

$$\alpha = \frac{r(1)}{r(0)}, \tag{2.9}$$

where $r(1)$ and $r(0)$ are autocorrelation coefficients.

Finally, pre-emphasis also alleviates the effect of recording devices and the transmission of speech through air. Some recording devices attenuate the higher frequencies more than the lower ones and higher frequencies gets attenuated while propagating through air.

It is also noted that it doesn't really matter if pre-emphasis is done before or after windowing [36, p.158].

2.3 Speech features

From each frame a vector of speech features are produced. Speech information is usually conveyed in the spectrum of the speech. The logical choice for speech features is one that represents the spectrum of the speech in a compact way. It is clear that a feature contains information about both the speech and the speaker. Over short periods the feature represents specific sounds and some information about the speaker as well. Over longer periods, many sounds are uttered and the accumulation of the features gradually fills the feature space in a unique way for each speaker.

The most popular speech feature currently in use for speech and speaker recognition is the cepstrum and its derivatives. In the next few sections more detailed descriptions of linear prediction (LP) analysis (as a prerequisite for the LP cepstrum), LP cepstrum and Mel-warped cepstrum is presented.

Stellenbosch University http://scholar.sun.ac.za

2.3.1 Linear prediction (LP) analysis

In speech analysis the spectral content of a frame of speech is often modelled by an AR process (an all pole filter). This approximation of the speech process is explained in detail in [36]. An AR filter, excited by a white noise sequence $v(n)$, produces the random process $x(n)$. In the time domain this is presented as

$$\sum_{k=0}^{M} a_k^* x(n-k) = v(n) \quad k = 0, \ldots, M, \tag{2.10}$$

and in the frequency domain

$$\begin{aligned} H(z) =& \frac{1}{A(z)} \\ =& \frac{1}{\sum_{k=0}^{M} a_k^* z^{-k}}, \end{aligned} \tag{2.11}$$

where M is the order of the AR process and a_k is the AR parameters. In order to obtain the AR parameters, the Yule-Walker [76, p.118] equation is solved

$$\begin{bmatrix} r_x(0) & r_x(1) & \cdots & r_x(M-1) \\ r^*(1) & r_x(0) & \cdots & r_x(M-2) \\ \vdots & \vdots & \ddots & \vdots \\ r^*(M-1) & r^*(M-2) & \cdots & r_x(0) \end{bmatrix} \begin{bmatrix} w_1 \\ w_2 \\ \vdots \\ w_M \end{bmatrix} = \begin{bmatrix} r^*(1) \\ r^*(2) \\ \vdots \\ r^*(M) \end{bmatrix} \tag{2.12}$$

where $r_x(l)$ is the autocorrelation of the input vector and $w_k = -a_k$. This can be written in a more compact matrix form as

$$\mathbf{R_x w} = \mathbf{r_x}. \tag{2.13}$$

This is the same equation as the Wiener-Hopf equation [76, p.203],

$$\mathbf{R_x w}_f = \mathbf{r_x}, \tag{2.14}$$

that needs to be solved in the forward linear prediction problem. This is why linear prediction is often used to estimate the spectrum of speech. The linear prediction coefficients are the poles of the AR process. The development of forward linear prediction, or just linear prediction, is discussed next.

Stellenbosch University http://scholar.sun.ac.za

In linear prediction the next sample in a sequence is predicted from a linear combination of the past M samples, where M is the order of the predictor. A forward linear predictor consists of a linear transversal filter with M tap weights, $w_1, \ldots, w_M$, and tap inputs $x(n-1), \ldots, x(n-M)$. We also assume that the tap weights (prediction coefficients) are from a wide-sense stationary stochastic process and the process has a zero mean. The linear predicted value of $x(n)$ is

$$\widehat{x}(n) = \sum_{k=1}^{M} w_k x(n-k) \tag{2.15}$$

and the forward prediction error is

$$f_M(n) = x(n) - \widehat{x}(n). \tag{2.16}$$

This leads to the forward prediction-error filter. The expected minimum mean squared prediction error is given by

$$P_M = E[|f_M(n)|^2], \tag{2.17}$$

where $E[\cdot]$ is the expectation of a function. Given that the tap inputs have zero mean, the forward prediction error, f_M, will also have a zero mean. This means that the minimum squared prediction error, P_M, is equal to the variance of f_M (P_M can also be seen as the power of f_M).

In order to solve the problem we solve the Wiener-Hopf equation (2.14). We therefore need to determine the correlation matrix of the input tap vectors and the cross-correlation vector between the tap inputs and the desired response. Given the input we define a vector

$$\mathbf{x}(n-1) = [x(n-1), \ldots, x(n-M)]^T. \tag{2.18}$$

The correlation matrix of the input tap vector is then

$$\begin{aligned} \mathbf{R}_\mathbf{x} &= E[\mathbf{x}(n-1)\mathbf{x}^H(n-1)] \\ &= \begin{bmatrix} r_x(0) & r_x(1) & \ldots & r_x(M-1) \\ r^*(1) & r_x(0) & \ldots & r_x(M-2) \\ \vdots & \vdots & \ddots & \vdots \\ r^*(M-1) & r^*(M-2) & \ldots & r_x(0) \end{bmatrix}, \end{aligned} \tag{2.19}$$

Stellenbosch University http://scholar.sun.ac.za

where $r_x(l)$ is the autocorrelation function for the l'th lag of the input process. The cross-correlation vector between the tap inputs and the desired response is

$$\begin{aligned}\mathbf{r_x} &= E[\mathbf{x}(n-1)x^*(n)] \\ &= \begin{bmatrix} r^*(1) \\ r^*(2) \\ \vdots \\ r^*(M) \end{bmatrix} = \begin{bmatrix} r_x(-1) \\ r_x(-2) \\ \vdots \\ r_x(-M) \end{bmatrix}.\end{aligned} \tag{2.20}$$

Finally with the tap weights given as

$$\mathbf{w}_f = [w_1, \ldots, w_M]^T. \tag{2.21}$$

We can write equation (2.14) as

$$\begin{bmatrix} r_x(0) & r_x(1) & \ldots & r_x(M-1) \\ r^*(1) & r_x(0) & \ldots & r_x(M-2) \\ \vdots & \vdots & \ddots & \vdots \\ r^*(M-1) & r^*(M-2) & \ldots & r_x(0) \end{bmatrix} \begin{bmatrix} w_1 \\ w_2 \\ \vdots \\ w_M \end{bmatrix} = \begin{bmatrix} r_x(-1) \\ r_x(-2) \\ \vdots \\ r_x(-M) \end{bmatrix}. \tag{2.22}$$

Furthermore, noting that the variance of $x(n)$ equals $r_x(0)$, the prediction error power is

$$P_M = r_x(0) - \mathbf{r_x}^H \mathbf{w}_f. \tag{2.23}$$

If we combine equation (2.14) and equation (2.23) we get the augmented Wiener-Hopf equations for forward linear prediction

$$\begin{bmatrix} r_x(0) & \mathbf{r_x}^H \\ \mathbf{r_x} & \mathbf{R_x} \end{bmatrix} \begin{bmatrix} 1 \\ -\mathbf{w}_f \end{bmatrix} = \begin{bmatrix} P_M \\ \mathbf{0} \end{bmatrix}. \tag{2.24}$$

This relation can also be expressed as a system of (M+1) simultaneous equations

$$\sum_{k=0}^{M} a_{M,k} r_x(k-i) = \begin{cases} P_M, & i = 0 \\ 0, & i = 1, 2, \ldots, M \end{cases}, \tag{2.25}$$

where

$$\mathbf{a}_M = \begin{bmatrix} 1 \\ -\mathbf{w}_f \end{bmatrix}. \tag{2.26}$$

Using equation (2.24) or equation (2.25) algorithms can be developed to calculate the linear prediction coefficients. One of the most commonly used algorithms is the Levinson-Durbin algorithm [76, p.254] which will not be explained here.

Stellenbosch University http://scholar.sun.ac.za

2.3.2 Cepstral features

It has been suggested that cepstral features produce superior performance for speaker recognition [40, 10]. The cepstrum is defined in the next section and two of the most popular variations, the LP cepstrum and Mel-warped cepstrum, follow.

2.3.2.1 Definition

Given a sequence $x(n)$ with a z-transform of $X(z)$, the complex cepstrum is defined as the inverse z-transform of the natural logarithm of the z-transform of $x(n)$ or

$$\begin{aligned} c_x(n) &= Z^{-1}\{ln\ Z\{x(n)\}\} \\ &= Z^{-1}\{C_x(z)\}. \end{aligned} \tag{2.27}$$

If $C_x(z)$ converge it can be represented by a Laurent Series

$$C_x(z) = ln\ X(z) = \sum_{n=-\infty}^{\infty} c_x(n) z^{-n}, \tag{2.28}$$

where

$$c_x(n) = \frac{1}{2\pi j} \oint_C ln\ X(z) z^{n-1} dz \tag{2.29}$$

and C is a closed circle around the origin within the region of convergence. Equation (2.28) and (2.29) are obtained using the definitions of the z-transform and the inverse z-transform respectively. If the complex cepstrum exists, $C_x(z)$ converge at the unit circle and then

$$C_x(w) = ln\ X(w) = \sum_{n=-\infty}^{\infty} c_x(n) e^{-j\omega n}, \tag{2.30}$$

where $c_x(n)$ is obtained from the inverse FT of $ln\ X(w)$:

$$c_x(n) = \frac{1}{2\pi} \int_{-\pi}^{\pi} ln\ X(w) e^{j\omega n} d\omega. \tag{2.31}$$

By expressing $X(w)$ in terms of magnitude and phase ($X(w) = |X(w)|e^{j\theta\omega}$), we can separate the complex cepstrum into a magnitude and a phase component

$$\begin{aligned} c_m(n) &= \frac{1}{2\pi} \int_{-\pi}^{\pi} ln\ |X(w)| e^{j\omega n} d\omega. \\ c_\theta(n) &= \frac{1}{2\pi} \int_{-\pi}^{\pi} \theta(\omega) e^{j\omega n} d\omega. \end{aligned} \tag{2.32}$$

Stellenbosch University http://scholar.sun.ac.za

In speech processing only the magnitude component, $c_m(n)$, is calculated. The real cepstrum is used to separate/estimate the spectral content of the of the speech from its pitch frequencies.

2.3.2.2 LP cepstrum

The LP cepstrum can be calculated recursively from the LP coefficients using the following equation

$$c_x(m) = \begin{cases} ln(\sqrt{P_M}), & m = 0 \\ -a(m) - \sum_{k=1}^{m-1} \left(\frac{k}{m}\right) c_x(k)a(m-1), & m > 0 \end{cases}. \tag{2.33}$$

In this study we use the LP cepstrum in all experiments (see Chapter 5).

2.3.2.3 Mel-warped cepstrum

The mel-warped cepstrum [7, 6] is based on the non-linear nature of perception of frequencies by a human listener. The Mel is a unit of measure of perceived pitch or frequency of a tone and is approximated by

$$F_{mel} = \frac{1000}{\log 2}\left(1 + \frac{F_{Hz}}{1000}\right). \tag{2.34}$$

The mel-warped cepstrum is calculated by first transforming the speech from the time domain to the log frequency domain, by taking the logarithm of the FFT of the speech. The second step is to warp the frequency using the mel-scale. The third step is to extract a number of components from fixed positions in the frequency scale. There is usually a number of linearly spaced components (in the 0-1 kHz range) followed by number of log spaced components (above 1 kHz). The process can be terminated at this stage to produce a feature. In the next step the total log energy in a critical band around the chosen frequencies can be calculated and used as a feature, rather than just the frequency component itself. This is based on the observation that the perception of a frequency component is influenced by the energy in a critical band around the frequency component.

Stellenbosch University http://scholar.sun.ac.za

2.4 Summary

In this chapter speech analysis techniques were introduced. An analogue speech signal is converted to a digital signal using an ADC. The number of quantisation bits of the ADC determines the SQNR. This is also the maximum SNR that the system is capable of obtaining. A number of preprocessing techniques are usually applied before using the speech in recognition systems. This includes segmenting the speech into smaller frames, applying a window function, applying a pre-emphasis filter and, as an optional step, performing speech enhancement on the speech to reduce noise in the signal. The speech features are then calculated. We discussed both the LP cepstrum and the Mel-warped cepstrum. In this study only the LP cepstrum features were used.

Stellenbosch University http://scholar.sun.ac.za

Chapter 3

Speaker recognition

The purpose of a speaker recognition (SR) system is to recognise a speaker by his or her voice. The aim is to identify the differences in vocal tract, nasal cavity and the vocal cord characteristics between different speakers.

A speaker recognition system can either be text-dependent or text-independent. A text-dependent system requires that the training and testing text be the same and known. The known text can be exploited to increase performance and can be useful in security applications. A text-independent system places no constraints on the text being used for training and testing, and the text can be different in both cases. The advantage of such a system is that a speaker can be identified from any speech utterance if the system was trained to recognise the speaker.

A speaker recognition system can not identify a speaker unless the system is trained with speech produced by that speaker, just as a human can not identify a person unless he already knows that person from previous encounters. The first task of any speaker recognition system is to collect data from all the speakers to be identified and train models to represent each speaker. The second task of a speaker recognition system is to process a segment of speech from an unidentified speaker and attempt to determine his or her identity from the database of speakers known to the system. This problem is also known as closed-set speaker recognition. Open-set speaker recognition is concerned with determining whether the speech under test belongs to one of the speakers in the known speaker set or not. A special case of this problem is known as

Stellenbosch University http://scholar.sun.ac.za

speaker verification. In speaker verification the goal is to determine whether the speaker is who he claims to be. In such a system, the speaker under investigation can be one of the known set of speakers who claims to be someone else or the speaker could be a person totally unknown to the system. Speaker identification and speaker verification will be covered in more detail in Section 3.1.4.

The biggest consideration in constructing a speaker recognition system is the choice of model to represent the speaker. The current state-of-the-art model is the GMM [11, 12]. This is also the model used in this study. This chapter is devoted to providing a detailed description of the GMM and its application to speaker recognition. For references to other models that have been investigated for speaker recognition see Section 1.3.3.

3.1 Speaker modelling using Gaussian mixture models

There are two reasons for using a GMM to model speakers. Firstly, the individual Gaussian components represent some broad acoustic classes that can overlap. The importance of this aspect is that it is assumed that a speaker fills the feature space in subclasses that can each represent some general speaker-dependent characteristic, such as vocal tract configurations for different phonetic events. Secondly, a Gaussian mixture density is shown to provide a smooth approximation of the underlying long-term sample distribution of observations obtained from utterances by a given speaker. This gives the GMM the ability to approximate arbitrarily-shaped density functions.

3.1.1 Description of the Gaussian mixture model

A Gaussian mixture PDF is the weighted sum of M component densities and is given by the equation

$$f(\mathbf{x}|\lambda) = \sum_{i=1}^{M} m_i b_i(\mathbf{x}), \tag{3.1}$$

Stellenbosch University http://scholar.sun.ac.za

where $\mathbf{x}$ is a D-dimensional random vector, $b_i(\mathbf{x})$, $i = 1, \ldots, M$, are the component densities and m_i, $i, \ldots, M$, are the mixture weights. Each component density is a D-variate Gaussian function of the form

$$b_i(\mathbf{x}) = \frac{1}{(2\pi)^{D/2}|\boldsymbol{\Sigma_i}|^{1/2}} e^{\{-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu_i})^T \boldsymbol{\Sigma_i}^{-1}(\mathbf{x}-\boldsymbol{\mu_i})\}}, \tag{3.2}$$

where $\boldsymbol{\mu_i}$ is the mean vector and $\boldsymbol{\Sigma_i}$ is the covariance matrix of the i'th Gaussian PDF. The mixture weights further satisfy the constraint that $\sum_{i=1}^{M} m_i = 1$.

The complete Gaussian mixture PDF is represented by the mean vector, covariance matrices and mixture weights of all the component densities. These parameters are collectively represented by the notation

$$\lambda = \{m_i, \boldsymbol{\mu_i}, \boldsymbol{\Sigma_i}\}, \qquad \text{with } i = 1, \ldots, M, \tag{3.3}$$

where λ is the GMM for each speaker.

It is possible to reduce the computational load of the classification process by simplifying the full covariance matrix of a speaker model to a diagonal covariance matrix, thus assuming that the covariance between the dimensions is zero. Since the underlying Gaussian components act together to model the overall PDF, a full covariance matrix is not necessary. A set of full covariance Gaussians can be approximated by a larger set of diagonal covariance Gaussians, since a linear combination of diagonal covariance Gaussians are capable of modelling the correlation between feature vector elements [11].

3.1.2 Model training

The models are trained by estimating the parameters of the GMM from speech collected from each participating speaker. Maximum likelihood (ML) estimation is used to estimate the parameters. The aim of ML estimation is to find the parameters that maximise the likelihood of the GMM. Given a sequence of T training vectors, $X = \{\mathbf{x_1}, \ldots, \mathbf{x_T}\}$, the GMM likelihood is given by

$$f(X|\lambda) = \prod_{t=1}^{T} f(\mathbf{x_t}|\lambda). \tag{3.4}$$

Stellenbosch University http://scholar.sun.ac.za

Unfortunately this is a non-linear function of λ, due to the fact that the data that is needed to update the PDF parameters is only indirectly available through the observed data, X. Therefore direct estimation is not possible. Instead the parameters can be obtained iteratively by using the expectation maximisation (EM) algorithm. See Appendix C for a more detailed description of the EM algorithm. The EM algorithm starts with an initial parameter set of the model and then calculates a new set of parameters so that the likelihood that the training data were generated by the current estimate of the model, defined by the new parameter set, is higher than that of the previous estimate. The newly estimated model is then used as the starting point for the next iteration. This process is repeated until some convergence threshold is reached.

The following formulas are used to re-estimate the model parameters iteratively using the EM algorithm. Using these formulas it is guaranteed that the likelihood of the models increases monotonically with each iteration.

Mixture Weights:

$$\overline{m_i} = \frac{1}{T}\sum_{t=1}^{T} f(i|\mathbf{x_t}, \lambda). \tag{3.5}$$

Means:

$$\overline{\boldsymbol{\mu_i}} = \frac{\sum_{t=1}^{T} f(i|\mathbf{x_t}, \lambda)\mathbf{x_t}}{\sum_{t=1}^{T} f(i|\mathbf{x_t}, \lambda)}. \tag{3.6}$$

Variances:

$$\overline{\boldsymbol{\sigma_i}}^2 = \frac{\sum_{t=1}^{T} f(i|\mathbf{x_t}, \lambda)x_t^2}{\sum_{t=1}^{T} f(i|\mathbf{x_t}, \lambda)} - \overline{\boldsymbol{\mu_i}}^2, \tag{3.7}$$

where $\overline{m_i}$ $\overline{\boldsymbol{\sigma_i}}^2$, $\overline{\boldsymbol{\mu_i}}$ is the new estimate values of m_i, $\boldsymbol{\sigma_i}^2$, $\boldsymbol{\mu_i}$ respectively, and σ_i^2, x_t, μ_i refer to arbitrary element of the vectors $\boldsymbol{\sigma_i}^2$, $\mathbf{x_t}$, $\boldsymbol{\mu_i}$ respectively. The posterior probability for the i'th class is given by

$$f(i|\mathbf{x_t}, \lambda) = \frac{m_i b_i(\mathbf{x_t})}{\sum_{k=1}^{M} m_k b_k(\mathbf{x_t})}. \tag{3.8}$$

3.1.3 Practical considerations

The following practical considerations should be considered when training a GMM:

Stellenbosch University http://scholar.sun.ac.za

3.1.3.1 Initialisation

It is known that in the estimation of the model parameters we can encounter a local maximum. We therefore need to choose the initial conditions with care. Usually a set of vector quantisation (VQ) codebook vectors are used as the initial values for the model means. In this study, a non-uniform binary split VQ was used to initialise the Gaussian centroids. The resulting codebook vectors were used as initial values for a full K-means VQ step.

A description of the K-means and non-uniform binary split VQ algorithms is given in Appendix B.

3.1.3.2 Model Order

It is important to choose a model order, M, that can sufficiently represent a speaker, while keeping the model size as small as possible for computational efficiency. There is no theoretical method to choose the order beforehand. It is left to the implementer to determine the optimum size for both performance and speed. Frequently encountered model orders are between 16 and 32 Gaussian components. It was also found that increasing the order beyond 32 can result in performance loss [11]. Another consideration is that during training the weights of one or more components become very small compared to the weights of the other components. In such cases we choose to discard these components, thus reducing the order of the model, which results in a performance increase.

3.1.3.3 Variance limiting

Due to limited training data it can happen that the variance of some Gaussian components can become very small. This can cause numerical problems when the model is used. To prevent this, we limit the variance to a small positive 'floor' value. If the variance is below a predefined limit σ^2_{min}, the variance is set to be σ^2_{min}.

Stellenbosch University http://scholar.sun.ac.za

3.1.4 Classification

The speaker identification and verification classifiers are covered separately.

3.1.4.1 Speaker identification

As previously stated, the purpose of speaker identification is to determine the identity of a speaker from a test utterance from that speaker, by searching through a set of trained speakers. For the open-set case it can be that the speaker being tested is not part of the speaker set.

For a set of c speakers, represented by the models $\lambda_1, \lambda_2, \ldots, \lambda_c$, the objective is to find the speaker model which has the maximum posterior probability for the input feature vector sequence, $X = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N\}$.

For any particular speaker model the Bayes classifier (BC) is the optimal classifier. The classification of a single feature vector, $\mathbf{x}$, is shown. We start with the Bayesian-rule

$$P(\lambda_j|\mathbf{x}) = \frac{f(\mathbf{x}|\lambda_j)P(\lambda_j)}{f(\mathbf{x})}, \tag{3.9}$$

where $P(\lambda_j|\mathbf{x})$ is the probability of model λ_j being the correct speaker if feature vector $\mathbf{x}$ is observed. $P(\lambda_j)$ is the probability of λ_j being the selected model. If we assume the more general case of an open-set problem we can make two decisions. The first is that speaker i has the highest probability of producing the feature $\mathbf{x}$, thus $\lambda(\mathbf{x}) = \lambda_i$. The second, the rejection option, is $\lambda(\mathbf{x}) = \lambda_o$, where λ_o is chosen if the model is not in the set of known speaker models. We define a function to associate a cost with each decision. We call this the cost function $\beta(\lambda_a|\lambda_b)$. This function indicates the cost of deciding on λ_a if the observed feature vector really belonged to λ_b. In order to simplify the equations, a few assumptions about the cost function is made:

- There is no cost in making a correct decision: $\beta(\lambda_i|\lambda_i) = 0$.
- All incorrect decisions have the same cost: $\beta(\lambda_i|\lambda_j) = 1$ for $i \neq j$.
- The cost of the rejection option is β_r: $\beta(\lambda_i|\lambda_i) = \beta_r$.

Stellenbosch University http://scholar.sun.ac.za

With some manipulation [77, chap. 4] we obtain the decision rule

$$\lambda(\mathbf{x}) = \lambda_i \text{ if}$$
$$f(\mathbf{x}|\lambda_i) = max_{j=1\ldots c}\ f(\mathbf{x}|\lambda_j) \geq (1-\beta_r)\sum_{k=1}^{c} f(\mathbf{x}|\lambda_k), \tag{3.10}$$

$$\text{else } \lambda(\mathbf{x}) = \lambda_o. \tag{3.11}$$

(3.12)

If we want to make a classification decision on more than one feature vector at a time, the above equation becomes

$$\lambda(\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N) = \lambda_i \ if$$
$$P(\lambda_i)\prod_{k=1}^{N} f(\mathbf{x}_k|\lambda_i) = max_{j=1\ldots c} P(\lambda_j)\prod_{k=1}^{N} f(\mathbf{x}_k|\lambda_j) \tag{3.13}$$
$$\geq (1-\beta_r)\sum_{l=1}^{c} P(\lambda_l)\prod_{k=1}^{N} f(\mathbf{x}_k|\lambda_l),$$

$$\text{else } \lambda(\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N) = \lambda_o. \tag{3.14}$$

(3.15)

If we take the logarithm of the equation to simplify the computations we get

$$\lambda(\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N) = \lambda_i \ if$$
$$\ln(P(\lambda_i)) + \sum_{k=1}^{N} \ln(f(\mathbf{x}_k|\lambda_i)) = max_{j=1\ldots c} \ln(P(\lambda_j)) + \sum_{k=1}^{N} \ln(f(\mathbf{x}_k|\lambda_j)) \tag{3.16}$$
$$\geq \ln(1-\beta_r) + \ln(\sum_{l=1}^{c} P(\lambda_l)\prod_{k=1}^{N} f(\mathbf{x}_k|\lambda_l)),$$

$$\text{else } \lambda(\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N) = \lambda_o. \tag{3.17}$$

(3.18)

If we assume that all speakers have an equal probability to be selected, $P(\lambda_j) = \frac{1}{c}$, and that all the test speakers are known, then the above equation simplify to the following

$$\lambda(\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N) = \lambda_i \ if$$
$$\sum_{k=1}^{N} ln(f(\mathbf{x}_k|\lambda_i)) = max_{j=1\ldots c} \sum_{k=1}^{N} \ln f(\mathbf{x}_k|\lambda_j)$$
$$= max_{j=1\ldots c} \sum_{k=1}^{N} \ln \sum_{i=1}^{M} \left(\frac{m_{ji}}{2\pi^{D/2}|\boldsymbol{\Sigma}_{ji}|^{1/2}}\right) e^{-\frac{1}{2}(\mathbf{x}_k-\boldsymbol{\mu}_{ji})^T \boldsymbol{\Sigma}_{ji}^{-1}(\mathbf{x}_k-\boldsymbol{\mu}_{ji})}, \tag{3.19}$$

Stellenbosch University http://scholar.sun.ac.za

where m_{ji}, $\boldsymbol{\mu_{ji}}$, $\boldsymbol{\Sigma_{ji}}$ are the the i'th mixture weight, mean vector and covariance matrix of the j'th speaker model respectively. In a speaker identification system the log likelihood scores of all speaker models are calculated. The model with the highest total score is then selected as the correct speaker.

3.1.4.2 Speaker verification

As previously stated, the purpose of speaker verification is to verify the identity claim of the speaker under evaluation. The speaker claims to be one of the speakers previously trained on the system and the system should determine whether the person is who he claims to be. The system does not need to known the speaker being tested.

The approach to speaker verification is to apply a likelihood ratio test to the input utterance. If the ratio is above a certain threshold, σ, the claim will be accepted. The likelihood ratio effectively gives an indication of how much better the model of the claimed speaker is than that of the non-claimant model. The threshold can be set to create a tradeoff between rejecting true claimants, which is known as the false rejection error, and accepting false claimants, known as the false acceptance error. For a claimed speaker λ_C and utterance $X = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N\}$ the speaker will be accepted if

$$\frac{f(X|\lambda_C)}{f(X|\lambda_{\overline{C}})} > t_1, \tag{3.20}$$

or in the log domain

$$\ln f(X|\lambda_C) - \ln f(X|\lambda_{\overline{C}}) > t_2, \tag{3.21}$$

where t_1 and t_2 are threshold constants, and $f(X|\lambda_C)$ is the likelihood that the utterance was produced by λ_C and $f(X|\lambda_{\overline{C}})$ is the likelihood that the utterance was not produced by the claimed speaker. The first likelihood is computed as follow

$$\begin{aligned} \ln f(X|\lambda_C) &= \frac{1}{N}\sum_{t=1}^{N} \ln f(x_i|\lambda_C) \\ &= \frac{1}{N}\sum_{t=1}^{N} \ln \sum_{i=1}^{M} \left(\frac{m_{Ci}}{2\pi^{D/2}|\boldsymbol{\Sigma_{Ci}}|^{1/2}} \right) e^{-\frac{1}{2}(\mathbf{x}_t - \boldsymbol{\mu_{Ci}})^T \boldsymbol{\Sigma_{Ci}}^{-1} (\mathbf{x}_t - \boldsymbol{\mu_{Ci}})}, \end{aligned} \tag{3.22}$$

Stellenbosch University http://scholar.sun.ac.za

where m_{Ci}, $\boldsymbol{\mu_{Ci}}$, $\boldsymbol{\Sigma_{Ci}}$ are the i'th mixture weight, mean vector and covariance matrix of the claimed speaker model, λ_C, respectively. The likelihood that the utterance was not produced by the speaker is determined from a collection of background speaker models. With a set of B background models the likelihood is calculated as follow

$$\begin{aligned}\ln f(X|\lambda_{\overline{C}}) &= \ln\left(\frac{1}{B}\sum_{b=1}^{B} f(X|\lambda_b)\right) \\ &= \ln\left(\frac{1}{B}\sum_{b=1}^{B}\left(\frac{1}{N}\sum_{t=1}^{T}\ln\sum_{i=1}^{M}\left(\frac{m_{bi}}{2\pi^{D/2}|\boldsymbol{\Sigma_{bi}}|^{1/2}}\right) e^{-\frac{1}{2}(\mathbf{x}_t-\boldsymbol{\mu_{bi}})^T\boldsymbol{\Sigma_{bi}}^{-1}(\mathbf{x}_t-\boldsymbol{\mu_{bi}})}\right)\right),\end{aligned} \tag{3.23}$$

where m_{bi}, $\boldsymbol{\mu_{bi}}$, $\boldsymbol{\Sigma_{bi}}$ are the i'th mixture weight, mean vector and covariance matrix of the b'th background speaker model respectively. It is clear that for the speaker verification case a subset of background speakers must be found from within the set of available speaker models. The two main issues in choosing background speakers are the number of speakers and the selection criterion. The ideal choice would be to use all the speakers as background models, but this may not be a computationally feasible solution in large speaker sets. Therefore, a tradeoff between the background size and computational efficiency must be made.

For the selection criteria one could choose the top closest speakers to the claimant. For two speaker models λ_a and λ_b, and training feature vector sequences X_a and X_b, the distance between the two speaker models using two training sequences, X_a from speaker a and X_b from speaker b, are defined as [53]

$$d(\lambda_a, \lambda_b) = \ln\frac{f(X_a|\lambda_a)}{f(X_a|\lambda_b)} + \ln\frac{f(X_b|\lambda_b)}{f(X_b|\lambda_a)}. \tag{3.24}$$

The ratio $\frac{f(X_a|\lambda_a)}{f(X_a|\lambda_b)}$ is a measure of how well speaker b scores with speaker a's speech compared to how well speaker a scores with his own speech. The more similar the models, the smaller the distance will become. Using these distance measures a background speaker set can be selected.

3.2 Summary

In this chapter we discussed speaker recognition using Gaussian mixture models. The initialisation and training of Gaussian mixture speaker models were described along with a number of

Stellenbosch University http://scholar.sun.ac.za

practical considerations. Classification of two speaker recognition applications, speaker identification and speaker verification, was provided. In Chapter 5 we will evaluate the performance of speaker identification using high quality and HF transmitted speech.

Stellenbosch University http://scholar.sun.ac.za

Part II

Robust speaker recognition

Stellenbosch University http://scholar.sun.ac.za

Chapter 4

Robust feature analysis techniques

In this chapter we provide a detailed description of the robust feature analysis techniques introduced in Section 1.2.2. Two feature sets, perceptual linear prediction and relative autocorrelation sequence mel-scale cepstrum, are discussed. This is followed by a number of intra-frame processing techniques: discarding the zero'th order cepstral coefficient, cepstral weighting, adaptive component weighting and cepstral post filtering. Finally inter-frame processing techniques are discussed: cepstral derivatives, cepstral mean subtraction, pole-filtered cepstral mean subtraction and RASTA processing.

4.1 Robust features

4.1.1 Perceptual linear prediction (PLP)

Perceptual linear prediction [41] takes advantage of several well known properties of the auditory system. These properties are simulated by practical approximations. The resulting auditory-like spectrum of speech is approximated by the autoregressive all-pole model.
The steps in calculating the PLP are as follows:

1. Convert the speech signal to the frequency domain using an FFT.

Stellenbosch University http://scholar.sun.ac.za

2. Warp radial frequency to Bark frequency and apply the critical-band filter.
3. Apply equal-loudness pre-emphasis.
4. Apply the intensity-loudness power-law.
5. Convert the signal back to the time domain using an IFFT.
6. Calculate the all-pole model using linear prediction.

This feature is a bit more robust than ordinary linear prediction coefficients, but still suffers from convolutional and additive noise. It is therefore commonly used in conjunction with other robust techniques.

The steps in calculating the PLP feature are discussed further in the following sections.

Time to frequency domain

A frame of speech is multiplied by a window function, such as the Hamming window, and then transformed to the frequency domain using the FFT.

Critical-band resolution curve

The first step is to warp the spectrum from the radial frequency (ω) to the Bark frequency (Ω) scale using the following relation

$$\Omega(\omega) = 6 \ln \left(\frac{\omega}{1200\pi} + \sqrt{1 + \left(\frac{\omega}{1200\pi} \right)^2} \right). \tag{4.1}$$

The power spectrum, $S(\Omega)$, of the warped spectrum is then calculated. The next step is to convolve the warped power spectrum with the power spectrum of a simulated critical-band

Stellenbosch University http://scholar.sun.ac.za

masking curve

$$\Psi(\Omega) = \begin{cases} 0, & \Omega < -1.3 \\ 10^{2.5(\Omega+0.5)}, & -1.3 \leq \Omega \leq -0.5 \\ 1, & -0.5 < \Omega < 0.5 \\ 10^{-1(\Omega-0.5)}, & 0.5 \leq \Omega \leq 2.5 \\ 0, & \Omega > 2.5 \end{cases} . \tag{4.2}$$

The convolution, given as

$$\Theta(\Omega_i) = \sum_{\Omega=-1.3}^{2.5} S(\Omega - \Omega_i)\Psi(\Omega), \qquad \text{for } i = 1, 2, \ldots, B, \tag{4.3}$$

produce samples of the critical band power spectrum at the chosen bark frequencies, Ω_i, where B is the number of samples. The sampling intervals are chosen so that when the critical bands are added together it equally represent the frequency scale.

Equal-loudness curve

Loudness is the perceived magnitude of an applied tone in the auditory system. A sound with a frequency of 1 kHz and a known intensity is used as a reference. The perceived loudness of another tone can then be determined by comparing it with the 1 kHz reference tone. An equal-loudness curve can then be constructed from this data. An approximation used in PLP for frequencies up to 5 kHz is [41]

$$E(\omega) = \frac{\omega^4(\omega^2 + 56.8 \cdot 10^6)}{(\omega^2 + 6.3 \cdot 10^2)^2 \cdot (\omega^2 + 0.38 \cdot 10^9)}. \tag{4.4}$$

For frequencies higher than 5 kHz an additional term is included

$$E(\omega) = \frac{\omega^4(\omega^2 + 56.8 \cdot 10^6)}{(\omega^2 + 6.3 \cdot 10^2)^2 \cdot (\omega^2 + 0.38 \cdot 10^9) \cdot (\omega^6 + 9.58 \cdot 10^{26})}. \tag{4.5}$$

The sampled bark power spectrum is then pre-emphasised by the simulated equal-loudness curve

$$L(\Omega(\omega)) = E(\omega)\Theta(\Omega(\omega)). \tag{4.6}$$

Stellenbosch University http://scholar.sun.ac.za

Intensity-loudness power-law

The following modification simulates the non-linear relationship between the intensity of a sound and its perceived loudness. A cubic root compression of the amplitude is performed [41]

$$\Phi(\Omega) = L(\Omega)^{\frac{1}{3}}. \tag{4.7}$$

Together with the equal-loudness pre-emphasis, this operation reduces the spectral amplitude variation of the critical band spectrum. Only a small AR model order is then required [41].

Frequency to time domain

The power spectrum that resulted from the amplitude compression in the previous steps is converted back to the time domain using the IFFT. The power spectrum in the frequency domain will result in an autocorrelation sequence in the time domain. The autocorrelation sequence is then directly used to calculate the LP coefficients.

Calculating the all-pole LP coefficients

The autocorrelation signal can be used to calculate the all-pole LP coefficients using the Levinson-Durbin [76, p.254] or Marple [78] algorithms.

4.1.2 Relative autocorrelation sequence (RAS) mel-scale cepstrum

Speech is mainly corrupted by additive and convolutional noise. Convolutional noise is convolved with the spectrum of the speech in the frequency domain, while being additive in the logarithmic and cepstral domain. Many techniques were developed to remove this kind of noise. Some of these, such as cepstral mean subtraction and RASTA processing will also be discussed in this chapter. Additive noise on the other hand, is added to the speech spectrum in the frequency and power spectrum domain. Speech enhancement techniques, such as spectral subtraction, attempts to remove the noise component in these domains.

Stellenbosch University http://scholar.sun.ac.za

The approach followed in [42] is based on the observation that additive noise is also additive in the autocorrelation domain due to the relation between the power spectrum and autocorrelation of a signal. The effect of the additive noise component is minimised by filtering the temporal trajectories of the short-time one-sided autocorrelation sequence of the speech. The resulting sequence is called the relative autocorrelation sequence (RAS).

The RASs are used, instead of the original speech signal, to calculate mel-scale frequency cepstral coefficients (MFCC). The resulting feature is called the RAS-MFCC. Furthermore, the delta cepstrum can be calculated as usual. The result is called the (RAS-MFCC-delta).

To demonstrate the trajectory filtering of the autocorrelation sequence we start by partitioning the noisy speech into M frames of length N (these frames can overlap if required), so that the framed speech is modelled as

$$y(m,n) = x(m,n) + w(m,n), \qquad \text{for } 0 \leq m \leq M-1 \text{ and } 0 \leq n \leq N-1, \tag{4.8}$$

where m is the frame index, n is the time index within a frame, $x(m,n)$ is the clean speech, $w(m,n)$ is the noise and $y(m,n)$ is the resulting noisy speech.

It is assumed that the additive noise component, $w(m,n)$, is uncorrelated with the speech. Therefore, the autocorrelation of the noisy speech is the sum of the autocorrelation sequences of the clean and noisy speech

$$r_{yy}(m,k) = r_{xx}(m,k) + r_{ww}(m,k), \qquad \text{for } 0 \leq m \leq M-1 \text{ and } 0 \leq k \leq N-1, \tag{4.9}$$

where k is the autocorrelation index and $r_{yy}(m,k)$, $r_{xx}(m,k)$, $r_{ww}(m,k)$ are the one-sided autocorrelation sequence of the noisy, clean and noise signals of the m'th frame respectively.

The unbiased one-sided autocorrelation sequence of a signal $y(m,n)$ can be calculated with

$$r_{yy}(m,k) = \frac{1}{N-k} \sum_{j=0}^{N-1-k} y(m,j)y(m,j+k), \qquad \text{for } 0 \leq k \leq N-1. \tag{4.10}$$

If the noise signal is assumed to be stationary over all frames, the autocorrelation sequences of all $r_{ww}(m,n)$ are identical and the subscript m can be dropped so that the sequence only

Stellenbosch University http://scholar.sun.ac.za

depends on the autocorrelation index k. Equation (4.9) then becomes

$$r_{yy}(m,k) = r_{xx}(m,k) + r_{ww}(k), \qquad \text{for } 0 \leq m \leq M-1 \text{ and } 0 \leq k \leq N-1. \quad (4.11)$$

By differentiating both sides of equation (4.11) with respect to the frame index m, the constant term $r_{ww}(k)$ disappears, so that

$$\frac{\partial r_{yy}(m,k)}{\partial m} = \frac{\partial r_{xx}(m,k)}{\partial m}, \qquad \text{for } 0 \leq m \leq M-1 \text{ and } 0 \leq k \leq N-1. \quad (4.12)$$

The sequence

$$\{\frac{\partial r_{yy}(m,k)}{\partial m}\}_{k=0}^{N-1}, \quad (4.13)$$

is called the RAS of the m'th frame of the noisy speech. Equation (4.12) suggests that if our stationary assumption about the noise is true and that the noise is uncorrelated with the speech signal, then the RAS of the noisy speech is equal to the RAS of the original clean speech.

The RAS can be obtained by a polynomial approximation

$$\frac{\partial r_{yy}(m,k)}{\partial m} \approx \frac{1}{T_L} \sum_{t=-L}^{L} t \cdot r_{yy}(m+t,k), \qquad \text{for } 0 \leq m \leq M-1 \text{ and } 0 \leq k \leq N-1, \quad (4.14)$$

where L is the number of sequences before and after the current sequence, so that the summation is over the range of frames over which the polynomial fitting is done, and

$$T_L = \sum_{t=-L}^{L} t^2. \quad (4.15)$$

The operation in equation (4.14) can be interpreted as the filtering of the temporal autocorrelation trajectory using a finite impulse response (FIR) filter with the transfer function given by

$$H(z) = \frac{1}{T_L} \sum_{t=-L}^{L} t \cdot z^t. \quad (4.16)$$

This is a high-pass filter that suppresses the slowly varying components of the temporal autocorrelation trajectory.

Stellenbosch University http://scholar.sun.ac.za

4.2 Intra-frame processing of features

A cepstrum modification technique is classified as an intra-frame processor if it operates on, or takes advantage of, only the current analysis frame.

4.2.1 Discarding the zero'th order cepstral coefficient

When speaker recognition is performed in adverse conditions, the zero'th order cepstral coefficient is often discarded [43, 11]. This is due to the fact that the zero'th order cepstral coefficient has a direct relationship with the energy (or gain) of the represented speech frame. If the speech is corrupted by additive or correlated noise, the gain is modified and consequently the zero'th order cepstral coefficient will change as well. Furthermore, recalling equation (2.33), it is seen that the spectral information of the speech is conveyed by the other cepstrum coefficients. It can be argued that the zero'th order cepstral coefficient does not contribute a significant amount of information about the speaker and can therefore be discarded without significant information loss.

4.2.2 Cepstral weighting (CW)

Cepstral weighting (CW), or liftering, is the process of multiplying each feature component by a fixed weighting factor. CW can be interpreted as a window function multiplied by the cepstrum to emphasise or suppress certain coefficients of the cepstrum. The resulting cepstral feature vector is denoted by

$$c_{cw}(n) = w(n) \cdot c(n), \tag{4.17}$$

where $c_{cw}(n)$ is the resulting weighted cepstrum, $w(n)$ is the window function and $c(n)$ is the original cepstrum.

The idea behind CW is to account for the sensitivity of the lower order cepstral coefficients to the spectral slope or the sensitivity of higher order coefficients to noise. The three most common windowing functions, for L'th order cepstral coefficients, follows:

Stellenbosch University http://scholar.sun.ac.za

Rectangular

The rectangular window weight each component with the same value and is given by

$$w(n) = \begin{cases} 1, & n = 1, 2, \ldots, L \\ 0, & \text{otherwise} \end{cases}. \tag{4.18}$$

This is the simplest window and is the default CW window which is implicitly applied to all feature vectors if no other window is defined.

Quefrency liftering

Quefrency liftering or the ramp window weight each component with its index [45]

$$w(n) = \begin{cases} n, & n = 1, 2, \ldots, L \\ 0, & \text{otherwise} \end{cases}. \tag{4.19}$$

Quefrency liftering emphasise the higher order coefficients. It is therefore suitable only to compensate for spectral tilt. If the speech is contaminated by noise, it is not advisable to use this window function.

Band-pass liftering

Band-pass liftering suppress both the lower and higher order cepstral coefficients by applying a raised sinusoidal function of the form [46]

$$w(n) = \begin{cases} 1 + \frac{L}{2}\sin(\frac{n\pi}{L}), & n = 1, 2, \ldots, L \\ 0, & \text{otherwise} \end{cases}. \tag{4.20}$$

This window is useful in situations where the speech is contaminated by both additive and convolutional noise.

4.2.3 Adaptive component weighting (ACW)

In essence, the adaptive component weighting (ACW) [44, 47, 48] technique modifies the LP spectrum in two ways. It emphasises the narrow-band components and suppresses the broad-

Stellenbosch University http://scholar.sun.ac.za

band components. The result of these modifications is to produce a spectrum that is more robust for channel variations.

The all-pole model can be written in a form that more clearly indicates where the poles are

$$H(z) = \frac{1}{1 + \sum_{k=1}^{P} a_k z^{-k}} = \sum_{k=1}^{P} \frac{r_k}{1 - p_k z^{-1}}, \tag{4.21}$$

where P is the order of the model, a_k are the LP coefficients, r_k are the residues of the poles, p_k. Each pole represents the centre frequency, ω_k, and bandwidth, B_k, of the k'th component of the LP spectrum. In general, the narrow-band components represent formants, while the broad-band components are representations of the transmission channel and glottal characteristics [44].

To understand why the above mentioned modifications were made to the LP spectrum we made the following observations:
First, the sensitivity of an all-pole model is expressed as [49, p.336]

$$\frac{\partial p_k}{\partial a_l} = \frac{p_k^{P-l}}{\prod_{j=1, j\neq k}^{P}(p_k - p_j)} = r_k p_k^{P-l}, \qquad \text{for } k = 1, 2, \ldots, P \text{ and } l = 1, 2, \ldots, P. \tag{4.22}$$

Therefore, the sensitivity of a pole to errors in the LP coefficients is directly proportional to the size of its residue. Poles with larger residues will be affected more than poles with smaller residues.

The sensitivity of the centre frequency, bandwidth and residues of a pole to channel variations were investigated in [44]. It was found that the centre frequency and bandwidth of the broad-band components showed a larger variance than those of the narrow-band components. The variance of the residues of both the narrow and broad-band components were relatively large. This result confirms the conclusion of the first observation.

From these two observations it would seem that the LP spectrum must be modified so that the effects of the residues are either eliminated or reduced and that the contribution of the broad-band components must be attenuated.

In ACW these modifications are obtained by setting the residues of all the components equal to unity. This is equivalent to normalising each component by its residue. The modified ACW

Stellenbosch University http://scholar.sun.ac.za

spectrum is then given by

$$H_{acw}(z) = \sum_{k=1}^{P} \frac{1}{(1 - p_k z^{-1})} = \frac{N(z)}{1 + \sum_{k=1}^{P} a_k z^{-1}} = \frac{N(z)}{A(z)}, \tag{4.23}$$

where

$$N(z) = \sum_{l=1}^{P} \prod_{k=1 \neq l}^{P} (1 - p_k z^{-1}). \tag{4.24}$$

This can be written as a moving average (MA) filter with $P - 1$ zeros

$$N(z) = P(1 + \sum_{k=1}^{P-1} b_k z^{-k}). \tag{4.25}$$

where b_k is the LP coefficients of $N(z)$. This modification to the LP spectrum yields a peak value for each component that is inversely proportional to the bandwidth of the component. The narrow-band components are therefore emphasised, while the broad-band components are suppressed.

The ACW cepstrum is expressed in the z-domain by

$$\begin{aligned} C_{acw}(z) &= \ln H_{acw}(z) \\ &= \ln \frac{N(z)}{A(z)} \\ &= \ln \frac{1}{A(z)} - \ln \frac{1}{N(z)}, \end{aligned} \tag{4.26}$$

and the ACW cepstrum is given by

$$c_{awc}(n) = \begin{cases} \ln P, & n = 0 \\ c_{lp}(n) - c_{nn}(n), & n = 1, 2, \ldots, (P-1) \end{cases}, \tag{4.27}$$

where $c_{lp}(n)$ is the conventional LP cepstrum calculated from a_k and $c_{nn}(n)$ is the LP cepstrum that can be calculated from

$$b_k = \frac{P - k}{P} \cdot a_k, \qquad \text{for } k = 1, 2, \ldots, P. \tag{4.28}$$

The ACW cepstrum can thus be calculated in four easy steps:

1. Find $c_{lp}(n)$ from a_k.

Stellenbosch University http://scholar.sun.ac.za

2. Find b_k from a_k, which gives $N(z)$.

3. Find $c_{nn}(n)$ from b_k.

4. Find $c_{acw}(n) = c_{lp}(n) - c_{nn}(n)$.

Another interpretation of the ACW cepstrum is that it estimates the cepstrum of the channel, $c_{nn}(n)$, and that it is subtracted from the conventional LP cepstrum to obtain a robust feature that is more stable under channel variations. We also discuss two other techniques, cepstral mean subtraction and post-filter cepstral mean subtraction, that also attempt to estimate the contribution of the channel and then removes it. However, in [44, 48] it was shown that the ACW cepstrum outperforms both these techniques.

4.2.4 Cepstral post filtering (CPF)

Another feature that is designed to remove convolutional channel noise is the cepstral post filter [50, 48], which was first introduced for speech enhancement. Just as the ACW method, the post filter attempts to emphasise the formants. The post filter spectrum can be obtained from $A(z)$, the LP spectrum. The transfer function is expressed as

$$H_{pf}(z) = \frac{A(\frac{z}{\beta})}{A(\frac{z}{\alpha})}, \qquad \text{for } 0 < \beta < \alpha \leq 1. \tag{4.29}$$

In the cepstral domain the post filter cepstrum is given by

$$c_{pf}(n) = \begin{cases} 0, & n = 0 \\ (\alpha^n - \beta^n) \cdot c_{lp}(n), & n = 1, 2, \ldots, P \end{cases}, \tag{4.30}$$

where P is the order and $c_{lp}(n)$ is the original LP cepstrum. We can therefore think of the post filtered cepstrum as a weighted version of the standard LP cepstrum. Figure 4.1 shows the plot of the window function of a post filter with $\alpha = 1$ and $\beta = 0.9$.

The post filter is very sensitive to changes in α and β. As β decreases, the spectral tilt becomes more noticeable. A decrease in α causes a formant bandwidth broadening. This is not a desired effect. In practical applications α is usually set to unity [48] and equation (4.30) becomes

$$c_{pf}(n) = c_{lp}(n) - \beta^n c_{lp}(n), \qquad \text{for } n = 1, 2, \ldots, P. \tag{4.31}$$

Stellenbosch University http://scholar.sun.ac.za

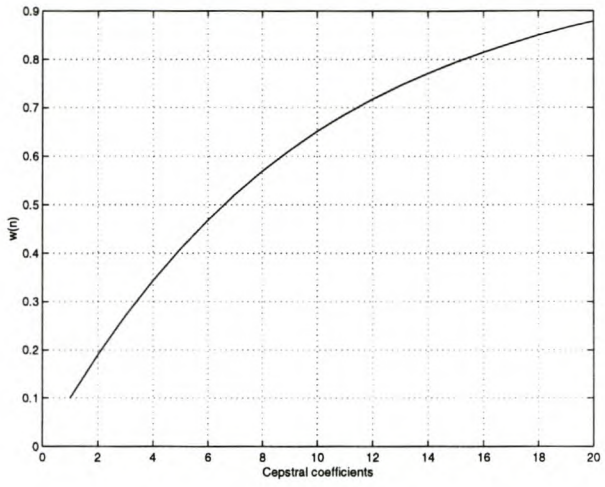

Figure 4.1: *The window function produced by a post filter with* $\alpha = 1$ *and* $\beta = 0.9$.

In this case the second term in equation (4.31) acts as an estimation of the channel, just as in cepstral mean subtraction and ACW, which is subtracted from the standard LP cepstrum.

4.3 Inter-frame processing of features

A cepstrum modification technique is classified as an inter-frame processor if it operates on, or takes advantage of not only the current frame, but also the neighbouring analysis frames.

4.3.1 Cepstral derivative (delta cepstrum)

A cepstrum represents the spectral properties of the current speech analysis frame. However, it doesn't carry any information about the temporal or transitional information between a sequence of speech frames. The cepstral derivative or delta cepstrum provides such information by differentiating a sequence of cepstral feature vectors over time. The delta cepstrum, which is the first derivative of the cepstrum, is defined as follow

$$\begin{aligned} \frac{\partial c_{lp}(n,m)}{\partial m} &= \Delta c_{lp}(n,m) \\ &\approx \mu \sum_{k=-K}^{K} k \cdot c_{lp}(n,m+k), \end{aligned} \tag{4.32}$$

Stellenbosch University http://scholar.sun.ac.za

where $c_{lp}(n, m)$ denotes the n'th LP cepstrum coefficient of the m'th time frame, K is the number of forward and backward frames that are included in the calculation and μ is a normalisation constant. The polynomial expansion is only an approximation [79] of the derivative. The polynomial approximation in equation (4.32) is used due to stability issues arising when evaluating the derivative directly.

The advantage of the temporal filtering, achieved by the delta cepstrum, is that slow varying processes, such as additive noise, have little effect on the cepstrum [51, 43]. The delta cepstrum is therefore more robust to noise and channel variations.

The delta cepstrum is used in conjunction with the standard cepstrum as a means to add temporal information to the cepstrum. On its own it does not perform well, but the combination of cepstrum and delta cepstrum perform better than just the cepstrum alone [51].

4.3.2 Cepstral mean subtraction (CMS)

Another popular technique is cepstral mean subtraction (CMS) [52, 79, 53, 43]. CMS attempts to create a cepstral feature that is less dependent on channel variations. The effect of a transmission channel, such as a telephone channel, can be approximated as a linear transform on the transmitted speech. This is shown as

$$Y(z) = X(z)H(z), \tag{4.33}$$

where $Y(z)$ is the filtered speech, $X(z)$ is the original speech and $H(z)$ is the response of the transmission channel. In the logarithmic domain, such as the cepstral domain, this multiplication by the channel response becomes an additive component

$$\ln Y(z) = \ln X(z) + \ln H(z). \tag{4.34}$$

CMS is based on two assumptions. The first is that the LP cepstrum represents both the speech/speaker and the channel. The second assumption is that the mean of the LP cepstrum of clean speech is zero. CMS takes advantage of equation (4.34), where it is stated that the contribution of the channel is additive, and the second assumption by estimating the mean of a

Stellenbosch University http://scholar.sun.ac.za

contaminated cepstrum and subtracting it from the cepstrum

$$c_{cms}(n) = c_{lp}(n) - E[c_{lp}(n)], \tag{4.35}$$

where $c_{lp}(n)$ is the unmodified LP cepstrum. The expectation, $E[c_{lp}(n)]$, is calculated over a number of frames. This expectation, or mean, of the LP cepstrum is considered to be the contribution of the transmission channel due to the second assumption.

CMS performs significantly better in cases where a system is trained with speech obtained by one channel, while testing is done using speech obtained from a different channel. On the other hand, CMS performs worse in systems where training and testing is done using speech obtained from the same channel. This performance loss is due to the assumption that the cepstral mean of clean speech is zero.

4.3.3 Pole-filtered cepstral mean subtraction (PFCMS)

Just as in CMS, pole-filtered CMS (PFCMS) attempts to estimate the contribution of a transmission channel and to removes it. As previously stated, the narrow-band components of the LP spectrum represent the formants and are less susceptible to noise and channel variations. The formants are more informative about the speech than about the channel or noise components. The broad-band LP components on the other hand are more indicative of the transmission channel, spectral tilt and glottal variations [54, 43]. The broad-band poles are therefore a better representations of the transmission channel.

PFCMS takes advantage of this fact through broadening the bandwidth of the formants. This is done by moving them radially away from the unit circle, while keeping the centre frequencies unchanged. All poles that have a radius larger than a preset threshold, r_{th}, are moved to the threshold value. PFCMS therefore attempts to remove the residual effects of speech from the cepstral mean used for channel normalisation [54]. The LP cepstrum of the resulting modified LP spectrum is then calculated and denoted by $c_{mlp}(n)$. Due to the suppression of the formants this cepstrum contains less speech information than $c_{lp}(n)$. An estimate of the channel can now be made from the modified cepstrum. Finally the channel estimate is subtracted from the

Stellenbosch University http://scholar.sun.ac.za

unmodified LP cepstrum to produce the PFCMS cepstrum

$$c_{pfcms}(n) = c_{lp}(n) - E[c_{mlp}(n)]. \tag{4.36}$$

This feature has been shown to perform better than the standard CMS technique [54, 43].
In summary, the steps for calculating the PFCMS cepstrum is as follow:

1. Select a threshold radius r_{th}.
2. For each speech frame:
   - Calculate the LP poles z_i.
   - For each pole move the pole to the threshold radius if $|z_i| > r_{th}$.
   - Calculate $c_{mlp}(n)$ from the modified LP spectrum.
3. Find the channel estimate, $E[c_{mlp}(n)]$, over a number of speech frames.
4. Calculate $c_{pfcms}(n)$ using equation (4.36).

4.3.4 Relative spectral (RASTA) processing

RelAtive SpecTrAl (RASTA) [56, 55] processing is another technique that transforms the speech to an alternative domain and then filters the speech trajectory. Two domains will be specified and discussed, namely the log domain and a lin-log domain.

Typical feature vector analysis techniques represents all information in the spectrum. This includes information about the speech, the transmission channel and noise components. It is noted in [55] that many of the non-linguistic related components in speech varies either slower or faster than the typical rate of change of the vocal tract. It was suggested that we use a band pass filter that, over time, suppress the slow and fast changing components in the speech that falls outside the typical range of vocal tract variation. The transfer function of the band pass filter applied in RASTA is expressed as

$$H(z) = \frac{2 + z^{-1} - z^{-3} - 2z^{-4}}{10z^{-4}(1 - 0.98z^{-1})}. \tag{4.37}$$

Stellenbosch University http://scholar.sun.ac.za

The low and high cut-off frequency regulates the slowest and fastest spectral changes allowed to pass through the band pass filter respectively. The high pass section of the filter will eliminate the slowly varying components of the speech, such as those contributed by the convolutional transmission channel. The low pass section of the filter will help to smooth out the fast varying components attributed to analysis artifacts [56, 55].

With the band pass filter already specified the two domains in which filtering is conducted will be described. The first transformation is from the linear spectral domain to the logarithmic spectral domain.
Forward transform

$$Y(\omega) = \ln X(\omega), \tag{4.38}$$

backward transform

$$X(\omega) = e^{Y(\omega)}, \tag{4.39}$$

where the backward transform is executed after the band pass filtering. In this domain the effects of a convolutional transmission channel is additive. Once again it is assumed that this additive component change slowly, or is constant, over a short period of time. Then the high pass section of the band pass filter will be especially effective to remove the convolutional noise.

RASTA filtering, combined with PLP, in the log domain demonstrated considerable improvement over the performance of only PLP when convolutional noise was added to the speech [55, 56]. However, the same source reported that the performance increase is not so considerable when the noise source is additive (uncorrelated noise) to the speech. The problem lies in the fact that the additive noise contributes to a non-linear effect in the log domain. The solution offered is to transform the speech to a domain that is linear for small amplitudes and logarithmic for larger amplitudes. This domain is denoted the lin-log domain and is expressed as:
forward transform

$$Y(\omega) = \ln(1 + JX(\omega)), \tag{4.40}$$

backward transform

$$\begin{aligned} X(\omega) &= \frac{e^{Y(\omega)} - 1}{J} \\ &\approx \frac{e^{Y(\omega)}}{J}, \end{aligned} \tag{4.41}$$

Stellenbosch University http://scholar.sun.ac.za

where J is a constant value. The approximation in the backward transform is to ensure that $X(\omega)$ is not negative after the transform. This transformation demonstrated an improvement over that of the log domain transform if the speech is corrupted by additive noise [56]. However, the results are highly dependent on the value of the constant J. For each noise level a different optimum value for J results. This poses a practical problem, since the noise levels are usually not known prior to the use of the system. Three approaches in handling this problem has been suggested in [80]:

1. Multiple recognisers - The first suggestion is to train more than one recogniser, each with a different J value from clean speech. In the recognition phase a noise estimate is made. A value for J and its corresponding recogniser is selected based on the noise estimate. The disadvantage of this method is that more than one recogniser must be trained.

2. Multiple J values for one recogniser - Instead of training multiple recognisers a single recogniser can be trained for a set of J values. During testing an estimate of the noise, E_{noise}, is made and use to construct a J value for classification

$$J = \frac{1}{C \cdot E_{noise}}, \tag{4.42}$$

   where C is a fixed constant during classification. When training the recogniser a number of different models ([80] used four) is trained, each with a different C value that depends on an order of magnitude. This method increases the size of the training set, adds more parameters to the classifier and is computationally more expensive.

3. Spectral mapping - Both of the previous two methods introduce significant overhead. In spectral mapping only one recogniser is needed, with one set of models and a fixed reference value, J_{ref}, of J. It was noted in [80] that the noise dependency on J introduces a deterministic source of variability into the analysis that could be analytically determined. However, in the absence of a suitable analytical solution, an empirically derived linear mapping that transforms the spectrum from a current J value, obtained from a noise estimate, to a spectrum processed with the reference value. This implies a mapping between $\ln(1 + JX(\omega))$ and $\ln(1 + J_{ref}X(\omega))$.

It was demonstrated that in most cases the spectral mapping technique performs better than the

Stellenbosch University http://scholar.sun.ac.za

other two methods [80].

4.4 Summary

The techniques considered in this chapter are used to combat the effects of additive and convolutional (transmission channel) noise on speech. The majority of them were concerned with removing the effects of convolutional noise. This is due to the fact that the convolutional noise results in an additive component in the logarithmic domain, such as the cepstral domain. The techniques can be sorted in the following subcategories:

- Techniques that suppress or emphasise certain components of the LP cepstrum that is more susceptible to noise and spectral tilt. They are: Discarding the zero'th order cepstral coefficient, cepstral weighting and cepstral post filtering.
- Techniques that modify the narrow-band (formants) or broad-band components of the LP spectrum. They are: Adaptive component weighting and pole-filtered cepstral mean subtraction.
- Techniques that estimate the contribution of the channel to the LP cepstrum and removes it: Cepstral mean subtraction and pole-filtered cepstral mean subtraction.
- Techniques that filter the spectrum or LP cepstrum over time, thus removing slow or fast varying components: Cepstral derivatives and RASTA.

A technique based on the human perception of speech, perceptual linear prediction, was introduced. The rationale is that if aspects of the auditory system is incorporated into the recognition system, the system will mimic the human capability to be more robust against certain distortions in the speech. Finally, the relative autocorrelation sequence mel-scale cepstrum was one of the techniques, together with lin-log RASTA, that attempts to remove additive noise. It is based on the observation that additive noise produce an additive component in the power spectrum and autocorrelation domain. In a similar fashion to that of cepstral derivatives, the autocorrelation sequences are filtered over time to remove the additive noise component.

Stellenbosch University http://scholar.sun.ac.za

Chapter 5

Speaker recognition experiments

In this chapter we present the results of some of our speaker recognition experiments. In the first section we introduce the the speech database used in all the experiments. The second section covers the speaker recognition experiments. Finally, the conclusion provides an overview of the results.

5.1 Speech database

A number of speech databases are available, each with its own characteristics. The major differences between the databases are:

- Speech quality.
- Speech bandwidth.
- Transmission channel.
- Recording conditions.
- Variation between recording sessions.
- Length of utterances.

Stellenbosch University http://scholar.sun.ac.za

- Database size.

The database used in all our experiments is the TIMIT acoustic-phonetic speech corpus. TIMIT was designed to provide speech data for the acquisition of acoustic-phonetic knowledge and for the development and evaluation of automatic speech recognition systems. However, TIMIT is also used in various other applications, such as speaker recognition. The reason for this is the high quality of the speech, which makes it ideal for testing a technique without the interference of noise and channel variations.

The TIMIT database contains a total of 6300 sentences, 10 sentences spoken by each of the 630 speakers from 8 major dialect regions in the United States. Of the 10 sentences per speaker, 2 was read by all speakers (SA), 5 was also read by 7 other speakers (SX) and 3 was only read by that speaker alone (SI).

For the HF tests a version of the TIMIT database that was transmitted over a long distance HF transmission channel was used.

5.2 Speaker identification

The speaker recognition experiments were all conducted on a subset of the TIMIT database. For all our experiments the 38 speakers (24 male and 14 female) from dialect region 1, in the training directory, were used. The 5 SX files of each speaker were used for training and the 3 SI and 2 SA files were used for identification. The same set of data was used for the HF tests.

The high quality speech data was passed through a pre-emphasis filter with $\alpha = 0.98$. The speech was segmented into 32 ms frames, shifted by 16 ms at a time. This means that there is a 50% overlap of the speech data between feature vectors. A Hamming window was applied to each frame before a 16'th order LP cepstrum was calculated.

The HF data were down sampled to 8 kHz. This is done because the low-pass filter of the modulation system has a very sharp cutoff frequency at around 3 kHz. The down sampling effectively discard the higher frequency component which contains only noise and no usable

Stellenbosch University http://scholar.sun.ac.za

speech information. The data were then passed through a pre-emphasis filter with $\alpha = 0.98$. Next the data were segmented into 32 ms frames, shifted by 16 ms at a time. A Hamming window was then applied to each frame before a 10'th order LP cepstrum was calculated.

The models were trained and tested on the same quality of speech. Therefore, a set of models were trained on the high quality speech for the tests involving the high quality speech, and another set of models were trained on the HF quality speech. The experiments using HF quality speech used these models instead of the high quality equivalents.

Three sets of speaker identification experiments were conducted. The speaker model for the first set was a full covariance Gaussian model. The second set used a diagonal covariance 16 mixture Gaussian model and the third set used a diagonal covariance 32 mixture Gaussian model. The three different models were then used to conduct speaker identification on high quality and HF speech using the following setups:

None: The LP cepstral features were used without modifying any of the coefficients. This setup was evaluated on models using both high quality and HF quality speech.

DZC: The zero'th order cepstral coefficient was discarded as described in Section 4.2.1. This setup was evaluated on all models using HF quality speech.

DZC + CMS: The zero'th order cepstral coefficient was discarded and cepstral mean subtraction (Section 4.3.2) was applied. This setup was evaluated on all models and HF quality speech.

DZC + CPF: The zero'th order cepstral coefficient was discarded and cepstral post filtering (Section 4.2.4) was applied. The CPF had an $\alpha = 1$ and a $\beta = 0.9$. This setup was evaluated on all models and HF quality speech.

The results of these experiments are shown in Table 5.1. The rows represent each model type, while the columns correspond to the various setups. The results in the first column were obtained from speaker identification on the high quality TIMIT data with no processing of the cepstral features. All three models demonstrated very good results, with the full Gaussian model performing slightly worse than the other two models. This is due to the capability of the

Stellenbosch University http://scholar.sun.ac.za

Table 5.1: *Average speaker recognition performance.*

Speaker Model	High Quality	High Frequency			
	Cepstral Processing				
	None	None	DZC	DZC+CMS	DZC+CPF
FG	99.5%	17.4%	19.5%	15.3%	19.5%
GMM(16)	100%	15.8%	17.9%	12.7%	20.3%
GMM(32)	100%	13.4%	17.1%	12.3%	18.9%

mixture models to better fit the subclasses produced by each speaker (see Section 3.1). The next four columns display the performance of the three models on HF quality speech data.

In the first of these setups, the cepstral features were not modified. All three models produced less than adequate results. This result prompted us to investigate the distortions introduced by the HF channel combined with the modulation system (Part III). We noticed that the mixture models perform slightly worse than the single Gaussian model on the HF speech. A possible reason for this is the ability of the GMMs to model subclasses in the speaker space. When the noise levels are as large as the levels we experienced in the HF speech, most of the LP cepstral features migrate to a single location in cepstral space (see Section 6.4). The mixtures therefore model the noise process and not the speaker at all. Since the single Gaussian model has to fit only one Gaussian PDF on all the data, it manages to include more of the speaker information.

The next column displays the performance of the models for the HF speech data when the zero'th order cepstral coefficient is discarded. By discarding the zero'th order cepstral coefficient we eliminate the influences of changes in the energy of the speech, the effect of ionospheric absorption (Section 6.3.2) and the effects of a phase error in modulation (Section 6.2.2). The single Gaussian model still performs better than the two mixture models, but we notice that the difference in performance between the 16 and 32 mixture models are smaller.

The next column shows the performance of the models when the zero'th order cepstral coefficient is discarded and cepstral mean subtraction (CMS) is performed. The performance is somewhat worse than the baseline results. This can happen when using CMS, if training and testing are performed on the same channel. However, the database was created over a period

Stellenbosch University http://scholar.sun.ac.za

of time and in different conditions and the transmission channel should be quite different for the various files. It seems like the distortion in the speech is not mainly due to channel distortions. That is, the distortion is not convolutional noise, for which CMS was designed for. The investigation in Chapter 6 confirms that the HF channel in not just a simple convolutional channel.

The last column displays the performance of the models when the zero'th cepstral coefficients is discarded and cepstral post filtering (CPF) is performed. The CPF improved the performance of the models slightly. This is due to the fact that CPF emphasises the formants of the speech. CPF was also designed to combat the effect of a convolutional channel. It is therefore also ineffective against additive noise.

5.3 Conclusion

The poor speaker identification results presented in this chapter lead to the investigation of the HF channel and the effects it has on the LP cepstrum. One of the largest contributors to the poor results is the noise in the signal and that is not easy to eliminate. Sadly, not even the best speech enhancement techniques will increase the SNR enough to make a noticeable difference to the speaker identification results. It therefore seems that speaker identification over an HF channel is only viable if the noise levels are moderate.

Stellenbosch University http://scholar.sun.ac.za

Part III

The HF transmission channel and speech enhancement

Stellenbosch University http://scholar.sun.ac.za

Chapter 6

High frequency transmitted speech

6.1 Introduction

One of the goals of this project is to conduct experiments on speech transmitted over a high frequency (HF) channel to determine how viable speaker recognition is for this transmission medium. Initial investigation [81], using a portion of the TIMIT database transmitted over an HF channel, reported very unsatisfactory results. This motivated us to identify the effects that are introduced by the HF channel and the modulation system [82]. We then determine how these effects deform the LP coefficients and LP cepstrum [83].

In each case we first describe the effect and then determine how it modifies the speech in the spectral or correlation domain. Using this knowledge, we then determine the deformation of the LP coefficients and LP cepstrum. In the cases where the correlation domain is used the autocorrelation of the signal is calculated. The result is then used to calculate the correlations in the augmented Wiener-Hopf equation (2.24)

$$\begin{bmatrix} r(0) & \mathbf{r}^H \\ \mathbf{r} & \mathbf{R} \end{bmatrix} \begin{bmatrix} 1 \\ -\mathbf{w}_f \end{bmatrix} = \begin{bmatrix} P_M \\ \mathbf{0} \end{bmatrix}. \tag{6.1}$$

The augmented Wiener-Hopf equation is then solved for $\mathbf{w}_f$ and P_M. This result is then used to recursively calculate the LP cepstrum using equation (2.33).

Stellenbosch University http://scholar.sun.ac.za

In Section 6.2 we investigate the effects introduced by the modulation system. Section 6.3 cover the effects introduced by the high frequency transmission channel and in Section 6.4 we investigate the effects of additive Gaussian white noise on the demodulated speech signal.

6.2 Single sideband modulation system

The system investigated here uses single sideband (SSB) modulation to modulate the speech signal before transmission and demodulate it after reception. In the following sections we first describe SSB modulation, followed by an investigation of the problems introduced by non-ideal modulation and demodulation.

6.2.1 Description of single sideband modulation

SSB modulation is very efficient in terms of the bandwidth required for transmission. Unlike other modulation systems, such as large carrier amplitude modulation, SSB only requires an amount of bandwidth equal to the bandwidth of the transmitted signal and not double that. In SSB modulation only the upper or lower sideband of the speech spectra is transmitted, not both as in the case of large carrier amplitude modulation. In a modulation system such as large carrier amplitude modulation the transmitted signal is multiplied with a sinusoidal signal, which copies the complete spectrum around both the positive and negative carrier frequency components. When the signal is demodulated two copies of the spectral component are shifted to the zero frequency. The information needed to reconstruct the original signal is therefore transmitted twice. SSB modulation eliminates this overhead by transmitting only one of the sidebands of the spectra and not both. Figure 6.1 shows a spectral representation of this process. SSB modulation is given by

$$y_{\mp}(t) = x(t)\cos(\omega_c t) \pm \widehat{x}(t)\sin(\omega_c t), \tag{6.2}$$

where $y_{+}(t)$ produce the upper sideband, $y_{-}(t)$ produces the lower sideband, $\widehat{x}(t)$ is the Hilbert transform of $x(t)$ and ω_c is the carrier frequency. The SSB signal can be generated by a balanced modulator. The modulated signal and carrier are phase shifted by $90°$ to form the second term in

Stellenbosch University http://scholar.sun.ac.za

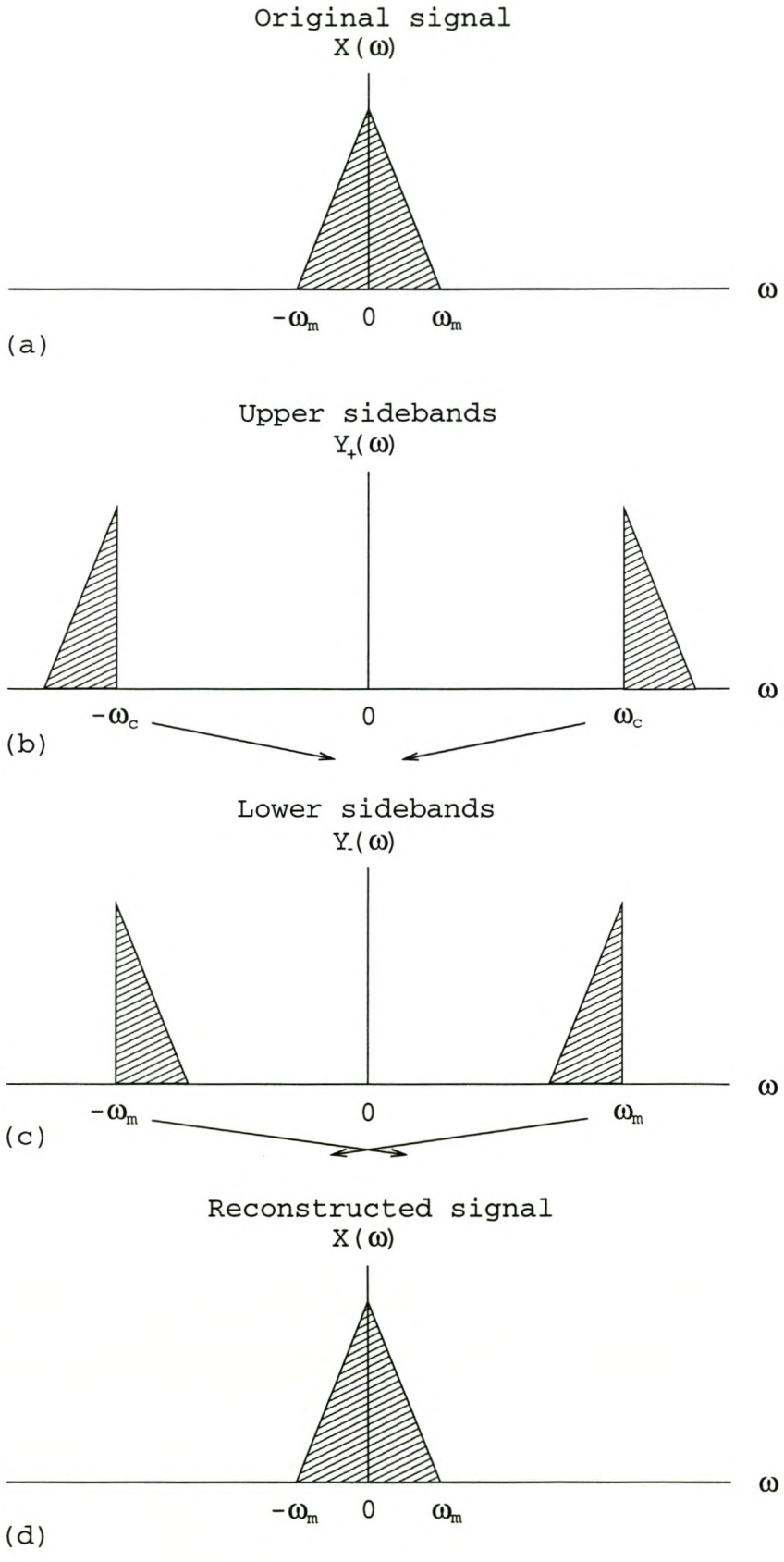

Figure 6.1: *SSB spectrum: Demonstrating upper and lower sideband modulation and their reconstruction.*

Stellenbosch University http://scholar.sun.ac.za

equation (6.2). This technique is known as the phase-shift method of generating SSB signals and is graphically presented in Figure 6.2. Demodulation of an SSB modulated signal is achieved

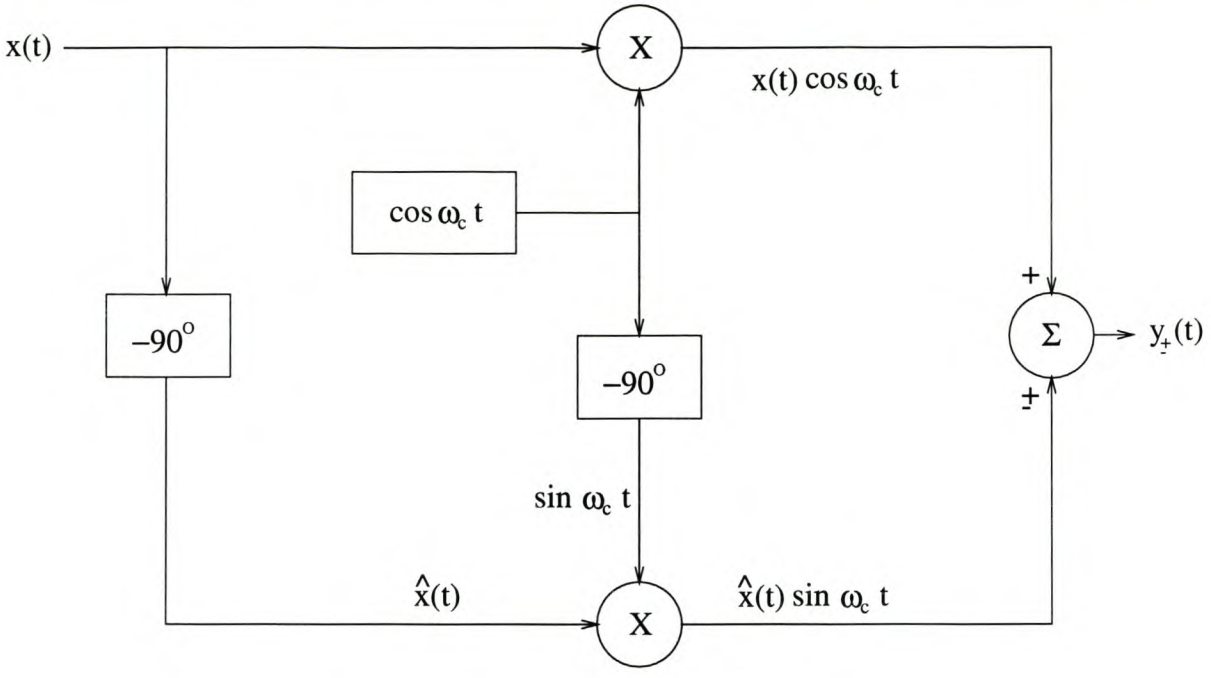

Figure 6.2: *Generation of an SSB signal.*

by multiplying the SSB modulated signal by a sinusoidal signal with a frequency equal to the carrier frequency

$$\begin{aligned}\widetilde{x}(t) &= y_{\mp}(t) \cdot \cos(\omega_c t) \\ &= [x(t)\cos(\omega_c t) \pm \widehat{x}(t)\sin(\omega_c t)] \cdot \cos(\omega_c t) \\ &= x(t)\cos^2(\omega_c t) \pm \widehat{x}(t)\sin(\omega_c t)\cos(\omega_c t) \\ &= \frac{1}{2}x(t) + \frac{1}{2}x(t)\cos(2\omega_c t) \pm \frac{1}{2}\widehat{x}(t)\sin(2\omega_c t),\end{aligned} \tag{6.3}$$

where the last two terms produce the upper or lower sidebands at twice the carrier frequency (see Figure 6.3). The two frequency terms at twice the carrier frequency can be removed by a low pass filter. This produces a scaled version of the transmitted signal

$$\widetilde{x}(t) = \frac{1}{2}x(t). \tag{6.4}$$

The demodulation process is demonstrated in Figure 6.4.

The above description of SSB modulation and demodulation assumes ideal conditions. However in practice the components in a system can cause deviations from expected performance.

Stellenbosch University http://scholar.sun.ac.za

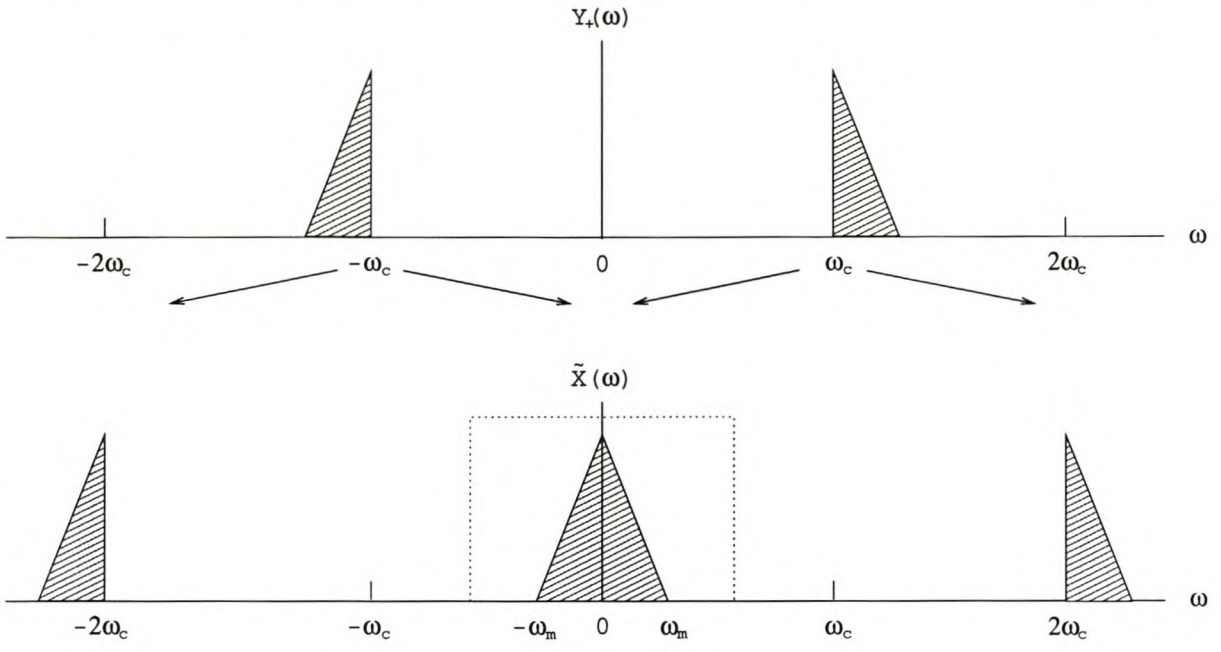

Figure 6.3: *Demonstrating SSB demodulation in spectral domain.*

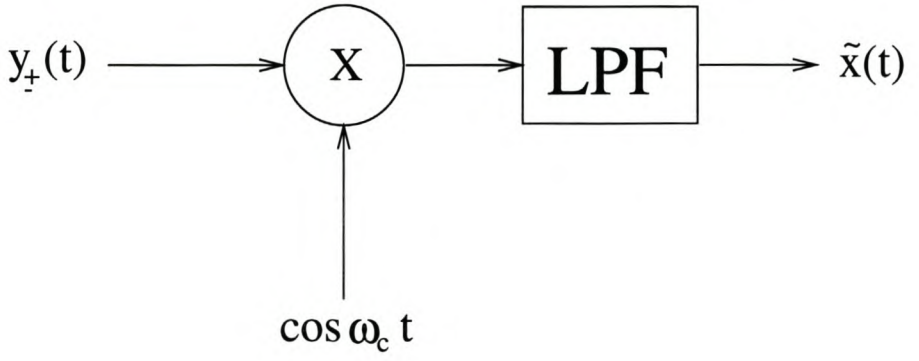

Figure 6.4: *Demodulation of an SSB signal.*

The most important of these deviations are frequency and phase errors in the sinusoidal signals used to modulate and demodulate the signals. This problem arises when the signals generated by the transmitter and the receiver are not synchronised. To demonstrate this effect we determine the resulting signal for both a phase and frequency error. Let the incoming SSB signal be

$$y_{\mp}(t) = x(t)\cos(\omega_c t) \pm \widehat{x}(t)\sin(\omega_c t), \tag{6.5}$$

as generated in equation (6.2). Furthermore let the locally generated carrier signal be

$$\cos[(\omega_c + \Delta\omega)t + \theta], \tag{6.6}$$

Stellenbosch University http://scholar.sun.ac.za

where $\Delta\omega$ is the difference in frequency and θ is the phase error between the signal generator of the receiver and the transmitter. If we demodulate the incoming signal we get

$$\begin{aligned}\widetilde{x}(t) &= [x(t)\cos(\omega_c t) \pm \widehat{x}(t)\sin(\omega_c t)] \cdot \cos\left((\omega_c + \Delta\omega)t + \theta\right) \\ &= \frac{1}{2}x(t)\left[\cos\left((\Delta\omega)t + \theta\right) + \cos\left((2\omega_c + \Delta\omega)t + \theta\right)\right] \\ &\quad \mp \frac{1}{2}\widehat{x}(t)\left[\sin\left((\Delta\omega)t + \theta\right) - \sin\left((2\omega_c + \Delta\omega)t + \theta\right)\right].\end{aligned} \tag{6.7}$$

The double frequency components can be eliminated by a low-pass filter so that equation (6.7) simplifies to

$$\widetilde{x}(t) = \frac{1}{2}x(t)\cos(\Delta\omega t + \theta) \mp \frac{1}{2}\widehat{x}(t)\sin(\Delta\omega t + \theta). \tag{6.8}$$

To verify the correctness of this equation we set both the frequency and phase errors equal to zero, so that equation (6.8) simplifies to

$$\widetilde{x}(t) = \frac{1}{2}x(t). \tag{6.9}$$

The autocorrelation of this is

$$r_{\widetilde{x}\widetilde{x}}(n) = \frac{1}{4}r_{xx}(n). \tag{6.10}$$

The autocorrelation of the received signal is therefore only a scaled version of the transmitted signal. However, since the gain of the transmitted signal is modified by the transmission and the amplifiers of the receiver, the scale factor may not be equal to 0.25.

This is the result of demodulation with no frequency and phase error. In the subsequent sections we will consider the two cases in isolation to determine how they deform the LP coefficients and the LP cepstrum.

6.2.2 Effect of a phase error on LP coefficients

In the first case we consider only a phase error by setting the frequency error equal to zero, $\Delta\omega = 0$. Equation (6.8) then simplifies to

$$\widetilde{x}(t) = \frac{1}{2}x(t)\cos(\theta) \mp \frac{1}{2}\widehat{x}(t)\sin(\theta). \tag{6.11}$$

Stellenbosch University http://scholar.sun.ac.za

The demodulated signal is a linear combination of the original transmitted signal and the Hilbert transformed version of the signal using equation (6.11). First we determine how the phase error will affect the correlation domain and then use this result to calculate the LP coefficients. Lets determine the autocorrelation of $\widetilde{x}(n)$

$$\begin{aligned} r_{\widetilde{x}\widetilde{x}}(n) &= \widetilde{x}(n) * \widetilde{x}(-n) \\ S_{\widetilde{x}\widetilde{x}}(\omega) &= \widetilde{X}(\omega) \cdot \widetilde{X}(-\omega) \\ &= \widetilde{X}(\omega) \cdot \widetilde{X}^*(\omega) \\ &= \frac{1}{2}X(\omega)[\cos(\theta) \mp \sin(\theta)H(\omega)] \cdot \frac{1}{2}X^*(\omega)[\cos(\theta) \mp \sin(\theta)H^*(\omega)], \\ &= \frac{1}{4}X(\omega)X^*(\omega) \cdot [\cos(\theta) \mp \sin(\theta)H(\omega)] \cdot [\cos(\theta) \mp \sin(\theta)H^*(\omega)] \\ &= \frac{1}{4}X(\omega)X^*(\omega) \cdot [\cos^2(\theta) \mp \sin(\theta)\cos(\theta)(H(\omega) + H^*(\omega)) + \sin^2(\theta)]. \end{aligned} \tag{6.12}$$

where $S_{\widetilde{x}\widetilde{x}}(\omega)$ is the power spectrum of $\widetilde{x}(n)$, and $H(\omega)$ is the frequency response of the Hilbert transform as given by equation (A.1). Since $[H(\omega) + H^*(\omega)] = 0$ and $[\cos^2(\theta) + \sin^2(\theta)] = 1$, this equations simplifies to

$$\begin{aligned} S_{\widetilde{x}\widetilde{x}}(\omega) &= \frac{1}{4}X(\omega)X^*(\omega) \\ r_{\widetilde{x}\widetilde{x}}(n) &= \frac{1}{4}r_{xx}(n). \end{aligned} \tag{6.13}$$

Thus, the autocorrelation of the demodulated signal with a phase error is only a scaled version of the autocorrelation of the transmitted signal and is the same as the correctly demodulated signal. Using equation (6.13) to calculate the correlations in the Wiener-Hopf equation (6.1), we can determine the AR parameters

$$\begin{aligned} \begin{bmatrix} r_{\widetilde{x}}(0) & \mathbf{r}_{\widetilde{\mathbf{x}}}{}^H \\ \mathbf{r}_{\widetilde{\mathbf{x}}} & \mathbf{R}_{\widetilde{\mathbf{x}}} \end{bmatrix} \begin{bmatrix} 1 \\ -\mathbf{w}_{\widetilde{x}} \end{bmatrix} &= \begin{bmatrix} P_{M_{\widetilde{x}}} \\ \mathbf{0} \end{bmatrix} \\ \begin{bmatrix} \frac{1}{4}r_x(0) & \frac{1}{4}\mathbf{r}_{\mathbf{x}}{}^H \\ \frac{1}{4}\mathbf{r}_{\mathbf{x}} & \frac{1}{4}\mathbf{R}_{\mathbf{x}} \end{bmatrix} \begin{bmatrix} 1 \\ -\mathbf{w}_{\widetilde{x}} \end{bmatrix} &= \begin{bmatrix} P_{M\widetilde{x}} \\ \mathbf{0} \end{bmatrix} \\ \begin{bmatrix} r_x(0) & \mathbf{r}_{\mathbf{x}}{}^H \\ \mathbf{r}_{\mathbf{x}} & \mathbf{R}_{\mathbf{x}} \end{bmatrix} \begin{bmatrix} 1 \\ -\mathbf{w}_{\widetilde{x}} \end{bmatrix} &= \begin{bmatrix} 4P_{M\widetilde{x}} \\ \mathbf{0} \end{bmatrix}. \end{aligned} \tag{6.14}$$

The LP coefficients of equation (6.11) is the same as the LP coefficients of the transmitted signal, $x(n)$. The minimum mean-squared error (MMSE), $P_{M_{\widetilde{x}}}$, of $\widetilde{x}(n)$ differ from the MMSE of

Stellenbosch University http://scholar.sun.ac.za

$x(n)$ by a constant only. The MMSE is only used to calculate the zero'th order cepstral coefficient. The zero'th order cepstral coefficient is the only component that changed and becomes

$$c_{\widetilde{x}}(0) = \ln \sqrt{0.25 P_{M_x}}. \tag{6.15}$$

In robust speech applications the zero'th order cepstral coefficient is usually discarded. A phase error in the demodulator will therefore not cause any major deformations in the LP coefficients and LP cepstrum.

6.2.3 Frequency error in SSB demodulation

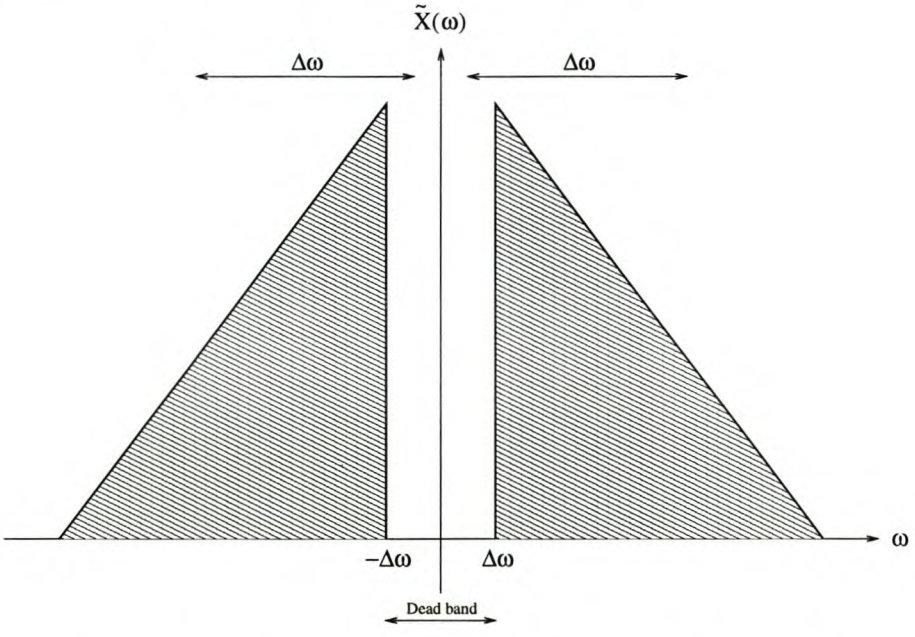

Figure 6.5: *Demonstrating the frequency shift of the SSB sidebands due to a frequency error in demodulation.*

We now consider the case where only a frequency error is present during demodulation. The demodulated signal with no phase error is obtained from equation (6.8) by setting $\theta = 0$, this leaves us with

$$\widetilde{x}(t) = \frac{1}{2}x(t)\cos(\Delta\omega t) \mp \frac{1}{2}\widehat{x}(t)\sin(\Delta\omega t). \tag{6.16}$$

In order to obtain more insight into the effect of a frequency error we transform the signal to

Stellenbosch University http://scholar.sun.ac.za

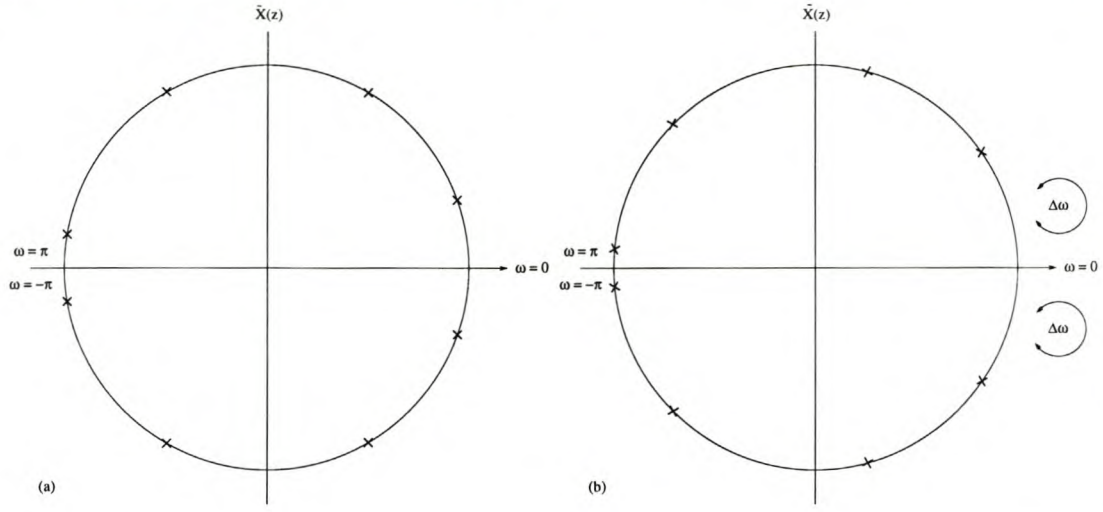

Figure 6.6: *Demonstrating the pole shifts, caused by a frequency error: (a) original poles, (b) shifted poles.*

the frequency domain. In the frequency domain equation (6.16) can be written as

$$\begin{aligned}\widetilde{X}(\omega) = &\frac{1}{4}[X(\omega-\Delta\omega)+X(\omega+\Delta\omega)] \mp \frac{1}{4j}[\widehat{X}(\omega-\Delta\omega)-\widehat{X}(\omega+\Delta\omega)] \\ =&\frac{1}{4}[X(\omega-\Delta\omega)+X(\omega+\Delta\omega)] \pm \frac{1}{4}[sgn(\omega-\Delta\omega)X(\omega-\Delta\omega) \\ &- sgn(\omega+\Delta\omega)X(\omega+\Delta\omega)] \\ =&\frac{1}{4}[(1 \pm sgn(\omega-\Delta\omega))X(\omega-\Delta\omega)+(1 \mp sgn(\omega+\Delta\omega))X(\omega+\Delta\omega)],\end{aligned} \tag{6.17}$$

so that

$$\widetilde{X}(\omega) = \frac{1}{2}[X(\omega-\Delta\omega)u(\pm\omega \mp \Delta\omega) + X(\omega+\Delta\omega)u(\mp\omega \mp \Delta\omega)], \tag{6.18}$$

where $u(t)$ is the unit step function given by

$$u(t) = \begin{cases} 1, & t \geq 0 \\ 0, & t < 0 \end{cases}. \tag{6.19}$$

A frequency error during demodulation produces a frequency shift of the two sidebands. Depending on the sign of $\Delta\omega$, and whether the upper or lower sidebands are transmitted, the sidebands will shift towards or away from each other. Figure 6.5 demonstrates this effect for $\Delta\omega < 0$ and upper sideband modulation. In the z-plane the poles and zeros will rotate towards or away from the $\omega = 0$ axis (see Figure 6.6). These shifts cause an aliasing effect to be introduced at either the $\omega = 0$ frequency component or around the Nyquist frequency. At the

Stellenbosch University http://scholar.sun.ac.za

other end of the spectrum a dead band is formed by shifting 'silent' parts of the spectrum into our analysis band. The frequency shift, aliasing and dead band is better demonstrated with a few examples. For the following examples a speech segment from the first sentence (*sa1.wav*) of the first speaker (*fcjf0* in dialect region one) in the TIMIT database is used. The first example demonstrates a frequency error of 10 Hz during demodulation. The result is shown in

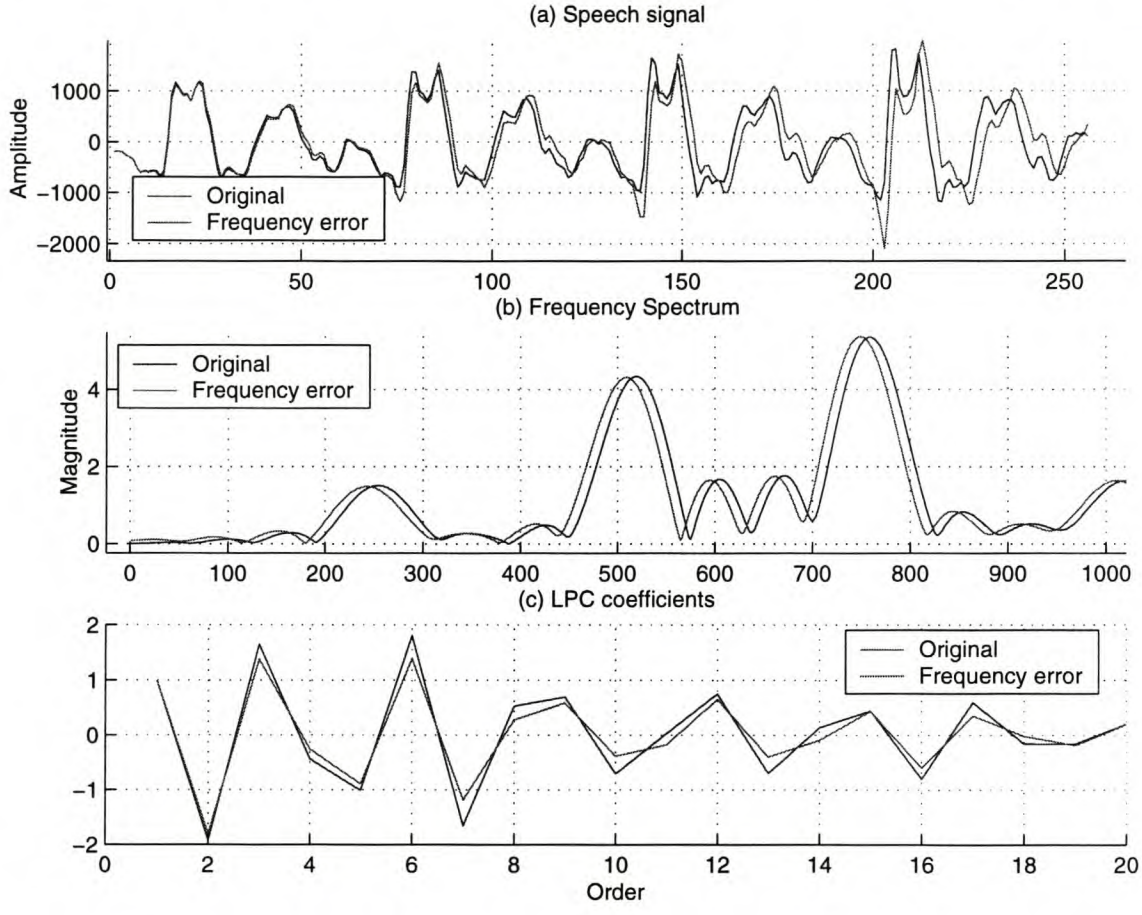

Figure 6.7: *The effects of a* 10 *Hz frequency error on a speech signal: (a) The speech signals, (b) The frequency spectrum, (c) The LP coefficients.*

Figure 6.7, part (a) shows the speech signals, part (b) shows the first 1000 Hz of the spectrum and part (c) shows the LP coefficients of the original and deformed signals. The speech signal displays only a slight mismatch between the two signals. The spectral shift of 10 Hz towards the zero frequency component can be clearly seen. The LP coefficients shows only minor deformation. A corresponding second example demonstrates a frequency error of 100 Hz. The result is shown in Figure 6.8, where part (a) shows the speech signals, part (b) the frequency spectrum of the first 1000 Hz and part (c) the LP coefficients. In this case the speech signal resulting

Stellenbosch University http://scholar.sun.ac.za

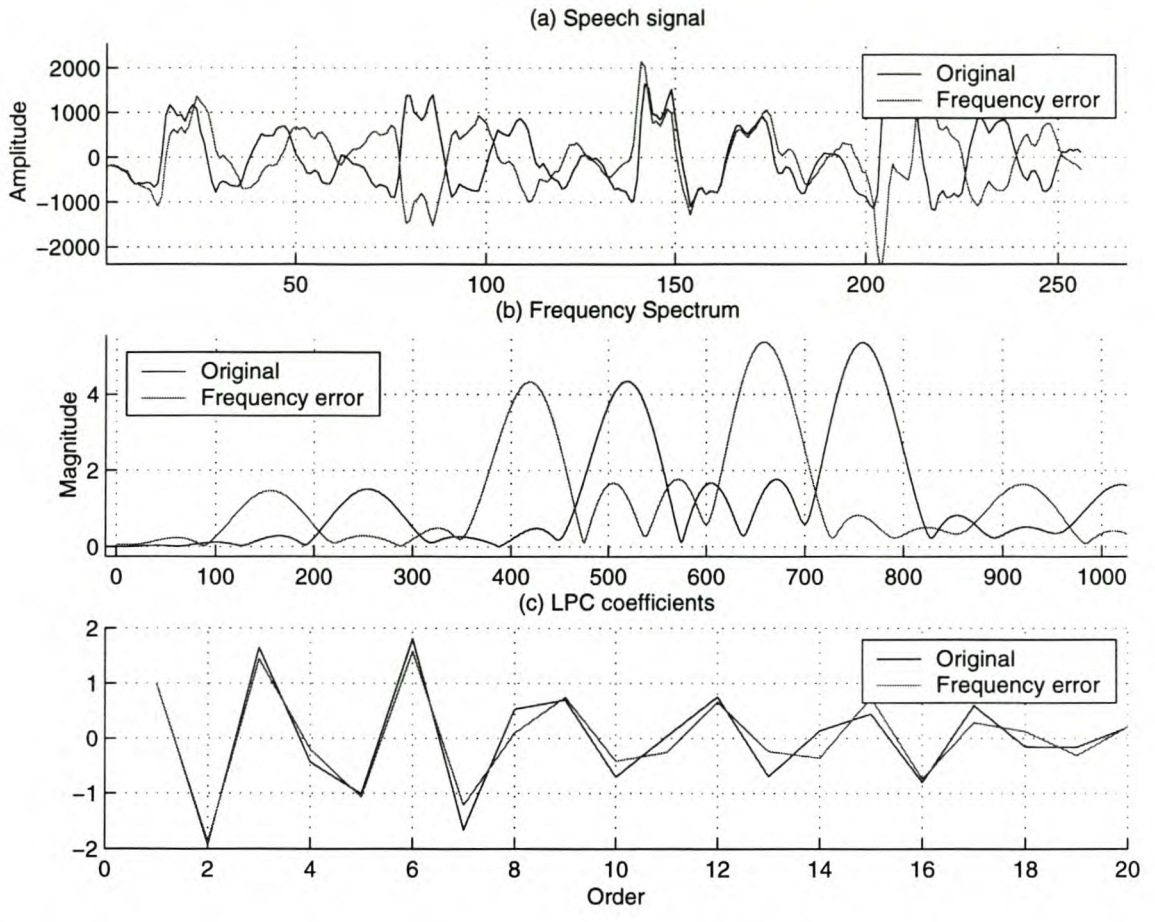

Figure 6.8: *The effects of a* 100 *Hz frequency error on a speech signal: (a) The speech signals, (b) The frequency spectrum, (c) The LP coefficients.*

from the frequency error is quite different from the original transmitted signal. Once again we note the spectral shift, in this case the shift is 100 Hz towards the zero frequency component. The LP coefficients, while somewhat more distorted than the 10 Hz case, still shows only minor deformation. As the frequency error increases the LP coefficients shows increasingly more deformation. The LP cepstrum shows similar behaviour.

In order to demonstrate the aliasing effect and the dead band more clearly a rather large frequency error of 1000 Hz is used. Figure 6.9 part (a) shows the aliasing effect. Note how the frequency component at approximately 760 Hz has shifted to around -240 Hz and the frequency component at approximately -760 Hz has shifted to around 240 Hz. Figure 6.9 part (b) shows the spectrum in the frequency range of 6000 to 8000 Hz. The frequency components at 8000 Hz has shifted to 7000 Hz. The 'silent' part of the spectrum that were above 8000 Hz has

Stellenbosch University http://scholar.sun.ac.za

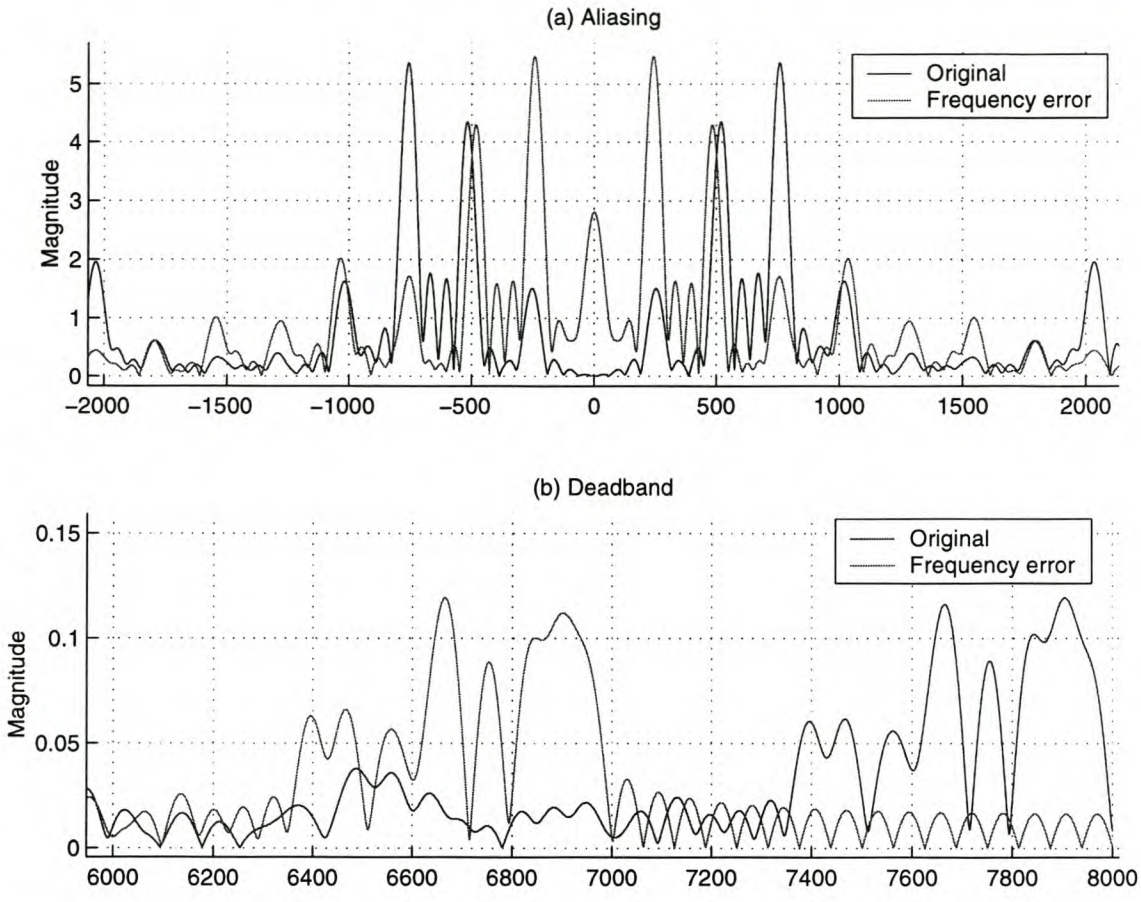

Figure 6.9: *Demonstrating the aliasing and dead band effects introduced by the frequency error: (a) Aliasing around the zero frequency, (b) The dead band at the Nyquist frequency.*

shifted into the 7000 Hz to 8000 Hz range to form the dead band.

6.2.4 Phase and frequency errors in SSB modulation

The effects of both a phase and frequency errors on the LP coefficients are shown in Figure 6.10. There are four plots. In both figures the phase error is the same ($\theta = \frac{\pi}{5}$), but with different frequency errors. Part (a) has a frequency error of 10 Hz, while part (b) has a frequency error of 100 Hz. In both figures the graphs have the following meaning: The blue line is the LP coefficients for no phase and frequency error. The black line is the LP coefficients of only a phase error. The red line is the LP coefficients of only a frequency error. The green line is the LP coefficients of both a phase and frequency error.

Stellenbosch University http://scholar.sun.ac.za

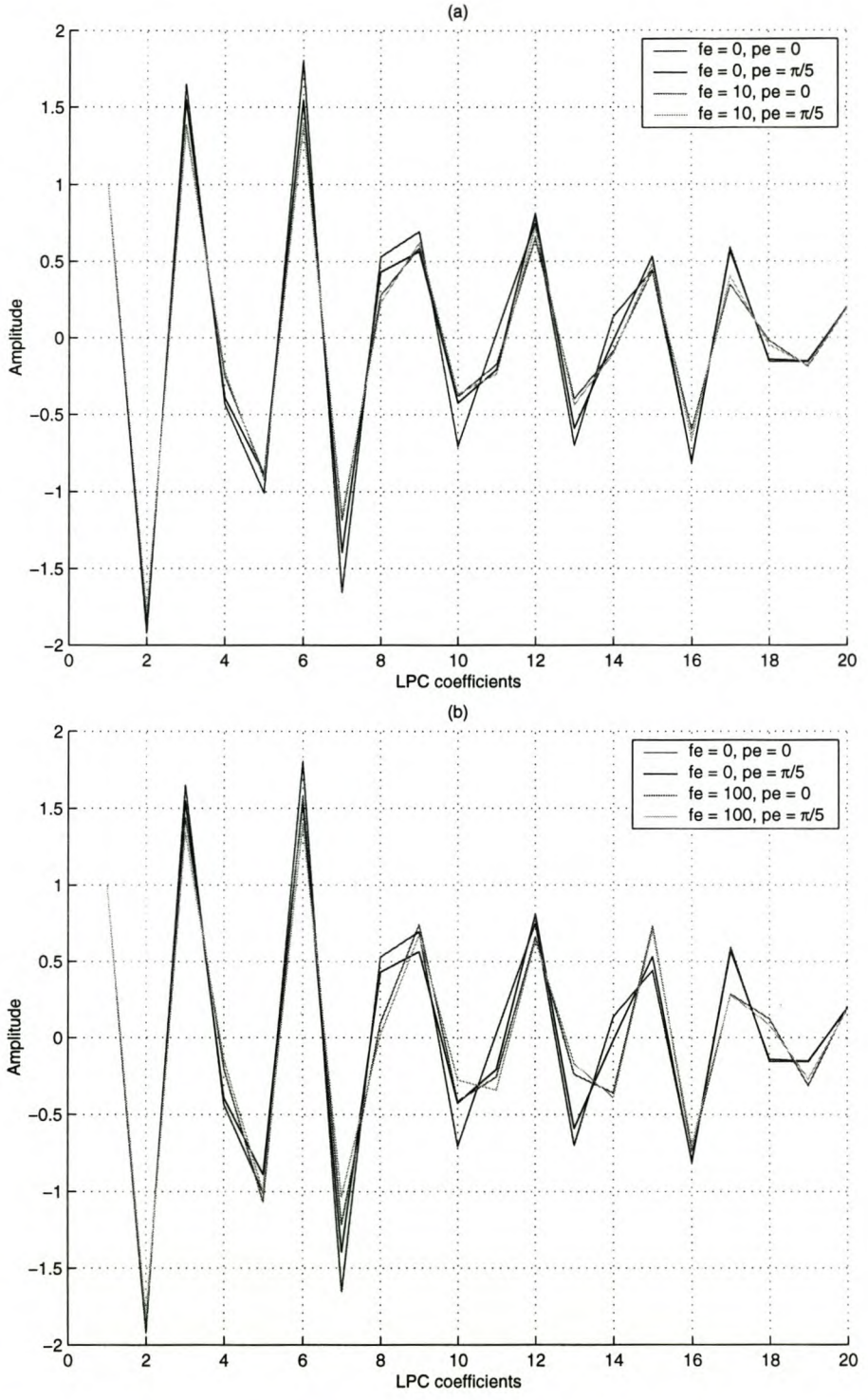

Figure 6.10: *Demonstrating the difference between no phase or frequency error, only a phase error, only a frequency error and both a phase and frequency error: (a) Frequency error is* 10 *Hz and the phase error is* $\frac{\pi}{5}$*, (b) Frequency error is* 100 *Hz and the phase error is* $\frac{\pi}{5}$.

Stellenbosch University http://scholar.sun.ac.za

It is seen that a phase error results in only a small deformation of the LP coefficients. This is because of the non-ideal nature of the Hilbert transform implementation. The frequency error clearly deforms the LP coefficients more than a phase error alone. The graph with both a phase and frequency error is very close to that of the frequency error graph. This results show that frequency errors are dominant over the phase errors. Furthermore, a combination of the two errors tend to reflect the effect of the frequency error.

6.2.5 Summary of single sideband modulation system

In this section SSB modulation was described. Non-ideal components in a system can cause the frequency and phase of the signal generators in the transmitter and the receiver to lose synchronisation. As a consequence frequency and phase errors are introduced during demodulation. The two effects were investigated in isolation.

We considered what happens if only a phase error where present during demodulation first. There was found that only the zero'th order cepstral coefficient changed, while the other components stayed unmodified.

Secondly, we investigated what happens if only a frequency error was present during demodulation. There was shown that the frequency error caused the spectrum of the sidebands to shift (depending on the sign of the frequency error and whether the upper or lower sidebands were transmitted) towards or away from the zero'th frequency component. Despite these shifts the LP coefficients of the demodulated signal showed relatively small changes to the LP coefficients of the original signal for smaller shifts.

When both a frequency and phase error are present the resulting deformation is mostly governed by the effect of the frequency error. We can expect modern modulation systems to produce only small frequency errors. Therefore, even with both errors present, the modulation system is expected to deform the LP coefficients and LP cepstrum only moderately.

Another consideration is that it is not possible to construct a perfect Hilbert transform. In practice only an approximation of the Hilbert transform is used. This approximation can possibly

Stellenbosch University http://scholar.sun.ac.za

cause minor deformation of the LP coefficients and the LP cepstrum.

6.3 The HF channel

6.3.1 Short description of the ionosphere

A complete study of the ionosphere is outside the scope of this project, but a quick overview is provided.

Radio wave transmission of signals with a carrier frequency between 3 and 30 MHz is termed high frequency band radio propagation. Transmission of this class of carrier frequencies depends on the refraction of the radio wave from the ionosphere. The ionosphere is a region high above the surface of the earth, where the air is sufficiently ionised by ultraviolet sunlight and X-rays so that radio waves can be absorbed or reflected. The reflection and absorption is controlled by the number of free electrons in this region [84, p.26-4]. The ionosphere is usually partitioned into the following layers [85, pp.276-303][84, p.26-4]:

- C Layer: This layer is insignificant in communication and is found 50-70 km above earth. It exists only during daylight and the density corresponds with the elevation of the sun.
- D Layer: This layer is 70-90 km above the earth and is only present during day light. The layer reflects very low and low frequency radio waves, absorb medium frequency waves and weakens high frequency waves through partial absorption. The absorption is caused by the high density of air and consequently high collision frequency between electron and neutral molecules.
- E Layer: This layer is 100-120km above the earth. It is important for transmission of high frequencies during the day over distances less than 1000 km. The ionisation density is highly dependent on the elevation of the sun. Areas of high ionisation density can sometimes be found in this layer. These irregular cloud like areas are known as sporadic E. Sporadic E can prevent waves that would usually penetrate the E layer from reaching higher layers.

Stellenbosch University http://scholar.sun.ac.za

- F_1 Layer: This layer is 175-225 km above the earth and only exists during the day. Some reflection of high frequency waves occur, but this layer act mostly as an absorber.

- F_2 Layer: This layer is 225-400 km above the earth and is the primary reflection region for long distance high frequency transmission. The height of the peak and ionisation levels depends on the day, season and sunspot cycles. Unlike the other layers the F_2 layer is not very dependent on the elevation of the sun. This is due to the fact that at such low air densities and molecular collision rates the medium can store the received solar energy for hours and can release electrons even during the night. During the night, the F_1 and F_2 layers merge to form the F layer with a peak at around 300 km above earth.

The height and density of these layers are not constant but fluctuate strongly depending on the level of radiation. Consequently the ionisation level differs considerably during the day and night.

In the sections to follow we discuss some of the effects caused by the ionosphere and investigate to what extend they deform the LP coefficients and the LP cepstrum. In all cases we will ignore the effect of noise and treat it in a separate section.

6.3.2 Absorption

The transmitted signal can be absorbed by the different layers of the ionosphere. This causes a fading effect at the receiver. Ionospheric fading usually change slowly within a period of about 10 minutes [82]. Most good receivers attempt to compensate for this effect with automatic gain control (AGC). To investigate the effect of absorption we first define a received signal that underwent fading (see Figure 6.11)

$$y(n) = Ax(n), \tag{6.20}$$

where A is the fading constant.

We first determine how ionospheric absorption will affect the correlation domain and then use this result to calculate the LP coefficients.

 Stellenbosch University http://scholar.sun.ac.za

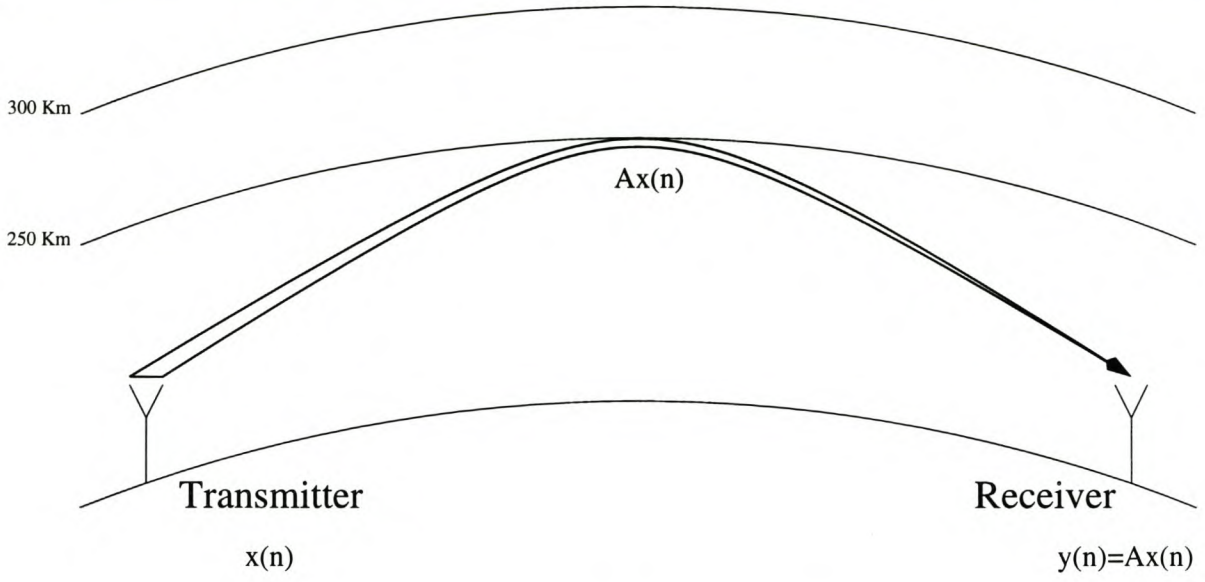

Figure 6.11: *Ionospheric Absorption.*

The autocorrelation of $y(n)$ is

$$\begin{aligned} r_{yy}(n) &= y(n) * y(-n) \\ S_{yy}(\omega) &= Y(\omega) \cdot Y^*(\omega) \\ &= AX(\omega) \cdot A^*X^*(\omega) \\ &= A^2 S_{xx}(\omega) \\ r_{yy}(n) &= A^2 r_{xx}(n), \end{aligned} \tag{6.21}$$

where $S_{xx}(\omega)$ is the power spectrum of $x(n)$, obtained by taking the Fourier transform of the cross correlation $r_{xx}(n)$. Using equation (6.21) we can determine the correlations in the augmented Wiener-Hopf equation (6.1)

$$\begin{aligned} \begin{bmatrix} r_y(0) & \mathbf{r_y}^H \\ \mathbf{r_y} & \mathbf{R_y} \end{bmatrix} \begin{bmatrix} 1 \\ -\mathbf{w}_y \end{bmatrix} &= \begin{bmatrix} P_{My} \\ \mathbf{0} \end{bmatrix} \\ \begin{bmatrix} A^2 r_x(0) & A^2\mathbf{r_x}^H \\ A^2\mathbf{r_x} & A^2\mathbf{R_x} \end{bmatrix} \begin{bmatrix} 1 \\ -\mathbf{w}_y \end{bmatrix} &= \begin{bmatrix} P_{My} \\ \mathbf{0} \end{bmatrix} \\ \begin{bmatrix} r_x(0) & \mathbf{r_x}^H \\ \mathbf{r_x} & \mathbf{R_x} \end{bmatrix} \begin{bmatrix} 1 \\ -\mathbf{w}_y \end{bmatrix} &= \begin{bmatrix} \frac{P_{My}}{A^2} \\ \mathbf{0} \end{bmatrix}. \end{aligned} \tag{6.22}$$

Thus the LP coefficients of $y(n)$ will be the same as the LP coefficients of the transmitted signal $x(n)$. The minimum mean-squared error (MMSE), P_{M_y}, of $y(n)$ differs from the MMSE of

Stellenbosch University http://scholar.sun.ac.za

$x(n)$ by only a constant factor of A^2. The MMSE is used only to calculate the zero'th order cepstral coefficient. The zero'th order cepstral coefficient is therefore the only component that will be modified and will become

$$c_y(0) = \ln \sqrt{A^2 P_{M_x}}. \tag{6.23}$$

Figure 6.12 shows an example of the LP cepstrum of $y(n)$ with a fading constant of $A = 0.5$.

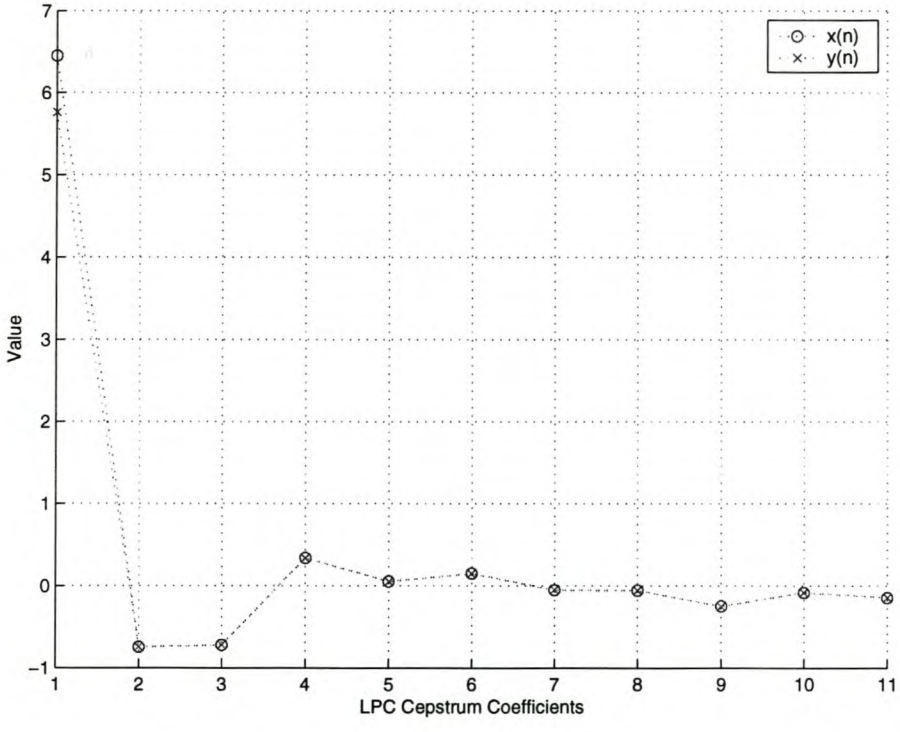

Figure 6.12: *The LP cepstrum of a frame of speech undergoing ionospheric fading.*

6.3.3 Multi-path interference

Multi-path signals results when the transmitted signal is reflected from more than one point in the ionosphere. Each reflected signal travels a different path, undergoes a different absorption and arrives at the receiver at different times. The multiple versions of the transmitted signal cause the received signal to be dispersed in time. To investigate the effect of multi-path signals we examine a simplified case with only one delayed signal,

$$y(n) = A_0 x(n) + A_1 x(n-k), \tag{6.24}$$

Stellenbosch University http://scholar.sun.ac.za

where k is the delay in samples and where A_0 and A_1 are the absorption constants of the original and delayed signals respectively.

We first determine how ionospheric absorption will affect the correlation domain and then use this result to calculate the LP coefficients.

The autocorrelation of $y(n)$ is

$$\begin{aligned}
r_{yy}(n) &= y(n) * y(-n) \\
S_{yy}(\omega) &= Y(\omega) \cdot Y^*(\omega) \\
&= [A_0 X(\omega) + A_1 X(\omega) e^{-j\omega k}] \cdot [A_0^* X^*(\omega) + A_1^* X^*(\omega) e^{+j\omega k}] \\
&= A_0 A_0^* X(\omega) X^*(\omega) + A_1 A_0^* X(\omega) X^*(\omega) e^{-j\omega k} \\
&\quad + A_0 A_1^* X(\omega) X^*(\omega) e^{+j\omega k} + A_1 A_1^* X(\omega) X^*(\omega) e^{j\omega(k-k)} \\
&= X(\omega) X^*(\omega) [A_0^2 + A_0 A_1 e^{-j\omega k} + A_0 A_1 e^{+j\omega k} + A_1^2] \\
&= (A_0^2 + A_1^2)\, S_{xx}(\omega) + A_0 A_1\, (S_{xx}(\omega) e^{-j\omega k} + S_{xx}(\omega) e^{+j\omega k}) \\
r_{yy}(n) &= (A_0^2 + A_1^2)\, r_{xx}(n) + A_0 A_1\, (r_{xx}(n-k) + r_{xx}(n+k)).
\end{aligned} \tag{6.25}$$

The autocorrelation of the received signal is a combination of the autocorrelation of the original transmitted signal and shifted versions of this autocorrelation. However, since no prior information about the transmitted signal is available it would be difficult to extract the original correlation term from the resulting one. The resulting correlation function can be very similar to the original one for small values of A_1, but the function quickly becomes unrecognisable from the original one if A_0 and A_1 are of the same order of magnitude and if k is large. In this case the LP coefficients and LP cepstrum of the original and multi-path signals will also differ significantly. Figure 6.13 shows the LP coefficients of a multi-path speech signal with one delayed signal. For both cases the delay is $k = 10$ ($625\mu s$) and $A_0 = 1$. In the first example $A_1 = 0.25$, while for the second example $A_1 = 1$. Multi-path signals can therefore seriously deform the LP coefficients and LP cepstrum if they are present.

Stellenbosch University http://scholar.sun.ac.za

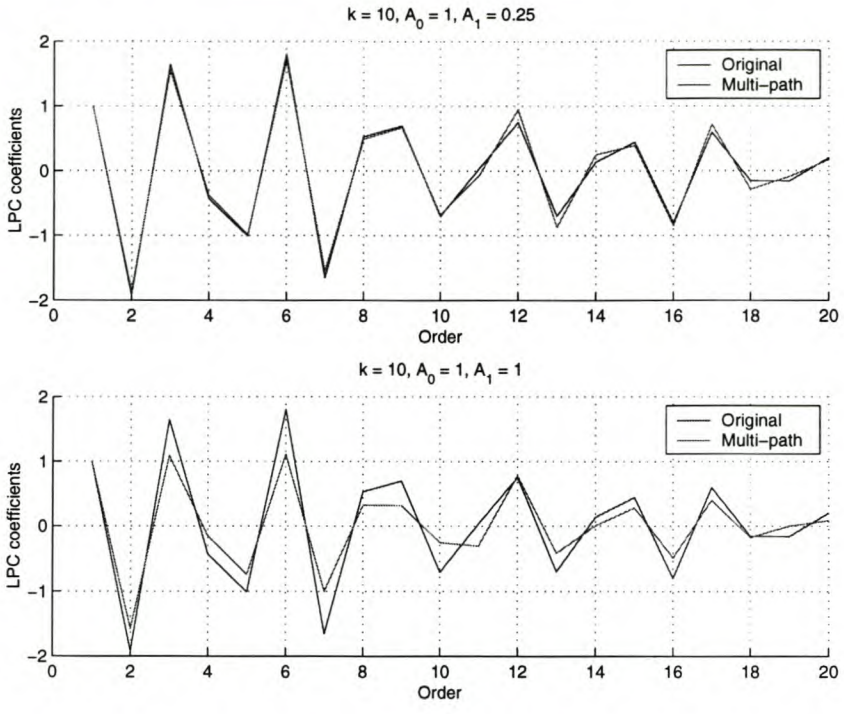

Figure 6.13: *Effect of multi-path signals on the LP coefficients: (a) Amplitude of the delayed signal is four times smaller than the amplitude of the original signal, (b) Amplitude of the original and delayed signal is of the same order.*

6.3.4 Doppler shifts

The point of reflection of radio waves from the ionosphere may move vertically due to day-night temperature variations, winds, electric fields, solar flares, geomagnetic storms and ionospheric storms [85, p.293-303]. When the point of reflection from the ionosphere change, the path length will change. The transmitted radio wave will then experience a Doppler shift. The Doppler shift will change the frequency of the reflected wave, thereby causing it to spread over different frequencies. This phenomenon is called frequency dispersion. A Doppler frequency shift is defined as

$$\Delta\omega = \omega[(\frac{v + v_o}{v - v_s}) - 1], \tag{6.26}$$

where ω is the frequency component that experiences the Doppler shift, v is the speed of the radio wave (speed of light), v_0 is the speed at which the observer (receiver) moves and v_s is the speed at which the source (transmitter) moves. Since the source and observer are stationary we take v_o and v_s as the speed relative to the moving reflection point. From equation (6.26)

Stellenbosch University http://scholar.sun.ac.za

we observe that the Doppler shift is nothing but a scaling of the frequency spectrum. However, since the bandwidth of the sidebands are small compared to the carrier frequency, $\Delta\omega$ at the lowest point in the band will be almost the same as $\Delta\omega$ at the highest point in the band. To illustrate this point we consider an example: Let's assume that the point of reflection change at a rate of 100 km per 12 h (day-night variation), the transmitted signal has a bandwidth of 4 kHz and a modulation frequency of $\omega_c = 10$ MHz. The speed at which the observer and the source move is calculated as

$$v_o = v_s = \frac{100\ km}{12\ h} = 2.315\ m.s^{-1}. \tag{6.27}$$

The Doppler shift at the carrier frequency is then

$$\Delta\omega_1 = 10.10^6\ [\frac{300.10^3 + 2.3148}{300.10^3 - 2.3148} - 1] = 154.3221\ Hz \tag{6.28}$$

and the Doppler shift at the upper side of the signal band (assuming an upper sideband signal) is

$$\Delta\omega_2 = (10.10^6 + 4.10^3)\ [\frac{300.10^3 + 2.3148}{300.10^3 - 2.3148} - 1] = 154.3838\ Hz. \tag{6.29}$$

The difference between $\Delta\omega_1$ and $\Delta\omega_2$ is so small that the scaling can be seen as a localised frequency shift. Therefore, after demodulation the Doppler shift appears to be a frequency shift of the sidebands. To demonstrate this we start with the modulated transmitted signal as defined in equation (6.2)

$$y_{\mp}(t) = x(t)\cos(\omega_c t) \pm \widehat{x}(t)\sin(\omega_c t), \tag{6.30}$$

where ω_c is the carrier frequency. Now add the Doppler shift, $\Delta\omega$

$$y_{\mp}(t) = x(t)\cos((\omega_c + \Delta\omega)t) \pm \widehat{x}(t)\sin((\omega_c + \Delta\omega)t) \tag{6.31}$$

and demodulate

$$\begin{aligned} y_{\mp}(t) =& [x(t)\cos((\omega_c + \Delta\omega)t) \pm \widehat{x}(t)\sin((\omega_c + \Delta\omega)t)] \cdot \cos(\omega_c t) \\ =& \frac{1}{2}x(t)[\cos((2\omega_c + \Delta\omega)t) + \cos(\Delta\omega t)] \\ & \pm \frac{1}{2}\widehat{x}(t)[\sin((2\omega_c + \Delta\omega)t) + \sin(\Delta\omega t)]. \end{aligned} \tag{6.32}$$

Finally filter out the double frequency components

$$y_{\mp}(t) = \frac{1}{2}[x(t)\cos(\Delta\omega t) \pm \widehat{x}(t)\sin(\Delta\omega t)]. \tag{6.33}$$

Stellenbosch University http://scholar.sun.ac.za

Comparing equation (6.33) and equation (6.16) we find that they are exactly the same. A Doppler shift in the transmission channel has the same effect as a frequency error in the demodulator. We have already discussed the effects of a frequency shift in the sidebands in Section 6.2.3 and refer to that section for details.

6.3.5 Summary of HF channel

In this section the HF channel was introduced and three of the main effects that can distort the transmitted signal were described.

The first of these effects, ionospheric absorption, is caused by the absorption of the transmitted signal by the ionosphere. Ionospheric absorption only change the zero'th order cepstral coefficient, while keeping the other components unmodified. Ionospheric absorption is therefore not a big concern in practical applications where the zero'th order cepstral coefficient is frequently discarded.

The second effect, multi-path signals, occur as a result of the transmitted signal being reflected by more than one point in the ionosphere. Each reflected signal travels its own path, and will consequently have a different absorption, and will arrive at the receiver at different times. The resulting signal consists of the transmitted signal plus one or more delayed versions of the original signal. The effect of the multi-path signals on the LP cepstrum depends on the relative amplitude of the original signal and the time delay between the signals. If the delayed signals are of a much smaller amplitude than the original signal the LP cepstrum will experience only minor deformation, otherwise these multi-path signals can have a major effect on the LP coefficients and the LP cepstrum.

The third effect, Doppler shift, is caused by the vertical movement of the ionosphere. This movement causes the transmitted signal to undergo a Doppler shift in frequency. The Doppler shift effectively scale each frequency component of the signal. However, since the bandwidth of the transmitted signal is very small compared to the carrier frequency, the frequency scaling of the lowest and highest frequency points in the sidebands is almost the same. The frequency scaling can therefore be seen as a frequency shift. It was shown that after demodulation the

Stellenbosch University http://scholar.sun.ac.za

Doppler shift produces the same equation as when a frequency error is present during demodulation. This means that the Doppler shift will shift the sidebands towards or away from each other. For small Doppler shifts the LP cepstrum will experience only minor deformation while large Doppler shifts can deform the LP cepstrum dramatically.

Of these three effects only multi-path signals and Doppler shifts have the potential to deform the LP cepstrum drastically. The Doppler shift only becomes a problem if large shifts are experienced.

6.4 Noise

During transmission the radio signals are contaminated by various noise sources. These include the electronics of the transmitter/receiver and background noise added by the environment. It is usually assumed that the noise in radio signals is additive, uncorrelated, Gaussian white noise. This type of noise is additive in the time, frequency and correlation domain, but is non-linear in the logarithmic domain, such as the cepstral domain. The additive nature of this type of noise has lead to most of the speech enhancement methods mentioned in Section 1.3.1.

To investigate the effects of noise we first define a signal contaminated by additive Gaussian white noise

$$y(n) = x(n) + \eta(n), \tag{6.34}$$

where $x(n)$ is the original clean speech signal, $\eta(n)$ is the added noise component and $y(n)$ is the noisy speech signal.

We first determine how ionospheric absorption will affect the correlation domain and then use this result to calculate the LP coefficients.

Stellenbosch University http://scholar.sun.ac.za

The autocorrelation of $y(n)$ is

$$\begin{aligned} r_{yy}(n) &= y(n) * y(-n) \\ S_{yy}(\omega) &= Y(\omega) \cdot Y^*(\omega) \\ &= [X(\omega) + N(\omega)] \cdot [X^*(\omega) + N^*(\omega)] \\ &= X(\omega)X^*(\omega) + N(\omega)X^*(\omega) + X(\omega)N^*(\omega) + N(\omega)N^*(\omega). \end{aligned} \tag{6.35}$$

Since the speech and the noise signals are uncorrelated, we can discard the cross-correlation terms, so that

$$\begin{aligned} S_{yy}(\omega) &= X(\omega)X^*(\omega) + N(\omega)N^*(\omega) \\ &= S_{xx}(\omega) + S_{\eta\eta}(\omega) \\ r_{yy}(n) &= r_{xx}(n) + r_{\eta\eta}(n). \end{aligned} \tag{6.36}$$

The autocorrelation of the noisy signal is the sum of the autocorrelation of the original signal and the autocorrelation of the noise signal. This confirms that the noise is additive in the correlation domain.

Next the result of equation (6.36) is used to determine the correlations in the augmented Wiener-Hopf equation (6.1)

$$\begin{aligned} \begin{bmatrix} r_y(0) & \mathbf{r_y}^H \\ \mathbf{r_y} & \mathbf{R_y} \end{bmatrix} \begin{bmatrix} 1 \\ -\mathbf{w}_y \end{bmatrix} &= \begin{bmatrix} P_{My} \\ \mathbf{0} \end{bmatrix} \\ \begin{bmatrix} (r_x(0) + r_n(0)) & (\mathbf{r_x} + \mathbf{r_n})^H \\ (\mathbf{r_x} + \mathbf{r}_\eta) & (\mathbf{R_x} + \mathbf{R}_\eta) \end{bmatrix} \begin{bmatrix} 1 \\ -\mathbf{w}_y \end{bmatrix} &= \begin{bmatrix} P_{My} \\ \mathbf{0} \end{bmatrix}. \end{aligned} \tag{6.37}$$

Unfortunately the LP coefficients obtained by solving equation (6.37) can not be written in a form that clearly separates the contribution of noise and the speech signal to the LP coefficients. Figure 6.14 demonstrates what happens to the LP coefficients when different levels of noise are added to the system. The LP coefficients are calculated from speaker *fcj0*, file *sx384* in the TIMIT database. For all four cases the same clean speech was used but the signal was contaminated by 40, 25, 10 and 0 dB noise respectively. In this example the first LP coefficient component is plotted against the second LP coefficient component. The effect can be demonstrated by using any other combination. As the SNR increases the LP coefficients of the noisy speech seem to contract to a smaller and smaller region. For pure random Gaussian noise this

Stellenbosch University http://scholar.sun.ac.za

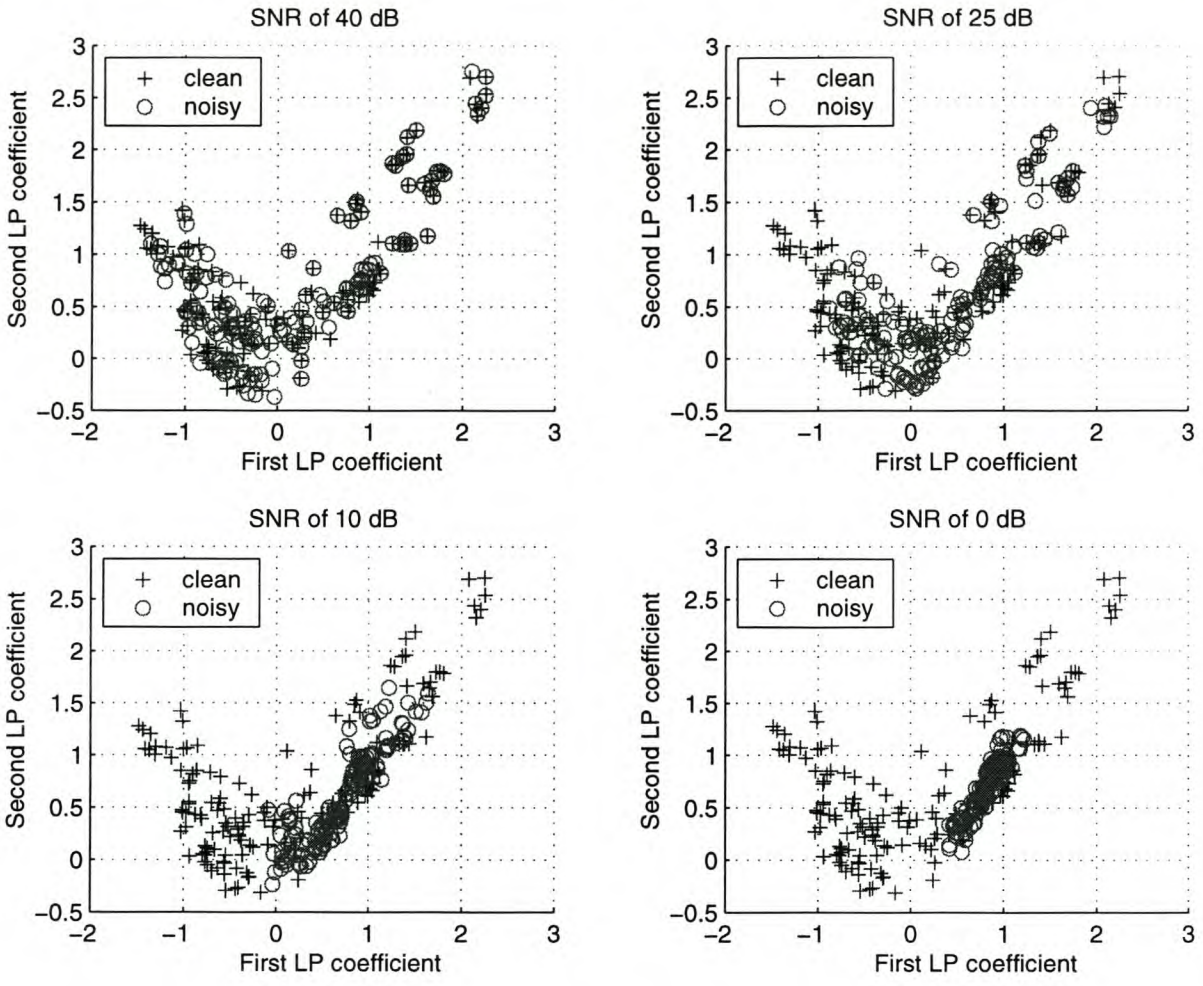

Figure 6.14: *The difference between the LP coefficients of clean and noisy signals for different SNR ratios.*

region is centred at the origin. This effect can be seen in Figure 6.15. The mean of all the LP coefficients (except the zero'th component) of all the sequences plotted in Figure 6.15 were approximately zero. This can be confirmed by the transfer function of an AR process given by

$$H(\omega) = \frac{1}{\sum_{n=0}^{N} a_n z^{-n}}, \tag{6.38}$$

where a_n is the n'th LP coefficient and N is the order of the AR process and $a_0 = 1$. If we now substitute $a_n = 0$ for $n \neq 0$ this equation simplifies to unity and we are left with a flat spectrum. This is what we would expect from a Gaussian white noise source.

Another effect that is not clearly shown in Figure 6.14, is that the trajectory of an LP coefficient, traced out as the SNR decrease, is not linear. This is shown in Figure 6.16. The trajectories are constructed by plotting the same LP coefficient for different noise levels and connecting the

Stellenbosch University http://scholar.sun.ac.za

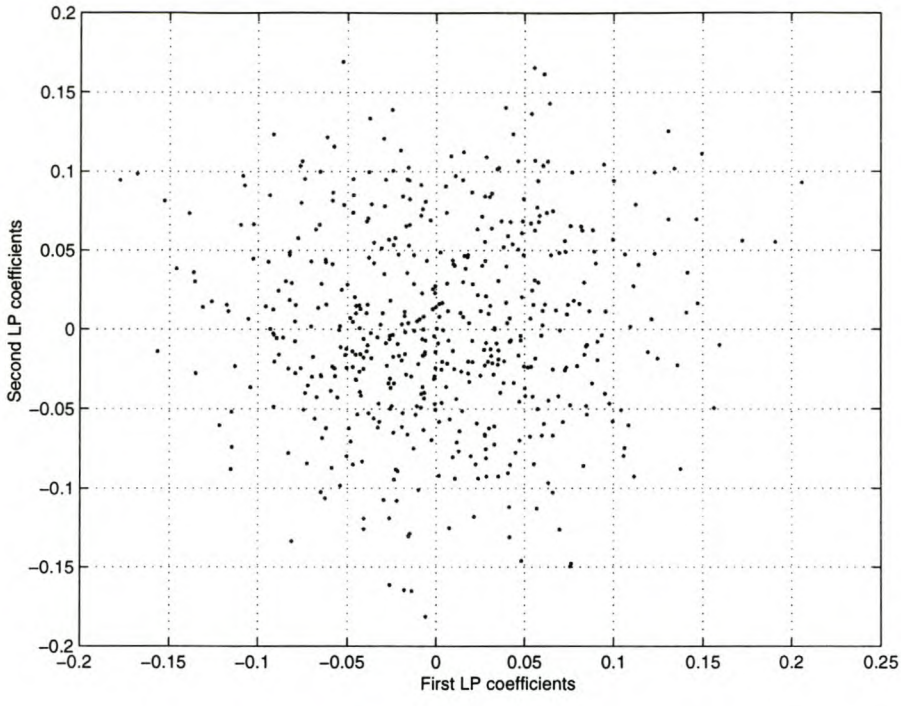

Figure 6.15: *The LP coefficients of Gaussian noise.*

path with a line. It is clear that the trajectories of the LP coefficients are non-linear. This non-linear behaviour combined with the fact that the LP coefficients move closer to the origin, makes a one-to-one reverse mapping to remove the noise very difficult if not impossible. Such a mapping would attempt to transform the LP coefficients of the noisy speech back to the LP coefficients of the clean speech if the amount of noise contamination is known.

In a similar manner the LP cepstrum will be transformed to cluster around the origin as the noise in the signal increases. The result is that we end up with LP cepstral coefficients where the contribution of the signal and noise are not separable. The non-linear nature of the cepstrum makes it very difficult to remove the effects of the noise on the cepstrum. For this reason most enhancements are done in the spectral or time domain where the noise is additive.

Stellenbosch University http://scholar.sun.ac.za

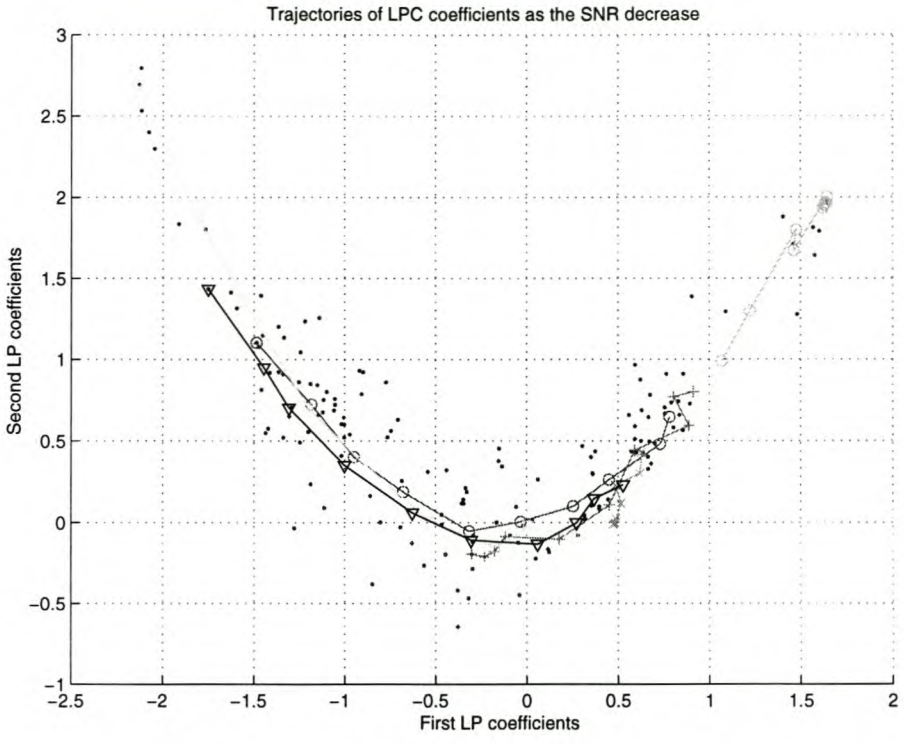

Figure 6.16: *Trajectories of a few selective LP coefficients as the SNR decrease.*

6.5 Conclusion

In this chapter we investigated the effects produced when speech is transmitted over an HF channel. We have considered the effects produced by both the SSB modulation system and the HF channel. These effects were analysed to determine how they deform the LP coefficients and LP cepstrum. The effects can be placed in three categories: The first category is the effects produced by the SSB modulation system. The second category is the effects produced by the HF channel and the third category is the effects produced by noise.

In the first category, SSB modulation, we investigated the effects of a phase and frequency error in the modulator and demodulator. If only a phase error is present, it was shown that only the zero'th order cepstral coefficient changed while the other coefficients stayed unmodified. The presence of a frequency error in the modulator or demodulator causes the spectrum of the sidebands to shift towards or away from each other. The shift of the sidebands can cause aliasing around the zero frequency component or the Nyquist frequency. If the frequency error is small only minor deformation of the LP coefficients and LP cepstrum is experienced. Since

Stellenbosch University http://scholar.sun.ac.za

we expect the modulation system to minimise these errors, the phase and frequency error in SSB modulation will only cause minor deformations of the LP cepstrum.

In both these cases the Hilbert transform of the signal is encountered. In our analysis an ideal Hilbert transform was assumed. However, it is not possible to construct an ideal Hilbert transform and consequently the non-ideal Hilbert transform can introduce minor deformation of the LP coefficients and LP cepstrum.

In the second category the effects of the HF channel were investigated. The first effect investigated was ionospheric absorption. It was shown that this effect only modifies the zero'th order cepstral coefficient. The second effect was multi-path signals. The transmitted signal can be reflected from more than one point in the ionosphere. The result is that the transmitted signal and one or more delayed versions of the signal arrive at the receiver. If the amplitudes of the delayed signals are small compared to the original signal the effect on the LP coefficients and LP cepstrum is small. However, for large amplitude differences between the original and delayed signals, the deformations will produce very different LP coefficients. The third effect was Doppler shifts which are caused by the vertical movement of the ionosphere. A Doppler shift causes a scaling of the frequency spectrum. It was shown that for narrow band signals around a large carrier frequency the scaling effect can be considered as a frequency shift. The Doppler shift produces the same effect as a frequency error in the modulator or demodulator.

In the third category we considered additive Gaussian white noise. This type of noise is additive in the time and frequency domain, but non-linear in the logarithmic domain. It was shown that the effect of noise on the LP coefficients and LP cepstrum depends on both the speech and noise in a non-linear fashion. The larger the noise component the more distorted the LP cepstrum will be. As the SNR decrease, the LP cepstral components becomes approximately zero. Even relatively small noise values can deform the LP cepstrum dramatically. Combatting the effect of noise in the cepstral domain in not viable due to the non-linear effect that the additive noise has in the log domain. Attempts to remove the effect of noise is usually made in the time or frequency domain where it is additive.

In all these cases the main cause of deformations are multi-path signals, large frequency errors, large Doppler shifts and noise. A thorough investigation to determine to what extent each of

Stellenbosch University http://scholar.sun.ac.za

these effects are present in a typical system can be conducted. Such an investigation requires the availability of a practical modulation system. Various experiments can then be conducted to examine each effect in isolation and determine the typical parameters encountered in a practical system. Such experiments will attempt to determine the phase and frequency errors in the SSB modulation system, typical absorption levels, Doppler shifts, multi-path signals and noise levels. With all these statistics available the effect that contributes most to the deformation of the LP cepstrum can be determined. An attempt can then be made to eliminate, or at least subdue, the deformations.

This analysis was not performed in this study. Instead we decided to concentrate on the noise in the system. In order to minimise the effects of noise on the system a method to enhance speech in the presence of additive Gaussian white noise was formulated. By focusing on removing noise from speech signals, we can apply the knowledge to a much wider set of applications. This speech enhancement system will be described in the next section.

Stellenbosch University http://scholar.sun.ac.za

Chapter 7

Speech enhancement using HMM

7.1 Introduction

After analysing the various effects introduced by the HF channel and SSB modulation we decided to investigate speech enhancement (SE) techniques. The reason for this is that when you listen to some of the speech transmitted over the HF channel, it is clear that very high levels of noise is present in the received signals. Our analysis (Section 6.4) also indicates the dominant role that noise plays in the degradation of HF speech. Furthermore, any advancements in this field can be applied to a much wider range of applications than just the HF transmitted speech. A short introduction to some speech enhancements techniques were presented in Section 1.3.1.

A common mathematical model for noise in a communication system is the additive noise channel. The transmitted signal $x(n)$ is corrupted by an additive random noise process $\eta(n)$

$$y(n) = x(n) + \eta(n), \tag{7.1}$$

where $y(n)$ is the resulting noise contaminated signal. It is also assumed that the noise samples are uncorrelated. The noise is statistically classified as Gaussian if the noise is generated by a Gaussian random process. Finally, noise is called white if its power density spectrum is constant over all frequency components. A random noise process with all these characteristics is called an additive uncorrelated Gaussian white noise process.

Stellenbosch University http://scholar.sun.ac.za

In the frequency domain equation (7.1) becomes

$$Y(\omega) = X(\omega) + N(\omega). \tag{7.2}$$

The fact that the noise component is additive in the frequency domain has led to speech enhancement methods such as spectral subtraction [16]. This method subtracts an estimate of the noise power spectrum from the power spectrum of $y(n)$. The noise statistics are estimated during silent periods in the signal. While this technique is fairly simple it can introduce negative spectral values. In order to remove this effect the negative values are usually set to zero or some small positive value. This ad hoc correction procedure produces musical-noise in the speech [7, p.510].

The technique we described in this study is based on statistical analysis of the speech and noise source. A number of techniques employing statistical modelling were mentioned in section 1.3.1.3. The majority of these methods attempt to iteratively estimate some parameters of the speech signal and then construct a Wiener or Kalman filter to filter the noisy signal [22, 23, 24, 25, 26]. These methods use the noisy speech to estimate the statistics of the clean speech and the noise. The problem is that the noisy signal does not contain enough information to estimate accurate models for both the speech and the noise [27]. The problem can be solved by developing speech models from clean speech alone. Hidden Markov models have been successfully used to model the clean speech [28, 29, 30, 31].

The speech enhancement system we developed uses clean speech to train an HMM that represent the speech process. The underlying probability density functions (PDF) of the states are composed so that the time dependent information is incorporated. The time information is obtained by constructing a feature vector that consists of a number of sequential speech samples. It is also possible to incorporate more time information in the model by increasing the order of the HMM. The advantage of an HMM, compared to a plain PDF-based classifier (which we also investigated), is to place constraints on the possible links between the states. It therefore limits the paths that a clean signal can follow through the model. Thus, if the process is in a known state, there is only a limited number of states to which the process can change to. Our speech enhancement system does not require a noise model. Rather, during the speech enhancement process the underlying PDFs are adapted to include the noise variance (Section 7.4.8).

Stellenbosch University http://scholar.sun.ac.za

In the next section we will offer a short introduction to hidden Markov models. This includes a method to construct and train higher order models. In Section 7.4 we then describe our speech enhancement system in detail. Finally Section 7.5 reflects on the advantages and disadvantages of the system.

7.2 Hidden Markov models

Hidden Markov models have been applied to many speech processing problems such as speech, speaker, language and phoneme recognition. In the next section we will define an HMM and provide algorithms to train and apply these models.

7.2.1 Definition

Before describing HMMs we first introduce Markov chains (MC). Consider a system that consists of N unique states. We denote these states: $S_1, S_2, \ldots, S_N$. At regular discrete time intervals the system can change from one state to another with a certain probability. At any given time the current state of the system is denoted by q_n, for $n = 1, 2, \ldots, T$. The order of an MC indicates how many previous states were used to describe transitions in the system. For example, a first order system only depends on the current state to make a transition to the next, while a second order system depends on the current and previous state to make a transition to the next state. For a first-order system the probability of going from the current state i to the next state j is the transition probability

$$a_{ij} = P(q_n = S_j | q_{n-1} = S_i), \qquad \text{for } 1 \leq i \text{ and } j \leq N. \tag{7.3}$$

The state transition probabilities are always positive,

$$a_{ij} \geq 0, \tag{7.4}$$

and the probabilities of the paths leaving a state all sum to unity,

$$\sum_{j=1}^{N} a_{ij} = 1. \tag{7.5}$$

Stellenbosch University http://scholar.sun.ac.za

The transition probabilities are written in matrix form as

$$\mathbf{A} = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1j} \\ a_{21} & a_{22} & \cdots & a_{2j} \\ \vdots & \vdots & \ddots & \vdots \\ a_{i1} & a_{2i} & \cdots & a_{ij} \end{bmatrix}. \tag{7.6}$$

The process described by an MC is said to be an observable Markov model. This is because the output of the model and the states at each moment in time is observable, since each state corresponds to an observable event. We can define an observation sequence $\mathbf{O}$, where $\mathbf{O} = \{O_1, O_2, \ldots, O_K\}$. Each observation corresponds to one of the discrete states, for example $\mathbf{O} = \{S_5, S_1, S_3, S_3, S_4, S_2\}$. The probability that the observed signal was produced by a given model, λ is given by

$$\begin{aligned} P(\mathbf{O}|\lambda) =& P(O_1, O_2, \ldots, O_K|\lambda) \\ =& P(S_5, S_1, S_3, S_3, S_4, S_2|\lambda) \\ =& P(S_5) \cdot P(S_1|S_5) \cdot P(S_3|S_1) \cdot P(S_3|S_3) \cdot P(S_4|S_3) \cdot P(S_2|S_4) \\ =& \pi_5 \cdot a_{51} \cdot a_{13} \cdot a_{33} \cdot a_{34} \cdot a_{42}, \end{aligned} \tag{7.7}$$

where π_i denote the initial state probabilities, which is given by

$$\pi_i = P(q_1 = S_i), \qquad \text{for } 1 \leq i \leq N. \tag{7.8}$$

All the initial state probabilities can be placed in an initial state vector $\boldsymbol{\pi}$.

We can extend the idea of the MC by replacing the concrete observable states with probabilistic functions. The resulting model is called a hidden Markov model, because the underlying stochastic process is hidden or not observable. They can only be observed through another set of stochastic processes that produce the observations.

In addition to the definitions that describe an MC, the HMM can also be characterised by the number of distinct observation symbols, M, per state. The observation symbols correspond to the physical output of the system. The symbols are denoted by $\mathbf{V} = \{v_1, v_2, \ldots, v_M\}$. We define the observation symbol probability distribution in state j as

$$b_j(k) = P(v_k|q_t = S_j), \qquad \text{for } 1 \leq j \leq N \text{ and } 1 \leq k \leq M. \tag{7.9}$$

Stellenbosch University http://scholar.sun.ac.za

This is the probability that v_k is observed from state S_j at time t. The collection of all the observation probabilities can be placed in the observation symbol matrix, $\mathbf{B}$.

A complete description of an HMM requires the specification of N and M, and the specification of the probability measures $\mathbf{A}$, $\mathbf{B}$ and $\boldsymbol{\pi}$. A common compact notation for an HMM is

$$\lambda = (\mathbf{A}, \mathbf{B}, \boldsymbol{\pi}). \tag{7.10}$$

It is also possible to replace the discrete density functions of the states with a continuous function such as a Gaussian PDF.

In order for the HMM to be of practical use, there are three basic problems that need to be solved. These problems are referred to as the evaluation, optimal state and re-estimation problems. They are briefly described next:

Evaluation problem: If we are given a model $\lambda = (\mathbf{A}, \mathbf{B}, \boldsymbol{\pi})$ and an observation sequence $\mathbf{O} = \{O_1, O_2, \ldots, O_T\}$, how do we compute the probability that the observation was produced by the model $P(\mathbf{O}|\lambda)$? The solution is used to determine the 'best' model, from a set of models, for an observation sequence. The solution to this problem, the forward-backward approach, which is also known as the Baum-Welch algorithm, is given in [86, p.262-263], [87] and in [7, p.686-692].

Optimal state sequence problem: In this problem we attempt to find the hidden part of the model. That is, given an observation sequence $\mathbf{O} = \{O_1, O_2, \ldots, O_T\}$, what is the state sequence $\mathbf{Q} = \{q_1, q_2, \ldots, q_T\}$ that 'best' matches the observation sequence? It should be clear that there can be more than one solution. We therefore use some optimality criteria to solve this problem. The solution to this problem, the Viterbi algorithm, is given in [86, p.263-264], [87] and in [7, p.692-699].

Re-estimation problem: In this problem we are faced with the question of re-estimating the parameters of a model $\lambda = (\mathbf{A}, \mathbf{B}, \boldsymbol{\pi})$, so that $P(\mathbf{O}|\lambda)$ is maximised. This is also called the training problem. An observation sequence is used to train a model that best represent it. The solutions to this problem, the forward-backward (Baum-Welch) re-estimation algorithm and the Viterbi re-estimation algorithm, are given in [86, p.264-266], [87] and in [7, p.699-705].

Stellenbosch University http://scholar.sun.ac.za

7.2.2 Higher order HMMs

We already stated that in a first order MC or HMM we only use the current state when making a transition to the next state. In a higher order HMM, the current and $N-1$ previous states (where N is the order of the HMM) are used to make a transition decision.

Algorithms that extend the Baum-Welch and Viterbi algorithms for second order HMMs, are available [88, 89]. The processing of these second order HMMs are computationally very expensive. This expense and the lack of training algorithms for third and higher order HMMs has limited the application of higher order HMMs.

Instead of finding algorithms for the training of higher order HMMs, it was suggested that a higher order HMM can be reduced to a first order equivalent model [90]. Reducing the order of the higher order HMMs makes it theoretically possible to train these models. However, the practical application of this technique is limited by the lack of computer memory and processing constraints. Therefore, higher order HMMs with many states can still take considerable time to train due to the huge number of parameters that the equivalent first order model will have. In order to alleviate this problem the fast incremental training (FIT) algorithm was provided [90, 91]. This algorithm incrementally 'grows' the higher order HMM from lower order HMMs. For example if we want to train an N'th order HMM we first train a first order HMM. The first order HMM is then extended to a second order HMM, which in turn is reduced to an equavalent first order HMM. This model is then trained. The process is repeated $N-1$ times until an N'th order equivalent HMM is reached. The advantages of this system is that redundant transitional probabilities and states are removed during training before the next order HMM is created. This approach is not equivalent to reducing an N'th order HMM to a first order HMM straight away, the reason being that the optimisation process can converge to a different local optimum during training. It was shown in [90] that this technique is considerably faster and leads to smaller and more accurate models than using the straight order reduction approach.

In our speech enhancement system we used the FIT approach to obtain higher order HMMs. Since we exploited the time dependent nature of speech, the higher order HMMs will enable us to extract extra time dependent information from the speech process.

Stellenbosch University http://scholar.sun.ac.za

7.3 An existing SE technique using HMMs

A number of existing speech enhancement techniques using HMMs are described in [28, 29, 30, 31, 32]. We will briefly describe the technique shown in [29].

Their approach is based on statistical modelling of the speech and noise processes. An HMM with autoregressive (AR) mixture Gaussian output probability distribution functions is used to model the clean speech process. The noise model depends on the type of noise considered. In their study, only Gaussian white noise was used. For this noise type a low order Gaussian AR model is used to represent the noise. The models are trained using the Baum [92] re-estimation or the EM algorithm. Enhancement of the noise contaminated speech is achieved by using the EM algorithm. A more efficient approximation of the training and enhancement procedures are also described.

Model construction and training

The clean speech process is modelled by an autoregressive mixture Gaussian HMM. The model parameters are obtained through the maximum likelihood (ML) approach, using the Baum or the EM re-estimation algorithms. These algorithms locally maximise the PDF of the model observation sequence for the clean speech training data. An efficient approximation of the Baum re-estimation algorithm, the segmental k-means algorithm [93], is also used to train the clean speech models. This algorithm locally maximises the joint PDF of the state and observation sequences of the HMM for the training data. The algorithm iteratively maximises the joint PDF over all the state sequences for a given parameter set and then maximises over all the parameter sets, given the most likely state sequence. The most likely state sequence is obtained using the Viterbi algorithm, while the estimation of the model parameter set is done with the Baum re-estimation formulas.

The Gaussian white noise process is modelled by a low order Gaussian AR model. The noise model parameters are obtained by AR modelling of the centroid covariance matrix of the noise training data.

Stellenbosch University http://scholar.sun.ac.za

Speech enhancement

For a set of of speech and noise models, speech enhancement is performed using a MAP estimation process. MAP speech enhancement is achieved through the application of the EM algorithm, which estimates the clean speech signal from a given noisy speech signal. The EM algorithm locally maximises the conditional PDF of the clean speech signal for the given noisy speech signal. The algorithm starts with an estimate of the clean speech signal (the initial estimate is the noisy speech signal) and iteratively estimates a new clean speech signal by maximising a specified auxiliary function.

A more efficient approximation to the MAP approach was also developed. In the approximate MAP approach, the algorithm locally maximises the joint conditional PDF of the state and observation sequences of the clean speech signal for a given noisy speech signal. This is done by alternatively maximising the conditional PDF over all the state sequences, while assuming that the clean speech vectors are given, and then maximising over the speech vectors while assuming that the most likely state sequence is given. The estimate of the most likely state sequence is obtained by applying the Viterbi algorithm on the current estimate of the clean speech signal. The next clean speech estimate is then obtained by filtering the noisy speech signal with a time varying Wiener filter. The Wiener filters are constructed from the autoregressive covariance matrices of the PDFs associated with the most likely state sequence of the clean speech model and the autoregressive covariance matrix of the noise model.

7.4 A new SE technique using HMMs

In this section we describe a new speech enhancement technique that uses HMMs. This new technique attempts to model the speech process on the sample level. Our efforts concentrated on the construction of clean speech models. Compensation for Gaussian white noise is done later by modifying the clean speech models.

The speech samples are used directly as features. Ideally, we would like to represent each possible quantisation level of the digitised speech by a Gaussian PDF. However, this will result

Stellenbosch University http://scholar.sun.ac.za

in models that are too large to manage. A subset of quantisation levels is selected to represent all the possible quantisation levels of the digitised speech signal. For each of these selected quantisation levels a Gaussian PDF is constructed. Since these PDFs represent all the possible sample values of the speech signal, it is possible to synthesise a speech signal by using the PDFs.

An HMM, where each of the PDFs is one state in the model, is used to model the temporal information of the speech process. Since we are working directly on the sample level, only a very short amount of time separates adjoining samples. A very high order HMM is therefore required to sufficiently model the transitional characteristics of the speech process. Unfortunately, due to the large number of states in our HMM, it is not practically possible to obtain an HMM with an order larger than two. In order to build time information into our system, we changed the feature vector. Instead of using one sample as a feature, we used a number of consecutive samples. Once again a subset of all the possible feature vectors is selected and used to construct Gaussian PDFs to represent the speech process. Each of these PDFs is then a state in an ergodic HMM with equal transition probabilities between the states. Such a model relies completely on the temporal information built into the state PDFs to model the transitional behaviour of the speech process. By training the HMM we modify the transition probabilities between states and by doing this, extract more transitional characteristics from the speech process. To further extract transitional characteristics from the model, higher order HMMs can be constructed. However, due to practical limitations only a second order HMM were created.

For speech enhancement, the clean speech models are adapted according to the type of noise we want to remove. These adapted models are then used to enhance the noisy speech. Two methods are used depending on the clean speech model. For PDF based classifiers, each feature vector of the noisy speech is presented to the model. The underlying PDF that produces the highest log likelihood score for the feature is selected. After all the feature vectors are processed, the sequence of selected PDFs are used to synthesise a new speech signal. For the HMMs, all the feature vectors of the noisy speech are presented to the model. The Viterbi algorithm is used to obtain the most likely state sequence produced by the features. This state sequence is then used to synthesise the new speech signal.

Stellenbosch University http://scholar.sun.ac.za

The following sections explain the steps in constructing clean speech models and how to use them to enhance the noisy speech in more detail.

7.4.1 Normalisation of the speech

The first step is to normalise the speech. Normalisation is performed so that all the speech data are placed in the same dynamic range. We call the method we use the robust unity power normaliser. The unity power normaliser normalises the signal so that the total power of the signal is close to unity. Since not all signals have the same duration of silence or speech, it would be difficult to obtain the desired effect of scaling to a fixed dynamic range by simply dividing by the average power in the signal. Instead the power in small segments of the speech signal is calculated and ranked. Of these ranked blocks, the power of the 85'th percentile is used to scale the whole signal. This method limits the dynamic range of the signal more accurately.

7.4.2 Partitioning of the speech

As stated earlier, it would not be feasible to represent each possible quantisation level by its own PDF or state. Doing this will result in very large, unmanageable models. If the speech is quantised in 16 bits we will need $65,536$ states and $65,536^2$ links to construct a full ergodic HMM. For our experiments we have compromised on 8 bits, 256 quantisation levels. The quantisation levels are not linearly spaced, but are spaced in a non-linear fashion similar to that used in μ-law and A-law companding. We used vector quantisation (VQ) to determine 256 clusters from all the speech data. VQ minimises the distortion between the calculated mean values and the sample data, so that we have an optimum partitioning of the speech sample space. Figure 7.1 shows the sorted mean values of the clusters, as obtained from the clean speech data. We then use each of these mean values to construct a Gaussian PDF, which is used in turn to construct a 256 state ergodic HMM. Such an HMM, when extended into a higher order HMM, becomes very large. For example a 256 state ergodic HMM has roughly 256^2 links. A second order HMM, created from this first order model, has roughly 256^2 states and roughly $256^3 \approx 2^{24}$ links. This is a very large model which we are not able to manage with current memory and

Stellenbosch University http://scholar.sun.ac.za

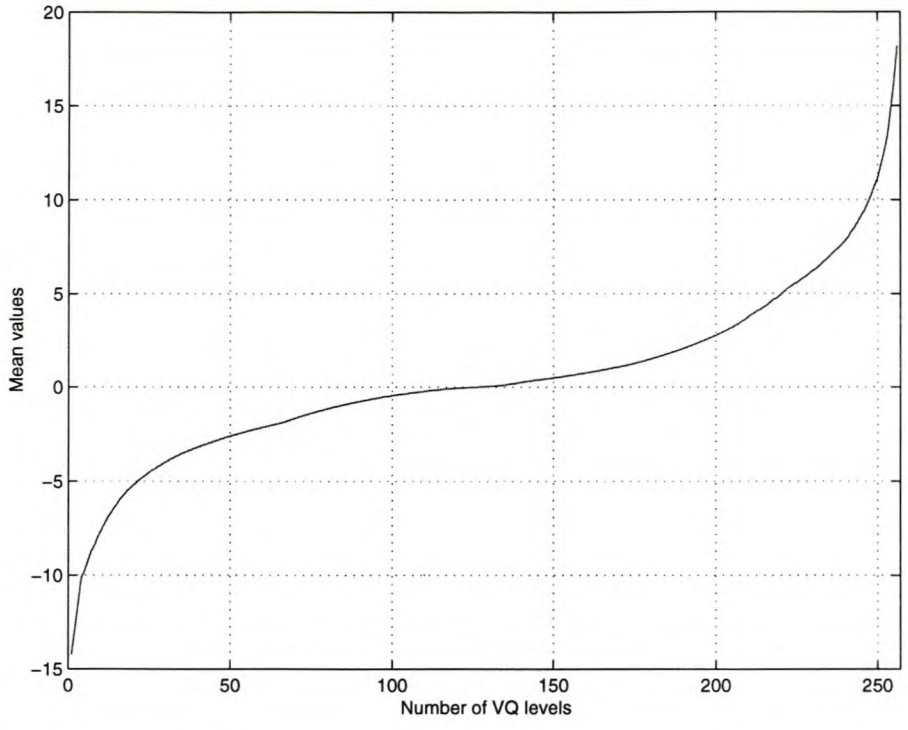

Figure 7.1: *Mean values of the clusters calculated from the clean speech data, using VQ.*

computing capabilities. Incremental training is used to simplify the first order model which is then extended to a second order model. This enables us to create a second order HMM. Creating a third order HMM is not yet possible at the time of publication.

The inability to create higher order models severely limits our capability to model the transitional information in the speech process. Our partial solution is to incorporate the time information into the feature vectors. This is done by creating a feature vector that consists of the current sample, n samples from the past and n samples from the future. This produces a feature vector of length $2n + 1$. A feature vector is just a number of speech samples centred around the current sample under analysis. Figure 7.2 shows a few examples of possible feature vectors.

Since the feature vectors are used to represent some of the temporal information, it is possible to construct PDF-based classifier models that can be used to enhance the speech. Such models are described next. The PDF-based classifier models are used to create an HMM to further increase our modelling capability.

Stellenbosch University http://scholar.sun.ac.za

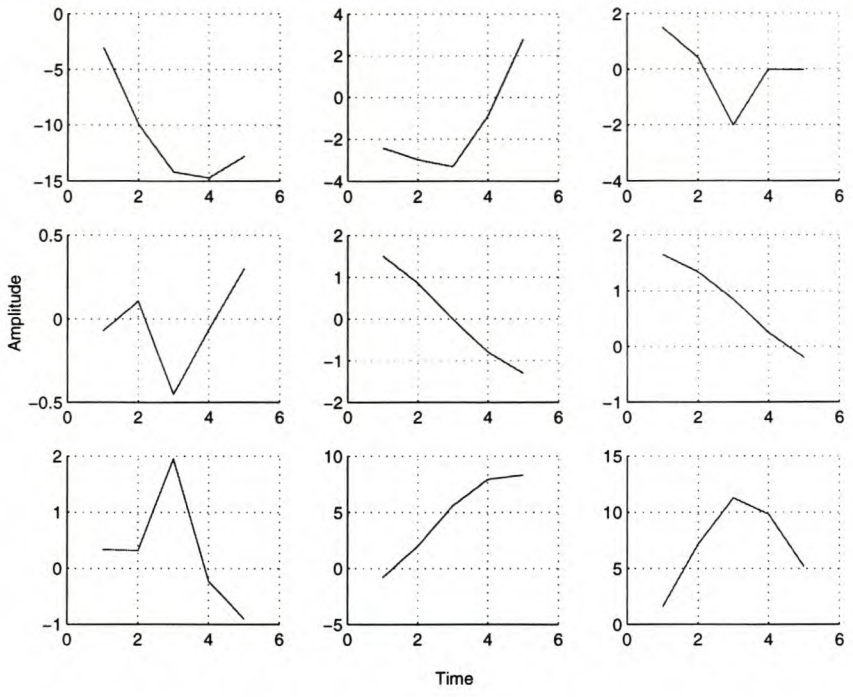

Figure 7.2: *Some possible feature vectors.*

7.4.3 PDF-based classifier with Gaussian PDFs

Our first model represents the clean speech by using the quantisation levels and different contexts that the feature vectors create around them. The quantisation levels are crude representations of the different values of a speech signal, but is not enough to represent the speech process. It is also necessary to model the time evolution or contexts of the speech process. In this first model we use Gaussian PDFs to model the variety of contexts that can occur in a clean speech signal.

Each feature vector corresponds to one of the 256 quantisation levels obtained from the clustering process in the previous section. A feature vector corresponds to one of the quantisation levels when its centre value (the sample around which the vector was constructed) is the closest to that quantisation level. There is of course a number of feature vectors corresponding to each quantisation level and each feature vector represent a different context. In order to represent all the different contexts in a compact way, we need to model some of the representative values. The feature vectors of each quantisation level is collected and clustered, using VQ [94], into M vectors. For our experiments we choose M to be 8, mainly to keep the model size small.

Stellenbosch University http://scholar.sun.ac.za

Figure 7.3 shows the clustering result for the feature vectors of one of the quantisation levels.

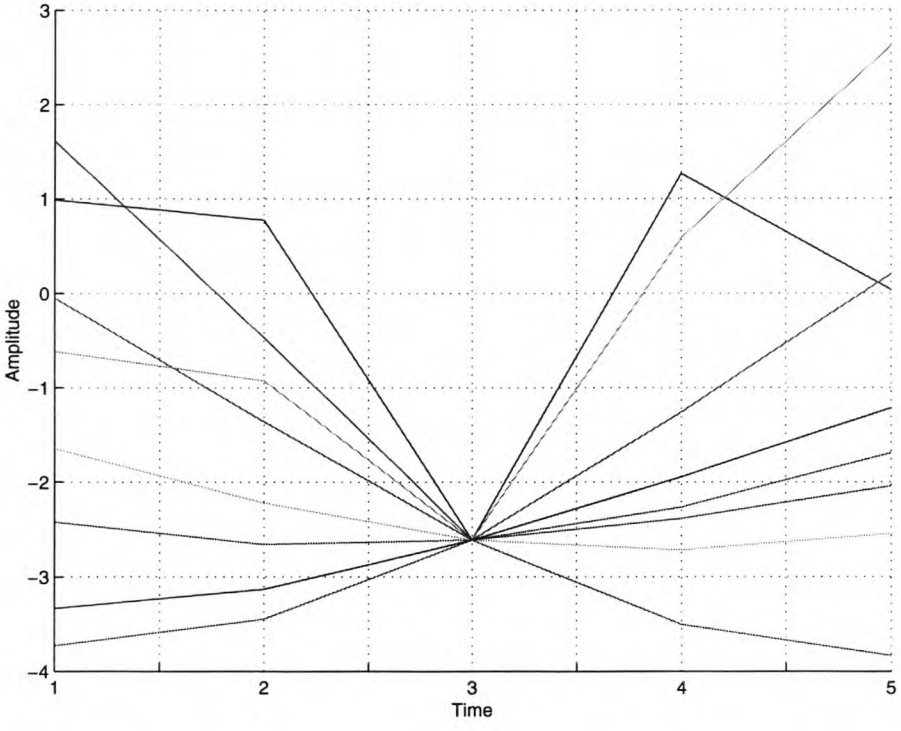

Figure 7.3: *Clustering result of one of the quantisation levels.*

Each of these vectors is then used as the mean vector of a Gaussian PDF. The covariance of the PDF can be either a full or diagonal covariance matrix. In the development stages we mostly used diagonal covariance PDFs, because they require less computational power. However, in some situations the full covariance PDFs may produce better speech enhancement results.

The Gaussian PDFs are trained to update the covariance matrices and mean vector values.

7.4.4 PDF-based classifier with Gaussian mixture PDFs

Our second clean speech model represents the speech process using a number of Gaussian mixture PDFs. The goal is to use the Gaussian PDFs to create an initial HMM to model the speech process. However, if we use all the 2048 (256x8) available PDFs of the first model, a very large HMM will result. In order to reduce the size of the resulting HMM we first collect all the Gaussian PDFs of each quantisation level in a single mixture PDF and create an HMM

Stellenbosch University http://scholar.sun.ac.za

from these mixtures.

To do this, the M Gaussian PDFs corresponding to each quantisation level are collected and used to construct a Gaussian Mixture PDF. Clean speech is used to train the Gaussian mixture PDFs. During training the mixture weights are updated and in some cases the less frequently encountered PDFs (including those inside the mixtures) are eliminated.

This model is just an intermediate step to reduce the number of PDFs in the PDF-based classifier. In the next step the PDF-based classifier is used to construct an ergodic HMM, where each state is one of the mixture PDFs of the classifier. Thus, by placing all the PDFs of each quantisation level in a Gaussian mixture, the number of states is reduced by a factor M. This mixture model can now be used to produce a much smaller and manageable HMM.

7.4.5 First order HMM

The PDF-based classifier models only make a decision on a single sample vector basis. That is, they classify each sample vector based on the likelihood scores for each PDF. In the first order HMM, the classification result depends not only on the current likelihood score, but also on that of the previous state the process was in and the likelihood of making a transition to the current state. For this reason we employ the first order HMM to extract additional temporal information from the speech process.

An ergodic first order HMM is constructed from the Gaussian mixture PDF-based classifier. Each state of the HMM is one of the Gaussian mixtures. Initially, the transitional probabilities of the ergodic HMM are all set to the same value. In such a configuration the HMM and the PDF-based classifier is mathematically identical.

HMMs are very powerful tools and given enough freedom they will exploit any aspect of the data or the structure of the model. One should therefore be careful what degree of freedom an HMM has. After the first training iteration, we noted that the state PDFs were modified considerably, resulting in poorer speech enhancement performance. These modifications are not desirable since it is important for the states to represent our original quantisation levels. In

Stellenbosch University http://scholar.sun.ac.za

addition to this, we also want the HMM to restrict the paths between the various states. To force this behaviour the state PDFs were frozen so that they can not be modified during the training process.

Another observation that we made is that in some experiments the speech enhancement algorithm performs better if all the state transitions leaving a state are set equal after training. In such a setup the HMM only restricts the possible paths between states, but does not regard one path to be be more likely than another. One possible reason for the poorer performance is that a path less likely during training will rarely be chosen during application even if it is the correct one. If an unlimited amount of training data was available this would probably not be a problem.

7.4.6 Expanding the mixtures of the first order HMM

In the second clean speech model, we grouped all the Gaussian PDFs associated with a quantisation level into a Gaussian mixture PDF. This was done to produce a smaller model to initialise the first order HMM. Each of the individual Gaussian PDFs, model one of several contexts. In the Gaussian mixture PDFs a number of these contexts are grouped together. The links of the HMM indicate possible transitions between these groups of models. Naturally we would expect that all the possible combinations of transitions between the individual contexts would not necessarily be possible or very likely. It would be ideal if each of the contexts were represented by their own state in the HMM. However, because of the large model that would result if we created a first order HMM directly from each of the Gaussian PDFs, we rather use the Gaussian mixture PDFs. During the training of the first order HMM the number of links in the model is reduced. After training we can expand the mixture model so that each of the Gaussian components form their own states. The expansion is done in such a way that all the possible transition combinations are produced. This means that no context information is lost during the intermediate step. This method and the algorithm to expand the mixtures are explained shortly.

The expansion algorithm takes a state containing a mixture PDF and expands it into a number of individual states, each containing one of the mixture PDFs.

Stellenbosch University http://scholar.sun.ac.za

Mixture expansion algorithm

Our expansion algorithm can expand the mixture PDFs in a state to states containing a single PDF or it can group some of the mixture components together to form a new state with a mixture PDF.

We first consider the case where all the mixtures are converted to states containing a single PDF state. The algorithm extracts each PDF from the mixture and creates a new state from it. All the possible links of the parent state are duplicated. The incoming links are multiplied by the mixture weight corresponding to the PDF it was created from.

In the second case subgroups of the mixture PDF are converted to a new mixture PDF state. The PDF of the new state is a mixture PDF consisting of one or more Gaussian PDFs extracted from the parent mixture. The parent links are once again duplicated, but this time the incoming links are multiplied by the sum of the mixture weights of the PDFs grouped together to form the new mixture.

The algorithm is now demonstrated with a simple example, see Figure 7.4. Part (a) of the figure shows the original HMM, where states $S1, S2, S3$ are mixture PDFs with three Gaussian PDFs. Part (b) of the figure shows the expanded equivalent of part (a). $S11, S12, S13$ are the Gaussian PDFs extracted from state $S1$. $S21, S22, S23$ are the Gaussian PDFs extracted from state $S2$ and $S31, S32, S33$ are the Gaussian PDFs extracted from state $S3$. We removed each of the underlying PDFs of the mixture and stacked them on top of each other, while preserving the possible links between the states. The link values were then updated as described above.

With this new model we gain the ability to further constrain the possible links between the quantisation levels. To illustrate this point, we will again make use of the structure shown in Figure 7.4. From the trained first order HMM we obtain the possible transitions as show in part (a) of the figure. The model is expanded to the structure shown in part (b) and is trained to produce a structure as shown in part (c). Thus, after training it is possible for the system to go only to states $S22$ and $S23$, and to none of the other states that the unexpanded HMM allowed, if it was in state $S11$. The number of possible links are therefore reduced. In our experiments we found that the number of links were reduced by roughly a factor of eight.

Stellenbosch University http://scholar.sun.ac.za

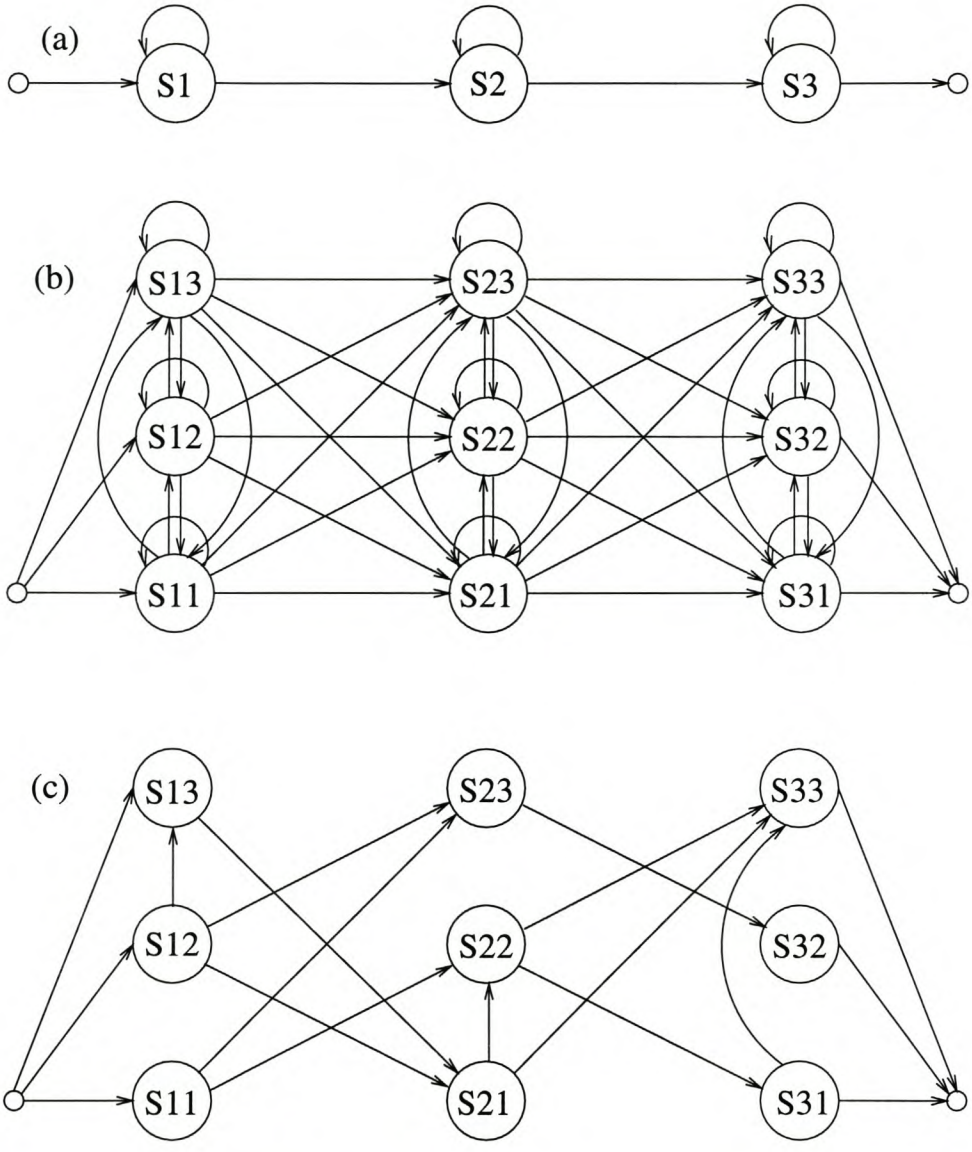

Figure 7.4: *Expanding the Gaussian mixture PDF states of an HMM to single Gaussian PDF states: (a) Original HMM with mixture PDF states, (b) Fully expanded version of the original HMM, (c) Expanded HMM after some links are eliminated during training.*

Stellenbosch University http://scholar.sun.ac.za

A modified version of the algorithm source code is provided in Appendix D.

7.4.7 A second order HMM

A second order HMM is employed in an attempt to extract even more context information from the speech process. Just like the ability of the first order HMM to constrain the number of transitions from one state to the next provide us with a more realistic speech model, we hope that the constraints imposed by the second order model can provide an even more realistic model.

The second order HMM is obtained by extending the trained first order HMM, with mixture PDF states, to a second order model. The second order model can further constrain the transitional probabilities present in the speech process. However, such a model is only effective if enough clean speech is available to train the second order model.

The second order model is very huge and takes a long time to train and to apply. Extending this model to a third order HMM or expanding the Gaussian mixture PDFs to single PDFs are not yet possible due to the large model sizes it produces.

7.4.8 Adapting the models to enhance the noisy signals

If we want to use the models to enhance noise contaminated speech, we need to adapt the model parameters to account for the noise. In our experiments we assume that only zero mean Gaussian noise contaminates the speech signals. The effect of this type of noise is to increase the variance of the speech signals. We compensated for this effect by adding the covariance of the Gaussian noise process to the covariance of the Gaussian PDFs. In order to do that we needed an estimate of the noise variance before we can apply our speech enhancement technique.

Stellenbosch University http://scholar.sun.ac.za

7.4.9 The speech enhancement algorithm

The speech enhancement algorithm we developed can take either a PDF-based classifier or an HMM as a model for the clean speech. The current algorithm can only remove zero mean white Gaussian noise, since it is very easy to adapt the clean speech models for this class of noise. The adaption is done by estimating the noise variance (possibly during silent periods of the speech signal) and adding it to the diagonal components of the covariance matrices of all the PDFs.

The system is presented with the feature vectors of a noisy speech signal. The feature vectors are constructed in the same manner as previously explained. From here onwards, the PDF-based classifier and HMM will operate differently:

- The PDF-based classifier accepts one feature vector at a time and classifies it. Classification is done by calculating the likelihood score the feature vector for each PDF. The PDF with the highest score is then selected as the one most likely to represent the feature vector. Since each PDF was constructed around one of the quantisation levels, the quantisation level corresponding to the selected PDF is the clean speech estimate of the noisy speech. The quantisation level can be obtained from the centre element of the PDF's mean vector. This process is repeated for all the feature vectors. Finally, all the clean speech estimates are concatenated to synthesise the enhanced speech signal.

- The HMM accepts all the feature vectors and uses the Viterbi algorithm to obtain an optimum state sequence through the model, where each state corresponds to one of the feature vectors. The quantisation level represented by each state is obtained in the same way as for the PDF-based classifier. The clean speech estimates are then linked, in the order that the state sequence dictates, in order to synthesise the enhanced speech signal.

The speech enhancement algorithm is very straight forward. Standard classification methods is used with only minor modifications to extract the quantisation levels from the PDFs and to construct the estimated clean speech signal.

Stellenbosch University http://scholar.sun.ac.za

7.4.10 Practical issues

The biggest problem when these models are used is that they are very big and it takes very long to train and apply them. It takes days to train a first order HMM with the number of states and links we are working with.

The Viterbi algorithm is used to train the HMMs and to determine the optimum state sequence through the model during the speech enhancement process. In order to save time and increase development speed a variant on the Viterbi algorithm, the Viterbi beam algorithm, was implemented in the later stages of the project. In the standard Viterbi algorithm all possible paths have to be remembered, while in the Viterbi beam algorithm only the paths that produced a score within a preset range from the highest score are retained. The use of this algorithm reduced the time required to train the hidden Markov models and greatly reduced the memory requirements of the Viterbi algorithm. The savings in time and memory gained from the Viterbi beam algorithm made it possible to work with the large expanded and second order HMMs. During the early stages of the development the number of links between the states had to be artificially limited in order to be able to construct and use the HMMs. However, with the introduction of the Viterbi beam algorithm it was possible to train a full ergodic HMM.

Another problem we faced was that the speech waveform fluctuate too rapidly for the HMM to properly model the speech contexts. Our system performs better if the speech waveform is smooth. The ideal way to make the speech waveform smoother is to increase the sampling frequency of the speech signal. However, since we were already working with a large quantity of data, we decided to low-pass filter the speech. This simulates the effect of an over-sampled signal. The original high quality (8 kHz) TIMIT database speech was filtered down to a 3 kHz bandwidth. This is close to the bandwidth encountered in telephone and other speech transmission channels.

Stellenbosch University http://scholar.sun.ac.za

7.5 Conclusion

In this chapter a statistical based speech enhancement algorithm was described. This technique exploits the temporal characteristics of speech process using an HMM. An HMM is a statistical model that can describe a context using the states of the HMM and the time relationship between them, using the links between the states. By choosing the states to be the speech samples, the state transitions describes the time dependency between the speech samples. By increasing the order of the HMM to N, the model models the relationship between $N+1$ consecutive samples. However, due to memory and computational constraints, it is currently not possible to create an HMM speech model with an order larger than two. This severely limits our capability to model the temporal information in the speech process. We addressed this problem by including extra time information in the feature vectors. As a side effect, speech enhancement can be done using a PDF-based classifier model. HMMs still have an edge over the PDF-based classifiers, since they can further model the interconnectivity of the speech samples.

Five basic clean speech models were developed: a Gaussian PDF-based classifier, a Gaussian mixture PDF-based classifier, a first order HMM, a second order HMM and an expanded mixture first order HMM. The performance of these models using our speech enhancement technique is evaluated in Chapter 8.

Stellenbosch University http://scholar.sun.ac.za

Chapter 8

Speech enhancement experiments

8.1 Introduction

We conducted a number of speech enhancement experiments to evaluate the performance of our models for various setups and data sets. Before presenting the experiments, we first provide an overview of the experimental setup. This covers the data sets and the steps in constructing the models.

Experiment one compares the performance of the speech enhancement algorithm for all the models (based on diagonal covariance Gaussian PDFs) with SNRs of 10 and 20 dB. Experiment two evaluates all the models on an independent data set for noisy speech with an SNR of 10 dB. Experiment three compares the performance between models based on full covariance Gaussian PDFs and models based on diagonal covariance Gaussian PDFs. Experiment four evaluates the performance of the first order HMM for noisy speech with different SNRs. Experiment five evaluates the performance of the first order HMM over the training iterations. Experiment six determines whether our speech enhancement technique is able to increase the speaker recognition performance of speech corrupted by Gaussian white noise. Finally, the conclusion provides an overview and discussion of these results.

Stellenbosch University http://scholar.sun.ac.za

8.2 Experimental setup and general procedure

The speech models were trained, and the speech enhancement algorithms were evaluated, using the 38 speakers from dialect region one of the TIMIT database. This results in 380 speech files that are approximately three seconds long. An independent test set, used only for testing, were constructed from the first 20 speakers from dialect region two of the same database (200 speech files that are each approximately three seconds long). All the speech were normalised using the robust unity power algorithm that was described in Section 7.4.1.

The noisy signals were generated by adding a random Gaussian noise signal with a fixed variance to each of the speech signals. Data sets were generated with average SNRs of 0, 5, 10, 15 and 20 dB. These data sets were used as input in our speech enhancement system.

Ten different clean speech models were constructed, two versions of each of the five models described in Section 7.4. In one version the Gaussian PDFs have diagonal covariance matrices, while in the other they have full covariance matrices. The diagonal covariance Gaussian PDFs were mostly used during development, because they are faster to train and apply.

Before constructing any of the models we extracted the quantisation levels. This was done by grouping the clean speech data into 256 clusters using VQ. The centroids of the clusters represent the quantisation levels (see Figure 7.1). The feature vectors were constructed next. Each feature vector is a segment of speech consisting of five samples (see Figure 7.2).

The first clean speech model consists of a number of Gaussian PDFs. For each quantisation level eight Gaussian PDFs were constructed. The feature vectors corresponding to each quantisation level were collected. A feature vector corresponds to one of the quantisation levels if the third element of the vector is the closest to that specific quantisation level. The feature vector set of each quantisation level is then segmented into eight clusters using VQ. The result of one clustering is shown in Figure 7.3 in Section 7.4.2. The centroid of each cluster is used to initialise a diagonal (or full) covariance Gaussian PDF. The Gaussian clean speech model consists of 2048 Gaussian PDFs, 8 for each of the 256 quantisation levels. The Gaussian model is trained to update the mean and covariance of each of the PDFs. We call this the Gaussian

Stellenbosch University http://scholar.sun.ac.za

model and uses the shorthand GM to denote it.

The second clean speech model is constructed from the first model by creating 256 GMMs with 8 mixtures each. A Gaussian mixture model is constructed from the group of 8 Gaussian PDFs that correspond to one of the 256 quantisation levels. The Gaussian mixture model is trained to update the mixture weights, the mean and the covariance of each Gaussian mixture model. During training, some of the mixture PDFs are discarded due to a lack of presence in the training data. We call this the Gaussian mixture model and use the shorthand GMM to denote it.

The third clean speech model uses the mixture PDFs of the second model to initialise a 256 state first order ergodic HMM. The PDF of each state is one of the mixture PDFs from the second model. The PDFs of the states are frozen, that means they are not allowed to change during training. This means that only the transition probabilities can be modified. During training any transition that is not observed in the clean speech data is eliminated. In some of the experiments it was found that the speech enhancement algorithm performs better if the remaining transition probabilities leaving a state are all set equal after training. We call this the first order HMM model and uses the shorthand HMM-1 to denote it. This model is used to construct the two other clean speech models.

The first of these, our fourth model, expands the mixtures of the GMMs as described in Section 7.4.5. The model is trained for one iteration, during which some states are eliminated and the transition probabilities are updated. We call this the expanded HMM model and uses the shorthand HMM-eMix to denote it.

The second of these, our fifth model, is simply a second order extension of the first order HMM. The model is trained for one iteration, during which some states are eliminated and the transition probabilities are updated. We call this the second order HMM model and uses the shorthand HMM-2 to denote it.

Stellenbosch University http://scholar.sun.ac.za

8.3 Experiment one: Comparing the clean speech models on the training data

Goal

The aim of this experiment is to evaluate the speech enhancement algorithm's performance for each of the five clean speech models. In this experiment only diagonal covariance Gaussian PDFs were used. We enhanced two sets of noisy speech data, one with an average SNR of 20 dB and another with an average SNR of 10 dB.

Results

The results of this experiment are shown in shown in Table 8.1. For the noisy speech with an average SNR of 20 dB, the Gaussian model and Gaussian mixture model produced similar results. The first order HMM model performed the best, increasing the SNR by almost 4 dB. The expanded HMM performed about the same as the Gaussian and Gaussian mixture models, while the second order HMM model showed the smallest SNR improvement of all the models.

Table 8.1: *Performance (in dB) of the speech enhancement algorithm using models constructed from diagonal covariance Gaussian PDFs.*

Speech SNR	Clean speech Model				
	GM	GMM	HMM-1	HMM-eMix	HMM-2
20 dB	22.33	22.35	23.87	22.66	21.77
10 dB	13.13	14.87	15.25	16.00	15.63

For the 10 dB SNR data the results are quite different. The Gaussian classifier model performed the worst, with an SNR increase of about 3 dB. The Gaussian mixture classifier model showed a considerable improvement compared to the Gaussian model. At this SNR level both the expanded and second order HMM performed better than the first order HMM.

We can use this result to compare our speech enhancement method to other techniques. As a comparison we used the HMM technique described in [28, 29]. However, a direct comparison

Stellenbosch University http://scholar.sun.ac.za

is somewhat difficult since the authors quoted only a minimum and maximum SNR increase. At 10 dB SNR they obtained a performance of between 14.4 and 15.5 dB for their MMSE estimator and between 14.0 and 15.0 dB for their MAP estimator. Our technique, using the first order HMM, performed between 13.0 and 18.0 dB. Using the expanded HMM we noted a performance of between 13.0 and 19.0 dB.

Conclusion

At 20 dB SNR, if would seem that the full power of the more complex models are not utilised. The first order HMM was the best suited model to remove additive noise of this level. At 10 dB SNR the strength of the expanded HMM and second order HMM are clearly seen. Since the expanded HMM performed better than the second order HMM with the mixture PDF states, it would suggest that ability provided by representing each state by only one PDF and eliminating the extra links, is worth more than the ability of the second order HMM to model second order transitional statistics. This indicates that is worth more to increase the number of PDFs to model the speech process than using higher order HMMs.

If we compare our technique to the HMM based speech enhancement technique described in [28, 29], our technique produces better results.

8.4 Experiment two: Comparing the clean speech models on an independent test set

Goal

In this experiment the speech enhancement performance of the models, using diagonal covariance based Gaussian PDFs, are evaluated on an independent test set. This experiment indicates if the clean speech models are general enough to represent the speech process. This experiment was conducted in order to determine how the clean speech models perform on speech that it was not trained on.

Stellenbosch University http://scholar.sun.ac.za

Results

Table 8.2: *Performance (in dB) of the speech enhancement algorithm using the models constructed from diagonal covariance Gaussian PDFs on the test set data at 10 dB SNR.*

GM	GMM	HMM-1	HMM-eMix	HMM-2
13.06	14.57	14.85	15.79	15.17

The results of this experiment are shown in Table 8.2. The performance is very similar to that obtained using the training data (see Table 8.1).

Conclusion

From the results we can conclude that our models are very well suited for the task and represent the speech process reasonably well. If the results dropped dramatically it would have been an indication that the clean speech models were not general enough and that it specialised on the training data. This would indicate that more training data are required for training of the models.

8.5 Experiment three: Comparing full and diagonal covariance Gaussian PDFs

Goal

The aim of this experiment is to make a comparison between models based on full and diagonal covariance Gaussian PDFs.

Results

The results of this experiment are shown in Table 8.3. The first three models performed slightly better when full covariance Gaussian PDFs were used, while the last two models performed slightly better when diagonal covariance Gaussian PDFs were used.

Stellenbosch University http://scholar.sun.ac.za

Table 8.3: *Comparing the performance (in dB) of the speech enhancement algorithm between models constructed using diagonal and full covariance Gaussian PDFs at an SNR of 10 dB.*

Covariance matrix	Clean speech Model				
	GM	GMM	HMM-1	HMM-eMix	HMM-2
Diagonal	13.13	14.87	15.25	16.00	15.63
Full	13.23	15.14	15.51	15.87	15.60

Conclusion

In general, the difference in performance between the two model sets are very small. This suggests that the covariance between the elements of the feature vectors does not contribute a significant amount of information about the system. We would have expected the full covariance models to perform better in all cases. It could be that sufficient data was not available to model the full covariance Gaussian PDFs in the last two models. The slight performance gain that the full covariance models offer do not justify the tremendous increase in processing power to train and apply such models.

8.6 Experiment four: Evaluate the first order HMM for different SNRs

Goal

In this experiment we evaluated the performance of the first order HMM, using diagonal covariance Gaussian PDFs, on various SNRs. This provides an indication of how well the algorithm scale over various SNR levels.

Results

The results of this experiment is shown in Table 8.4.

Stellenbosch University http://scholar.sun.ac.za

Table 8.4: *Performance (in dB) of the first order HMM in different SNRs.*

Model	SNR				
	0 dB	5 dB	10 dB	15 dB	20 dB
HMM-1	7.67	11.51	15.25	18.70	23.87

Conclusion

The lower the SNR, the larger the increase in SNR performance observed. The more noise contained in the speech, the larger the potential for improvement. However, the enhanced speech is less intelligible for the lower SNR speech.

8.7 Experiment five: The role of the number of training iterations

Goal

The aim of this experiment is to demonstrate that only one training iteration is required to update the HMM parameters for optimal SNR increase by the speech enhancement algorithm.

Results

Table 8.5: *Performance (in dB) of the first order HMM over a number of iterations.*

Iteration	1	2	3	4	5	6	7	8	9	10
HMM-1	15.25	15.12	15.09	15.09	15.09	15.08	15.09	15.10	15.10	15.09

The results of this experiment are shown in Table 8.5. We can see that the performance decrease slightly after the first iteration and then stabilises around a constant range.

Stellenbosch University http://scholar.sun.ac.za

Conclusion

During the development we found that only one iteration is required to train the HMMs. This fact saved a lot of time, since training these models are time consuming. For example it means waiting one day, instead of ten (if ten training iterations were used), for an HMM model to train.

8.8 Experiment six: The effect on speaker recognition

Goal

In this experiment we evaluated the performance of a speaker recognition system for clean speech, noisy speech and speech enhanced by our speech enhancement technique.

Setup

The 38 speakers from dialect region one, in the TIMIT speech database, were used in this experiment. Four different speech data sets were evaluated:

- The first set is the original high quality TIMIT speech data. This set is denoted as *HQ*.
- For the second set, the original data are normalised with the robust unity power normaliser and is then filtered with a low pass filter with a cutoff frequency of 3 kHz. This data is used to train the clean speech models used in our speech enhancement technique. This set is denoted by *RUP-LPF*.
- The third set is created by adding noise to the system so that the average SNR of the noisy speech signals are 10 dB. This set is denoted by *RUP-LPF-N10*.
- The last set is created by enhancing the third set using the expanded HMM clean speech model. This set is denoted by *RUP-LPF-N10-SE*.

The speech is pre-emphasised and a Hamming window is applied before a 10'th order LP cepstrum is calculated. The speakers are modelled by Gaussian mixture models with 16 mixtures.

Stellenbosch University http://scholar.sun.ac.za

The models was initialised by VQ codebooks and trained for 20 iterations using the five *SX* files of each speaker.

Finally, the models were tested using the 3 *SI* and 2 *SA* files of each speaker.

Results

Table 8.6: *Speaker recognition performance on clean, noisy and enhanced speech.*

Speech	HQ	RUP-LPF	RUP-LPF-N10	RUP-LPF-N10-SE
Accuracy	97.4%	65.7%	35.8%	41.6%

The results of this experiment are shown in Table 8.6. The performance of the speaker recognition system on the clean high quality TIMIT speech is as good as can be expected.

The performance of the low pass filtered speech dropped considerably. This is due to the fact that the bandwidth of the speech was reduced from 8 kHz to 3 kHz. This also indicates that information that is useful for speaker recognition is present in the frequency components higher than 3 kHz.

The addition of noise reduced the performance of the system even more, while the performance of the enhanced speech is somewhat better.

To determine if the outcome is statistically significant we conducted a one-sided McNemar test [95] on the results obtained from *RUP-LPF-N10* and *RUP-LPF-N10-SE*. The error distribution

Table 8.7: *The error distribution between* RUP-LPF-N10 *and* RUP-LPF-N10-SE.

		RUP-LPF-N10-SE	
		Correct	Incorrect
RUP-LPF-N10	Correct	55	13
	Incorrect	24	98

between *RUP-LPF-N10* and *RUP-LPF-N10-SE* is shown in Table 8.7. By substituting these values into the McNemar equations, we found that the observed differences in the results has a probability of only 4.94% of happening by chance. We can therefore conclude that there is a

Stellenbosch University http://scholar.sun.ac.za

genuine difference in the results of *RUP-LPF-N10* and *RUP-LPF-N10-SE* and that the resulting outcome is statistically significant.

Conclusion

This experiment demonstrates that our speech enhancement technique, using the expanded HMM clean speech model, can increase the performance of a speaker recognition system.

8.9 Conclusion

In this chapter we evaluated our speech enhancement technique using five different clean speech models. Of these models, depending on the SNR level, the first order HMM or the expanded HMM performed the best. For lower SNR levels the second order HMM performs better than the first order HMM, but not better than the expanded HMM.

The speech enhancement technique demonstrates similar performance on both the training and the test set. This suggests that our clean speech models are sufficiently generalised for the speech process.

We also compared the performance of the models using diagonal and full Gaussian covariance models. The results are very similar but in most cases the full covariance model performs slightly better. However, the full covariance models takes considerably longer to train and apply. The slight performance increase that they offer is outweighted by their computational cost.

We also observed that only one training iteration is required to train the HMMs. After each consecutive iteration the performance drops slightly and eventually stabilises around some constant value. This fact saved a considerable amount of development time.

Also demonstrated was the fact that our speech enhancement technique, using the expanded HMM clean speech model, increased the speaker recognition performance of speech corrupted by Gaussian white noise.

Stellenbosch University http://scholar.sun.ac.za

Since the speech enhancement method we introduced was different from all the other approaches, there is no proper benchmark against which the result can be compared. A comparison between our method and the HMM-based statistical speech enhancement methods discussed in section 1.3.1.3 and Section 7.3 is also difficult since the test data is not the same. If we do however draw a comparison based purely on the SNR improvements obtained, then our speech enhancement method generally performs better than those methods.

Stellenbosch University http://scholar.sun.ac.za

Chapter 9

Conclusion

9.1 Summary

In this project we studied a wide range of subjects including speech analysis, robust speech analysis, speaker recognition, speech enhancement and high frequency transmission channels. A short summary follows:

- The basics of speech analysis were investigated, with LP cepstral features being emphasised. This laid the foundation for speaker recognition and the investigation of the HF channel.
- Various speaker recognition systems were investigated and we described the current industry standard which employs Gaussian mixture models to model the speakers.
- Several robust feature analysis techniques were investigated. Most of them are based on inter- and intra-frame processing of the LP or mel-scale cepstral features. Two of these methods examined new cepstral features. The first was based on human perception and the second was based on the modification of autocorrelation sequences.
- A performance baseline was established for speaker recognition on speech transmitted over an HF channel. With the aid of two robust techniques, discarding the zero'th order

 Stellenbosch University http://scholar.sun.ac.za

cepstral coefficient and cepstral post filtering, the performance was slightly increased. Another robust technique, cepstral mean subtraction, actually decreased the performance.

- The unsatisfactory results obtained from the speaker recognition experiments lead to the investigation of how the HF channel and SSB modulation deform the LP coefficients and the LP cepstral features. The results are summarised below:
  - A Phase error in SSB modulation and demodulation only modify the zero'th order cepstral coefficient.
  - A frequency error in SSB modulation or demodulation causes the sidebands to shift towards or away from each other, causing aliasing and a dead band in the spectrum. If these shifts are small, as expected from a high quality modulation system, the LP cepstrum will only experience minor deformation.
  - Ionospheric absorption only modify the zero'th order cepstral coefficient.
  - Multi-path signals can seriously deform the LP cepstrum if the interfering signal is of the same order of amplitude as the expected signal.
  - Doppler shifts have the same effect as a frequency error in the modulation system. For small Doppler shifts the deformation might be small, but larger shifts can result in large deformation of the LP cepstrum.
  - Large noise components can seriously deform the LP cepstrum.

  Of these effects multi-path signals, large Doppler shifts and noise has the largest potential to deform the LP cepstrum.

- A new statistically based speech enhancement technique was introduced. A hidden Markov model is generated by incremental construction and training from simpler models such as Gaussian and Gaussian mixture models. The speech enhancement algorithm can operate on both PDF classifier and HMM clean speech models. The technique performs better than other HMM based methods found in the literature. It was also demonstrated that speech enhanced with the new technique increased the speaker recognition performance of speech corrupted by Gaussian white noise.

 Stellenbosch University http://scholar.sun.ac.za

9.2 Future work

A study such as this is never complete and there are a number of ideas that we would have liked to investigate were it not for time constraints:

- In Section 4.2 and Section 4.3 we introduced robust feature analysis techniques that modify the cepstral features. Only three of these robust techniques (discarding the zero'th order cepstral coefficient, cepstral post filtering and cepstral mean subtraction) were incorporated in our speaker recognition experiments. Implementation of all the robust techniques will take some time, but if they are available the HF speaker recognition experiments can be repeated to evaluate all the techniques.

- Only the LP cepstrum is used as a feature in all the experiments. An investigation that also include the Mel-warped cepstrum, PLP cepstrum and RAS mel-scale cepstrum will be useful. RAS-MFCC feature might prove to be very effective, due to the fact that it is designed to be more resistant against additive noise.

- This project did not conduct a study to determine the relative extent to which the various effects introduced by the HF channel and the SSB modulation system are present in typical HF transmitted speech. Such a study would complement the work presented in Chapter 6. With the investigation of HF specific solutions, such a project will be a large undertaking.

- The hidden Markov models used in the speech enhancement experiments were constructed from a fixed parameter set. This is largely due to the fact that the construction of these models take a considerable amount of time. Here is a list of parameter variations that we would like to investigate:

  - The initial number of quantisation levels was chosen to be 256. The effect of more or fewer levels can be investigated. Working with more quantisation levels will decrease the quantisation noise, but since the number states of the HMMs are determined by the number of quantisation levels, this can lead to even larger HMMs than was used in this study. By choosing fewer quantisation levels it could be possible to create higher order HMMs and study their performance.

  - The number of speech samples in the feature vector were set to 5. The effect of a feature vector using 3, 7, etc. samples can be investigated. The fewer samples in the vector, the fewer training data will be required to train the models, since the possible patterns will be less. However, for more samples more complex patterns can be recognised, thus providing the capability to learn even more context sensitive constraints.
  - The feature vector data associated with each quantisation level was clustered into eight subsections. This figure can be modified dynamically by increasing the number of clusters if enough data are available, and decreasing the number of clusters when the data are sparse.
  - To obtain a smoother signal we can rather over sample the speech data instead of low pass filtering it. This way no information is discarded.

- The training of the speech models and application of the speech enhancement algorithms require a significant amount of memory and computational power to execute. Optimisation of the code can help to increase development time. Optimisation can facilitate a faster development cycle and allow for increase in model complexity. We have already incorporated a number of optimisation techniques of which the Viterbi beam algorithm offered the most advantages.

Stellenbosch University http://scholar.sun.ac.za

Bibliography

[1] R. Cardin, T. Normandin and E. Millie, "Inter-word coarticulation modeling and MMIE training for improved connected digit recognition," in *IEEE ICASSP*, vol. 2, pp. 243–246, 1993.

[2] X. Aubert, R. Haeb-Umbach and H. Ney, "Continuous mixture densities and linear discriminant analysis for improved context-dependent acoustic models," in *IEEE ICASSP*, vol. 2, pp. 648–651, 1993.

[3] D. Reynolds, "Large population speaker identification using clean and telephone speech," *IEEE Signal Processing Letters*, vol. 2, pp. 46–48, Mar. 1995.

[4] B.A. Dautrich, L.R. Rabiner and T.B. Martin, "On the effects of varying filterbank parameters on isolated word recognition," *IEEE Transactions on Signal Processing*, vol. 31, pp. 793–806, Aug. 1983.

[5] B. Juang, "Speech recognition in adverse environments," *Computer Speech and Language*, pp. 275–294, 1991.

[6] H. Gish and M. Schmidt, "Text-independent speaker identification," *IEEE Signal Processing Magazine*, vol. 11, pp. 18–31, Oct. 1994.

[7] J.R. Deller, J.G. Proakis and J.H.L. Hansen, *Discrete-Time Processing of Speech Signals*. Prentice Hall International, Third ed., 1996.

[8] A. Higgins, L. Bahler, J. Porter, "Voice identification using nearest neighbor distance measure," in *IEEE ICASSP*, vol. 2, pp. 375–378, 1993.

 Stellenbosch University http://scholar.sun.ac.za

[9] F. Soong, A. Rosenberg, L. Rabiner and B. Juang, "A vector quantization approach to speaker recognition," in *IEEE ICASSP*, vol. 2, pp. 387–390, 1985.

[10] D. O'Shaughnessy, "Speaker recognition," *IEEE Acoustics, Speech & Signal Processing Magazine*, vol. 3, pp. 4–17, Oct. 1986.

[11] D. Reynolds and R. Rose, "Robust text-independent speaker identification using Gaussian mixture speaker models," *IEEE Transactions on Speech & Audio Processing*, vol. 3, pp. 72–83, Jan. 1995.

[12] D. Reynolds, "Speaker identification and verification using Gaussian mixture speaker models," *Speech Communication*, vol. 17, pp. 91–108, Mar. 1995.

[13] S. Burley and M. Darnell, "Robust impulse noise suppression using adaptive wavelet denoising," in *IEEE ICASSP*, vol. 5, pp. 3417–3420, 1997.

[14] I. Schick and H. Krim, "Robust wavelet thresholding for noise suppression," in *IEEE ICASSP*, vol. 5, pp. 3421–3424, 1997.

[15] J. Seok and K. Bae, "Speech enhancement with reduction of noise components in the wavelet domain," in *IEEE ICASSP*, vol. 2, pp. 1323–1324, 1997.

[16] S. F. Boll, "Suppression of acoustic noise in speech using spectral subtraction," *IEEE Transactions on Acoustics, Speech & Signal Processing*, vol. 27, pp. 113–120, Apr. 1979.

[17] D. Klatt, "A digital filter bank for spectral matching," in *IEEE ICASSP*, pp. 573–576, 1976.

[18] J. Holmes and N. Sedgwick, "Noise compensation for speech recognition using probabilistic models," in *IEEE ICASSP*, pp. 741–744, 1986.

[19] A. Varga, R. Moore, J. Bridle, K. Ponting and M. Russell, "Noise compensation algorithms for use with hidden Markov model based speech recognition," in *IEEE ICASSP*, pp. 481–484, 1988.

[20] N. Virag, "Single channel speech enhancement based on masking properties of the human auditory system," *IEEE Transactions on Speech & Audio Processing*, vol. 7, pp. 126–137, Mar. 1999.

 Stellenbosch University http://scholar.sun.ac.za

[21] Y. Gong, "Speech recognition in noisy environments: A survey," *Speech Communication*, vol. 16, pp. 261–291, 1995.

[22] J. Lim and A. Oppenheim, "All-pole modelling of degraded speech," *IEEE Transactions on Acoustics, Speech & Signal Processing*, vol. 26, pp. 197–210, 1978.

[23] A.P. Dempster, N.M. Laird and D.B Rubin, "Maximum likelihood from incomplete data via the EM algorithm," *Journal of the Royal Statistical Society*, vol. 39, pp. 1–88, 1977.

[24] J. Hansen and M. Clements, "Constrained iterative speech enhancement with application to speech recognition," *IEEE Transactions on Signal Processing*, vol. 39, pp. 795–805, Apr. 1991.

[25] T. Sreenivas and P. Kirnapure, "Codebook constrained Wiener filtering for speech enhancement," *IEEE Transactions on Speech & Audio Processing*, vol. 4, pp. 383–389, May 1996.

[26] K. Paliwal and A. Basu, "A speech enhancement method based on Kalman filtering," in *IEEE ICASSP*, pp. 177–180, 1987.

[27] F. Caruthers, "Prolog to statistical-model-based speech enhancement systems," *Proceedings IEEE*, vol. 80, pp. 1524–1525, Oct. 1992.

[28] Y. Ephraim, "Statistical-model-based speech enhancement systems," *Proceedings IEEE*, vol. 80, pp. 1526–1555, Oct. 1992.

[29] Y. Ephraim, D. Malah and H.H. Juang, "On the application of hidden Markov models for enhancing noisy speech," *IEEE Transactions on Acoustics, Speech & Signal Processing*, vol. 37, pp. 1846–1856, Dec. 1989.

[30] B. Logan and A. Robinson, "Enhancement and recognition of noisy speech within an autoregressive hidden Markov model framework using noise estimates from the noisy signal," in *IEEE ICASSP*, vol. 2, pp. 843–846, 1997.

[31] Y. Ephraim, "A Bayesian estimation approach for speech enhancement using hidden Markov models," *IEEE Transactions on Acoustics, Speech & Signal Processing*, vol. 40, pp. 725–735, Apr. 1992.

 Stellenbosch University http://scholar.sun.ac.za

[32] Y. Ephraim, "Speech enhancement using state dependent dynamical system model," in *IEEE ICASSP*, vol. 1, pp. 289–292, 1992.

[33] R. Lummis, "Speaker verification by computer using speech intensity for temporal registration," *IEEE Transactions on Audio Electro-accoust.*, vol. 21, pp. 80–89, 1973.

[34] B. Atal, "Automatic speaker recognition based on pitch contours," *Journal of the Acoustics Society of America*, vol. 52, pp. 1678–1697, 1972.

[35] S. Das and W. Mohn, "A scheme for speech processing in automatic speaker verification," *IEEE Transactions on Audio Electro-accoust.*, vol. 19, pp. 32–43, 1971.

[36] J. Markel and A. Gray, *Linear Prediction of Speech.* Springer-Verlag, 1976.

[37] A. Rosenberg and M. Sambur, "New techniques for automatic speaker verification," *IEEE Transactions on Acoustics, Speech & Signal Processing*, vol. 23, pp. 169–176, 1975.

[38] K.P. Li, G.W. Huges and A.S. House, "Correlation characteristics and dimensionality of speech spectra," *Journal of the Acoustics Society of America*, vol. 46, pp. 1019–1025, 1969.

[39] B. Imperl, Z. Kacic and B. Horvat, "A study of harmonic features for the speaker recognition," *Speech Communication*, vol. 22, pp. 385–402, Dec. 1997.

[40] B. Atal, "Effectiveness of linear prediction characteristics of the speech wave for automatic speaker identification and verification," *Journal of the Acoustics Society of America*, vol. 55, pp. 1304–1312, 1974.

[41] H. Hermansky, "Perceptual linear predictive (PLP) analysis of speech," *Journal of the Acoustics Society of America*, pp. 1738–1752, 1990.

[42] K. Yuo and H. Wang, "Robust features for noisy speech recognition based on temporal trajectory filtering of short-time autocorrelation sequences," *Speech Communication*, vol. 28, pp. 13–24, Jan. 1999.

[43] R. Mammone, X. Zhang, and R. Ramachandran, "Robust speaker recognition," *IEEE Signal Processing Magazine*, vol. 13, pp. 58–71, Sept. 1996.

 Stellenbosch University http://scholar.sun.ac.za

[44] K. Assaleh and R. Mammone, "New LP-derived features for speaker identification," *IEEE Transactions on Speech & Audio Processing*, vol. 2, pp. 630–638, Oct. 1994.

[45] L. Rabiner and B. Juang, *Fundamentals of Speech Recognition*. Prentice Hall, 1993.

[46] B.H. Juang, l.R. Rabiner and J.G. Wilpon, "On the use of bandpass liftering in speech recognition," *IEEE Transactions on Acoustics, Speech & Signal Processing*, vol. 35, pp. 948–947, July 1987.

[47] M. S. Zilovic, R. P. Ramachandran, and R. J. Mammone, "A fast algorithm for finding the adaptive component weighting cepstrum for speaker recognition," *IEEE Transactions on Speech & Audio Processing*, vol. 5, pp. 84–86, Jan. 1997.

[48] M.S. Zilovic, R.P. Ramachandran and R.J. Mammone, "Speaker identification based on the use of robust cepstral features obtained from pole-zero transfer functions," *IEEE Transactions on Speech & Audio Processing*, vol. 6, pp. 260–267, May 1998.

[49] A. Oppenheim and R. Schafer, *Discrete-Time Signal Processing*. Prentice Hall, 1989.

[50] V. Ramamoorthy, N.S. Jayant, R.V. Cox, M.M. Sondhi, "Enhancement of ADPCM speech coding with backward adaptive algorithms for postfiltering and noise feedback," *IEEE Journal on Selected Areas in Communication*, vol. 6, pp. 364–382, Feb. 1988.

[51] F. Soong and A. Rosenberg, "On the use of instantaneous and transitional spectral information in speaker recognition," *IEEE Transactions on Acoustics, Speech & Signal Processing*, vol. 36, pp. 871–879, June 1988.

[52] B. Atal, "Automatic recognition of speakers by their voices," *Proceedings IEEE*, vol. 64, pp. 460–475, Apr. 1976.

[53] D. Reynolds, "Experimental evaluation of features for robust speaker identification," *IEEE Transactions on Speech & Audio Processing*, vol. 2, pp. 639–643, Oct. 1994.

[54] D. Naik, "Pole-filtered cepstral mean subtraction," in *IEEE ICASSP*, vol. 1, pp. 157–160, 1995.

[55] H. Hermansky and N. Morgan, "RASTA processing of speech," *IEEE Transactions on Speech & Audio Processing*, vol. 2, pp. 587–589, Oct. 1994.

 Stellenbosch University http://scholar.sun.ac.za

[56] H. Hermansky, N. Morgan, H.G. Hirsch, "Recognition of speech in additive and convolutional noise based on RASTA spectral processing," in *IEEE ICASSP*, vol. 2, pp. 83–86, 1993.

[57] A. Higgins and L. Bahlar, "Text-independent speaker verification by discriminator counting," in *IEEE ICASSP*, pp. 405–408, 1991.

[58] K. Li and E. Wrench, "An approach to text-independent speaker recognition with short utterences," in *IEEE ICASSP*, pp. 555–558, 1983.

[59] A. Rosenberg and F. Soong, "Evaluation of a vector quantization talker recognition system in text independent and text dependent modes," *Computer Speech and Language*, pp. 143–157, 1987.

[60] F. Bimbot, I. Magrin-Chagnolleau and L. Mathan, "Second-order statistical measures for text-independent speaker identification," *Speech Communication*, vol. 17, pp. 177–192, Mar. 1995.

[61] F. Bimbot, L. Mathan, A. de Lima and G. Chollet, "Standard and target driven AR-vector models for speech analysis and speaker recognition," in *IEEE ICASSP*, vol. 2, pp. 5–8, 1992.

[62] C. Montacié, P. Deléglise, F. Bimbot and M. Caraty, "Cinematic techniques for speech processing: Temporal decomposition and multivariate linear prediction," in *IEEE ICASSP*, vol. 1, pp. 153–156, 1992.

[63] Y. Bennani and P. Gallnari, "Neural networks for discrimination and modelization of speakers," *Speech Communication*, vol. 17, pp. 159–175, Mar. 1995.

[64] K.R. Farrell, R.J. Mammone and K.T. Assaleh, "Speaker recognition using neural networks and conventional classifiers," *IEEE Transactions on Speech & Audio Processing*, vol. 2, pp. 194–205, Jan. 1994.

[65] J. Oglesby and J. Mason, "Optimization of neural models for speaker identification," in *IEEE ICASSP*, pp. 261–264, 1990.

[66] J. Oglesby and J. Mason, "Radial basis function networks for speaker recognition," in *IEEE ICASSP*, pp. 393–396, 1991.

 Stellenbosch University http://scholar.sun.ac.za

[67] Y. Bennani, F. Fogelman and P. Gallinari, "A connectionist approach for speaker identification," in *IEEE ICASSP*, pp. 265–268, 1990.

[68] Y. Bennani and P. Gallinari, "On the use of TDNN-extracted feature information in a talker identification," in *IEEE ICASSP*, pp. 385–388, 1991.

[69] M. Savic and S. K. Gupta, "Variable parameter speaker verification system based on hidden Markov modeling," in *IEEE ICASSP*, pp. 281–284, 1990.

[70] N. Tishby, "On the application of mixture AR hidden Markov models on text-independent speaker recognition," *IEEE Transactions on Signal Processing*, vol. 39, pp. 563–570, Mar. 1991.

[71] M.E. Forsyth, A.M. Sutherland, J.A. Elliott and M.A. Jack, "HMM speaker verification with sparse training data on telephone quality speech," *Speech Communication*, vol. 13, pp. 411–416, Dec. 1993.

[72] M. Forsyth, "Discriminating observation probability (DOP) HMM for speaker verification," *Speech Communication*, vol. 17, pp. 117–129, Mar. 1995.

[73] J. de Veth and H. Bourlard, "Comparison of hidden Markov model techniques for automatic speaker verification in real-world conditions," *Speech Communication*, vol. 17, pp. 81–90, Mar. 1995.

[74] P. Horowitz and W. Hill, *The Art of Electronics*. Cambridge University Press, Second, low price ed., 1996.

[75] J. Proakis and M. Manolakis, *Digital Signal Processing: Principles, Algorithms and Applications*. Prentice Hall International, Third International ed., 1996.

[76] S. Haykin, *Adaptive Filter Theory*. Prentice Hall International, third ed., 1996.

[77] J.A. du Preez, "Spraakonafhanklike sprekerherkenning met behulp van trosvorming," Master's thesis, University of Stellenbosch, Nov. 1985.

[78] L. Marple, "A new autoregressive spectrum analysis algorithm," *IEEE Transactions on ASSP*, vol. 28, pp. 441–454, Aug. 1980.

 Stellenbosch University http://scholar.sun.ac.za

[79] S. Furui, "Cepstral analysis technique for automatic speaker verification," *IEEE Transactions on Acoustics, Speech & Signal Processing*, vol. 29, pp. 254–272, Apr. 1981.

[80] J. Koehler, H. Hermansky, N. Morgan, H.G. Hirsch, G, Tome, "Integrating RASTA-PLP into speech recognition," in *IEEE ICASSP*, vol. 1, pp. 421–424, 1994.

[81] J. Pool and J. du Preez, "HF speaker recognition," tech. rep., University of Stellenbosch, Mar. 1999.

[82] J. Lourens, "HF propagation radio channel," tech. rep., University of Stellenbosch, Mar. 1999.

[83] J. Pool, J.A. du Preez and J.G. Lourens, "The effects of HF channels on LPC cepstral parameters," in *Pattern Recognition Association of South Africa*, Nov. 1999.

[84] *ITT Reference Data for Radio Engineers*. Howard W. Sams & Co. Inc., Fifth ed., 1970.

[85] Edited by M.P.M Hall, L.W. Barclay and M.T. Hewitt, *Propagation of Radiowaves*. The institution of Electrical Engineers, 1996.

[86] L. Rabiner, "A tutorial on hidden Markov models and selected applications in speech recognition," *Proceedings IEEE*, vol. 77, pp. 257–285, Feb. 1989.

[87] L. Rabiner and B. Juang, "An introduction to hidden Markov models," *IEEE Acoustics, Speech & Signal Processing Magazine*, vol. 3, pp. 4–16, Jan. 1986.

[88] Y. He, "Extended Viterbi algorithm for second-order hidden Markov process," in *Proceedings of the IEEE 9'th International Conference on Pattern Recognition*, 1988.

[89] A. Kriouile, J.F. Mari and J.P. Haton, "Some improvements in speech recognition based on HMM," in *IEEE ICASSP*, pp. 545–548, 1990.

[90] J. du Preez, *Efficient High-Order Hidden Markov Modelling*. PhD thesis, University of Stellenbosch, Mar. 1998.

[91] J. du Preez, "Efficient training of high-order hidden Markov models using first-order representation," *Computer Speech and Language*, pp. 23–39, 1998.

 Stellenbosch University http://scholar.sun.ac.za

[92] L. E. Baum, T. Petrie, G. Soules, and N. Weiss, "A maximization technique occurring in statistical analysis of probabilistic functions in Markov chains," *The Annals of Mathematical Statistics*, vol. 41, no. 1, pp. 164–171, 1970.

[93] L. Rabiner, J. Wilpon, and B. Juang, "A segmental K-means training procedure for connected word recognition," 1986.

[94] R. Gray, "Vector quantization," *IEEE Acoustics, Speech & Signal Processing Magazine*, vol. 1, pp. 2–29, Apr. 1984.

[95] L. Gillick and S. Cox, "Some statistical issues in the comparison of speech recognition algorithms," 1989.

[96] R.E. Ziemer and W.H. Tranter, *Principles of Communications: Systems, Modulation and Noise*. John Wiley & Sons, Inc, Fourth ed., 1995.

[97] K. Fukunaga, *Introduction To Statistical Pattern Recognition*. Morgan Kaufmannl, Second ed., 1990.

[98] T. Moon, "The expectation-maximization algorithm," *IEEE Signal Processing Magazine*, vol. 13, pp. 47–60, Nov. 1996.

[99] F. Jelinek, *Statistical Methods For Speech Recognition*. MIT Press, First ed., 1997.

Stellenbosch University http://scholar.sun.ac.za

Appendix A

The Hilbert transform

The Hilbert transform (HT) is an all-pass filter that shifts the phase of all positive frequency components of its input by $-\frac{1}{2}\pi$ radians (90°). The transfer function of the ideal HT is specified by [96, p.94][75, p.657]

$$H(\omega) = -j\ sgn(\omega), \tag{A.1}$$

where sgn is the signum function

$$sgn(\omega) = \begin{cases} 1, & \omega > 0 \\ 0, & \omega = 0 \\ -1, & \omega < 0 \end{cases}. \tag{A.2}$$

The unit impulse response of the HT is given by

$$h(n) = \begin{cases} \frac{2}{\pi}\frac{\sin^2(0.5\pi n)}{n}, & n \neq 0 \\ 0, & n = 0 \end{cases}. \tag{A.3}$$

The ideal impulse response is noncausal and infinite in duration. We note that the magnitude of the HT is

$$|H(\omega)| = 1 \tag{A.4}$$

Stellenbosch University http://scholar.sun.ac.za

and that the autocorrelation of the HT is

$$\begin{aligned} r_{hh}(n) &= h(n) * h(-n) \\ S_{hh}(\omega) &= H(\omega) \cdot H(-\omega) \\ &= H(\omega) \cdot H^*(\omega) \\ &= |H(\omega)|^2 \\ &= 1^2 = 1 \\ r_{hh}(n) &= \delta(n), \end{aligned} \tag{A.5}$$

where $S_{hh}(\omega)$ is the power spectrum of the HT.

Stellenbosch University http://scholar.sun.ac.za

Appendix B

Vector Quantisation

Vector quantisation is part of a larger family of clustering algorithms. In [97, p.508-563] a whole chapter is devoted to clustering algorithms. Vector quantisation is often used in data compression and coding applications [94]. The two algorithms used in our system are the K-means [7, p.71] and the non-uniform binary split [7, p.432].

K-means VQ (KMVQ)

For a K vector codebook we start by choosing K arbitrary initial codebook vectors, cb_k for $k = 1, 2, \ldots, K$. We then follow the next three steps:

- For each feature vector x we calculate the distance d to all the codebook vectors. The feature vector is then associated (quantised) with the nearest codebook vector. This process creates K clusters.
- For each cluster the centroids are computed. These centroids are now the new codebook vectors, cb_k.
- These steps are repeated for a predefined number of times or until a minimum distortion for the feature vectors are reached.

Stellenbosch University http://scholar.sun.ac.za

Non-uniform binary split VQ (NUBSVQ)

Non-uniform binary split VQ starts with one cluster (all the data) which is split into two new clusters using the K-means algorithm as described. The cluster with the largest total distortion, where distortion is the sum of the distances of all the vectors in the clusters from the cluster centroid, is then split into two new clusters. This process is repeated i times to produce 2^i codebooks. The process is described as follows:

- Select the cluster with the largest total distortion.
- Split this cluster into two new clusters using the K-means algorithm.
- Repeat this process i times.

Non-uniform binary split VQ repeatedly splits the cluster with the largest total distortion. This process is faster than the full K-means algorithm and provides better clustering of the data.

Stellenbosch University http://scholar.sun.ac.za

Appendix C

Expectation Maximisation (EM) algorithm

The EM algorithm is frequently used to estimate parameters in situations where the data necessary for estimation are not directly accessible or where some of the data are missing. The EM algorithm provides a maximum likelihood (ML) estimate of parameters where there is a many-to-one mapping between the underlying distribution and the distribution governing the observation data.

The EM algorithm consists of two steps, an expectation step and a maximisation step. The expectation step uses the current parameter set and observations to estimate the unobserved data. The maximisation step estimates a new parameter set from the estimated unobserved data. These two steps are iterated until some convergence criterium is satisfied.

The underlying principle of the EM algorithm has been developed by a number of independent researchers, but Dempster [23] brought all the ideas together and proved convergence of the algorithm. The EM algorithm has a wide range of applications in the signal processing as well as numerous other fields [98]. Our interest in the EM algorithm is in its application in the training of Gaussian mixture models [11] and hidden Markov models [99, p.148-164][7, p.677-744].

Stellenbosch University http://scholar.sun.ac.za

General statement of the EM algorithm

The general statement of the EM algorithm, as found in [98], will now be described.

Lets denote the observation sample space by Y, and let $\mathbf{y} \in \Re^m$ be an observation of Y. Let $\mathbf{x} \in \Re^n$, for $m < n$, be an outcome in the underlying space χ. We refer to $\mathbf{x}$ as the complete data. However, the complete data are not observed directly, only indirectly via $\mathbf{y}$. There is a many-to-one mapping between $\mathbf{x}$ and $\mathbf{y}$, given by

$$\mathbf{y} = g(\mathbf{x}). \tag{C.1}$$

This means that a subset of χ can be mapped to the observation $\mathbf{y}$. This subset is denoted by $\chi(\mathbf{y})$.

The PDF of the complete data is $f(\mathbf{x}|\theta)$, where $\theta \in \Theta \subset \Re^r$ is the set of parameters of the density. The PDF is assumed to be a continuous function of θ and is approximately differential. Furthermore, the ML estimate of θ is assumed to lie within the region Θ. The PDF of the incomplete data is

$$f(\mathbf{y}|\theta) = \int_{\chi(y)} f(\mathbf{x}|\theta) d\mathbf{x}. \tag{C.2}$$

The likelihood function is denoted by

$$l_{\mathbf{y}}(\theta) = f(\mathbf{y}|\theta), \tag{C.3}$$

and the log-likelihood function as

$$L_{\mathbf{y}}(\theta) = \log f(\mathbf{y}|\theta). \tag{C.4}$$

We want to find the θ that maximises $\log f(\mathbf{x}|\theta)$, but we do not have direct access to $\mathbf{x}$. Instead we maximise the expectation of $\log f(\mathbf{x}|\theta)$, given the current estimate of θ and the observed data $\mathbf{y}$. This process is expressed in two steps. The first step is the expectation or E-step, while the second is the maximisation or M-step. These steps are iterated until some convergence criteria is satisfied. The estimate of the parameters at the k'th iteration is denoted by $\theta^{[k]}$.

During the E-step we compute a Q function, given by:

$$Q(\theta|\theta^{[k]}) = E[\log f(\mathbf{x}|\theta)|\mathbf{y}, \theta^{[k]}]. \tag{C.5}$$

Stellenbosch University http://scholar.sun.ac.za

During the M-step the next estimate of the parameters is obtained by finding the θ that maximises the Q function. This is written as:

$$\theta^{[k+1]} = \arg\max_{\theta} Q(\theta|\theta^{[k]}). \tag{C.6}$$

These two steps are iterated until $||\theta^{[k]} - \theta^{[k-1]}|| < \epsilon$, where $||\cdot||$ is the distance measure and ϵ is a small constant value. The algorithm is repeated until the models are closer than a predefined distance from each other. In other words, the new parameters only differ slightly from each other. The EM algorithm is guaranteed to converge.

The EM algorithm is conceptually very simple, but computing the expectation and performing the maximisation can be quite complex and time consuming.

Stellenbosch University http://scholar.sun.ac.za

Appendix D

HMM mixture PDF expansion algorithm

```
void hmm_mixture_expand(const HMM& oldMod, HMM& newMod) const
{
  cout << "Enter the largest number of PDFs allowed per mixture PDF"
       << " after expansion: ";
  int largestMixSize;
  cin >> largestMixSize;

  if (largestMixSize < 1)
  {
    cerr << "ERROR: largest size the new mixture must be one or "
         << "larger" << endl;
    return true;
  } // largestMixSize

  // Obtain the PDF classifier from the old model.
  pdfClassifier* pdfClsPtr = oldMod.copyClassifier();

  // Used to store the number of old and new states.
  int numberOfOldStates = oldMod.states();
```

Stellenbosch University http://scholar.sun.ac.za

```
int numberOfNewStates = 0;

// Store old class indexes (Mapping between states and PDFs).
Vector<int> oldClassIndexes = oldMod.classIndexes();

// Create vectors to store the old links and their weights
Vector<Vector<int> > oldDest(numberOfOldStates);
Vector<Vector<REAL> > oldDestWeight(numberOfOldStates);


for (int state = 0; state < numberOfOldStates; state++)
{
  // Get the pdf id that the old state points to
  int oldPdfId = oldClassIndexes[state];

  // Determine old destination and weights
  oldDest[state].resize(oldMod.state(state).noOfLinks() );
  oldDestWeight[state].resize(oldMod.state(state).noOfLinks() );
  for (int j = 0; j < oldDest[state].size(); j++)
  {
    oldDest[state][j] = oldMod.state(state).linkDest(j);
    oldDestWeight[state][j] =
      EXP_B(oldMod.state(state).linkStrength(j) );
  } //for j

  // Count how many new states we will need
  if (oldPdfId == -9999) // Null state
  {
    numberOfNewStates++;
  }// if oldClassIndexes
  else
```

Stellenbosch University http://scholar.sun.ac.za

```
    {
      // Extract underlying pdf
      Pdf_N* pdfPtr;
      pdfPtr = pdfClsPtr->pdf(oldPdfId);

      const MixturePDF_N* mixPdfPtr;
      mixPdfPtr = dynamic_cast<const MixturePDF_N*> (pdfPtr);
      if (mixPdfPtr)
      {
        numberOfNewStates +=
          ( (mixPdfPtr->noOfPdfs() - 1) / largestMixSize) + 1;
      } // if mixPdfPtr
      else
      {
        numberOfNewStates++;
      }// else mixPdfPtr
    }// else oldOdfId
  }// state

  cout << "The old model had " << oldMod.states()
       << " states, the new model will have " << numberOfNewStates
       << " states." << endl;

  // Determine how many Pdf_Ns we will need
  int numberOfNewPdfs = 0;
  for (int pdf = 0; pdf < pdfClsPtr->noOfClasses(); pdf++)
  {
    const MixturePDF_N* mixPdfPtr;
    mixPdfPtr = dynamic_cast<const MixturePDF_N*>
         (pdfClsPtr->pdf(pdf) );
    if (mixPdfPtr)
```

Stellenbosch University http://scholar.sun.ac.za

```
    {
      numberOfNewPdfs +=
        ( (mixPdfPtr->noOfPdfs() - 1) / largestMixSize) + 1;
    } // if mixPdfPtr
    else
    {
      numberOfNewPdfs++;
    }// else mixPdfPtr
  }// pdf

  cout << "The old pdf classifier had " << pdfClsPtr->noOfClasses()
       << " pdfs, the new pdf classifier will have "
       << numberOfNewPdfs
       << " pdfs." << endl;

  // Variables needed for the new model
  Vector<int> newClassIndexes(numberOfNewStates);
  Vector<RCPtr<Pdf_N> > newPdfs(numberOfNewPdfs);
  string modlId = oldMod.modelId();
  modlId += "_ExpandeMixture";

  cout << "Expansion will now start." << endl;

  int newStateIdx = 0;
  int newPdfIdx = 0;
  Vector<Vector<int> > old2new(numberOfOldStates);
  Vector<int> new2old(numberOfNewStates);
  Vector<float> tmpWeights(numberOfNewStates);
  /* The new2oldPdfId associative map is used to keep track of the
   * relationship between the old pdf indexes and the new ones
   */
```

Stellenbosch University http://scholar.sun.ac.za

```
map<int,int> new2oldPdfId;
for (int state = 0; state < numberOfOldStates; state++)
{
  // Get the pdf id that the old state points to
  int oldPdfId = oldClassIndexes[state];
  if (oldPdfId == -9999) // Null state
  {
    // Not assosiated with a pdf
    newClassIndexes[newStateIdx] = -9999;
    tmpWeights[newStateIdx] = 1.0;
    old2new[state].resize(1);
    old2new[state][0] = newStateIdx;
    new2old[newStateIdx++] = state;
  }// if oldClassIndexes
  else
  {
    // See if the pdf have occured before
    bool repeatPdf = false;
    map<int,int>::iterator p = new2oldPdfId.find(oldPdfId);
    if (p == new2oldPdfId.end() ) // new pdf id
    {
      // Save the first new id in the map
      new2oldPdfId[oldPdfId] = newPdfIdx;
    } // if p
    else
    {
      repeatPdf = true;
    } // else p
    // Extract underlying pdf
    Pdf_N* pdfPtr;
    pdfPtr = pdfClsPtr->pdf(oldPdfId);
```

Stellenbosch University http://scholar.sun.ac.za

```
const MixturePDF_N* mixPdfPtr;
mixPdfPtr = dynamic_cast<const MixturePDF_N*> (pdfPtr);
if (mixPdfPtr)
{
  // Create new pdfs
  if (largestMixSize == 1)
  {
    // Split mixtures into single pdfs
    old2new[state].resize(mixPdfPtr->noOfPdfs() );
    // Only use when pdf is repeated
    int repeatMixOffset = 0;
    for (int i = 0; i < mixPdfPtr->noOfPdfs(); i++)
    {
      if (!repeatPdf) // first occurence of this pdf
      {
        newPdfs[newPdfIdx] = mixPdfPtr->pdf(i);
        newClassIndexes[newStateIdx] = newPdfIdx++;
      } // if !repeatPdf
      else // repeated occurence of this pdf
      {
        newClassIndexes[newStateIdx] =
          new2oldPdfId[oldPdfId] + repeatMixOffset++;
      } // else !repeatPdf
      tmpWeights[newStateIdx] = mixPdfPtr->w(i);
      old2new[state][i] = newStateIdx;
      new2old[newStateIdx++] = state;
    }// for i
  }// if largestMixSize
  else
  {
```

Stellenbosch University http://scholar.sun.ac.za

```
// Split mixtures into a number of smaller mixtures
int totalPdfs = mixPdfPtr->noOfPdfs();
int pdfsPerNewState =
      ( (totalPdfs - 1) / largestMixSize) + 1;
int mixPdfIdx = 0;
old2new[state].resize(pdfsPerNewState);
// Only use when pdf is repeated
int repeatMixOffset = 0;
for (int np = 0; np < pdfsPerNewState; np++)
{
  if ( (totalPdfs -= largestMixSize) >= 0)
  {
    Vector<RCPtr<PDF_N> > tmpPdfs(largestMixSize);
    Vector<REAL> weights(largestMixSize);
    for (int i = 0; i < largestMixSize; i++)
    {
      tmpPdfs[i] = mixPdfPtr->pdf(mixPdfIdx);
      weights[i] = mixPdfPtr->w(mixPdfIdx++);
    }// for i
    tmpWeights[newStateIdx] = weights.sum();
    // Normalize weights
    weights /= weights.sum();
    if (!repeatPdf) // first occurence of this pdf
    {
      newPdfs[newPdfIdx] =
         new MixturePDF_N(tmpPdfs,weights);
      newClassIndexes[newStateIdx] = newPdfIdx++;
    } // if !repeatPdf
    else // repeated occurence of this pdf
    {
      newClassIndexes[newStateIdx] =
```

Stellenbosch University http://scholar.sun.ac.za

```
            new2oldPdfId[oldPdfId] + repeatMixOffset++;
        } // else !repeatPdf
        old2new[state][np] = newStateIdx;
        new2old[newStateIdx++] = state;
      }// if totalPdfs
      else
      {
        if ( (totalPdfs += largestMixSize) > 1)
        {
          // Mixture
          Vector<RCPtr<PDF_N> > tmpPdfs(totalPdfs);
          Vector<REAL> weights(totalPdfs);
          for (int i = 0; i < totalPdfs; i++)
          {
            tmpPdfs[i] = mixPdfPtr->pdf(mixPdfIdx);
            weights[i] = mixPdfPtr->w(mixPdfIdx++);
          }// for i
          tmpWeights[newStateIdx] = weights.sum();
          // Normalize weights
          weights /= weights.sum();
          if (!repeatPdf) // first occurence of this pdf
          {
            newPdfs[newPdfIdx] =
                new MixturePDF_N(tmpPdfs,weights);
            newClassIndexes[newStateIdx] = newPdfIdx++;
          } // if !repeatPdf
          else // repeated occurence of this pdf
          {
            newClassIndexes[newStateIdx] =
              new2oldPdfId[oldPdfId] + repeatMixOffset++;
          } // else !repeatPdf
```

Stellenbosch University http://scholar.sun.ac.za

```
            old2new[state][np] = newStateIdx;
            new2old[newStateIdx++] = state;
          }// if totalPdfs
          else // single pdf
          {
            if (!repeatPdf) // first occurence of this pdf
            {
              newPdfs[newPdfIdx] = mixPdfPtr->pdf(mixPdfIdx);
              newClassIndexes[newStateIdx] = newPdfIdx++;
            } // if !repeatPdf
            else // repeated occurence of this pdf
            {
              newClassIndexes[newStateIdx] =
                new2oldPdfId[oldPdfId] + repeatMixOffset++;
            } // else !repeatPdf
            tmpWeights[newStateIdx] = mixPdfPtr->w(mixPdfIdx++);
            old2new[state][np] = newStateIdx;
            new2old[newStateIdx++] = state;
          }// totalPdfs
        }// else totalPdfs
      }// for np
    }// else largestMixSize
  } // if mixPdfPtr
  else // Single pdf
  {
    /*
     * First search the new2oldPdfId map to see if the old pdf
     * id had occured before if not, create a new pdf using
     * the new pdf id and store this information in map
     * otherwise extract the new pdf id corresponding to the
     * old pdf id and assign new class index.
```

Stellenbosch University http://scholar.sun.ac.za

```
        */
       // map<int,int>::iterator p = new2oldPdfId.find(oldPdfId);
       if (!repeatPdf) // the old pdf id has not occured before
       {
         // create a new pdf by extracting it from the old pdf
         // classifier
         newPdfs[newPdfIdx] = pdfPtr;
         // assign the new pdf id to the new class index
         newClassIndexes[newStateIdx] = newPdfIdx++;
         // add the key-value pair to the new2oldPdfId map
         // new2oldPdfId[oldPdfId] = newPdfIdx++;
       } // if p == new2oldPdfId.end()
       else // the old pdf id occured before
       {
         /*
          * No need to create a new pdf, just extract the id from
          * new2oldPdfId map and assing to the value to the new
          * class index
          */
         newClassIndexes[newStateIdx] = new2oldPdfId[oldPdfId];
       } // else p != new2oldPdfId.end()

       tmpWeights[newStateIdx] = 1.0;
       old2new[state].resize(1);
       old2new[state][0] = newStateIdx;
       new2old[newStateIdx++] = state;
     }// else mixPdfPtr
   }// else oldClassIndexes
}// for state

// Create new pdfClassifier
```

Stellenbosch University http://scholar.sun.ac.za

```
cout << "Create new pdfClassifier." << endl;
BankOfSimilarities_TxN* tmpClassPtr = new pdfClassifier(newPdfs);

/*
 * Count the number of new Destination links and link weights
 * Use the new to old mapping , old to new mapping and the old
 * destination links to count them
 */
Vector<Vector<int> > newDest(numberOfNewStates);
Vector<Vector<REAL> > newDestWeight(numberOfNewStates);
for (int count = 0; count < newDest.size(); count++)
{
  int newCount = 0;
  for (int oldCount = 0;
       oldCount < oldDest[new2old[count]].size();
       oldCount++)
  {
    newCount += old2new[oldDest[new2old[count]][oldCount]].size();
  }// oldCount
  newDest[count].resize(newCount);
  newDestWeight[count].resize(newCount);
}// for count

/*
 * Calculate the new Destination links
 * Use the old to new mapping and the old destination links
 * to create the new destination links
 */
for (int count = 0; count < newDest.size(); count++)
{
  int newCount = 0;
```

Stellenbosch University http://scholar.sun.ac.za

```
    for (int oldCount = 0;
         oldCount < oldDest[new2old[count]].size();
         oldCount++)
    {
      int oldSize =
        old2new[oldDest[new2old[count]][oldCount]].size();
      for (int numOld = 0; numOld < oldSize; numOld++)
      {
        newDest[count][newCount] =
          old2new[oldDest[new2old[count]][oldCount]][numOld];
        newDestWeight[count][newCount] =
          oldDestWeight[new2old[count]][oldCount] *
          tmpWeights[newDest[count][newCount]];
        newCount++;
      }// numOld
    }// oldCount
    // normalize the weights
    if (newDest[count].size() > 0)
    {
      newDestWeight[count] /= newDestWeight[count].sum();
    }// newDest
  }// for count

  // Create new HMM model
  cout << "Create the new HMM model." << endl;
  newMod.~HMM();  //destroy existing
  new( (void*) &newMod) HMM(newDest,
                            newDestWeight,
                            modlId,
                            newClassIndexes,
                            tmpClassPtr);
```

Stellenbosch University http://scholar.sun.ac.za

```
  cout << "Expansion complete!" << endl;
} // hmm_mixture_expand
```