Gaussian Mixture Model Based Coding of Speech and Audio

Sam Vakil

Department of Electrical & Computer Engineering
McGill University
Montreal, Canada

October 2004

A thesis submitted to McGill University in partial fulfillment of the requirements for the degree of Master of Engineering.

Your file Votre référence
ISBN: 0-494-06592-3
Our file Notre référence
ISBN: 0-494-06592-3

Abstract

The transmission of speech and audio over communication channels has always required speech and audio coders with reasonable search and computational complexity and good performance relative to the corresponding distortion measure.

This work introduces a coding scheme which works in a perceptual auditory domain. The input high dimensional frames of audio and speech are transformed to power spectral domain, using either DFT or MDCT. The log spectral vectors are then transformed to the excitation domain. In the quantizer section the vectors are DCT transformed and decorrelated. This operation gives the possibility of using diagonal covariances in modelling the data. Finally, a GMM based VQ is performed on the vectors.

In the decoder part the inverse operations are done. However, in order to prevent negative power spectrum elements due to inverse perceptual transformation in the decoder, instead of direct inversion, a *Nonnegative Least Squares Algorithm* has been used to switch back to frequency domain. For the sake of comparison, a reference subband based "Excitation Distortion coder" is implemented and comparing the resulting coded files showed a better performance for the proposed GMM based coder.

Sommaire

La transmission de la parole et de l'audio sur des canaux de communication a toujours exigé des codeurs de la parole et de l'audio avec une recherche et complexité d'exécution raisonnable et d'une bonne performance relative à la mesure de déformation correspondante.

Ce travail présente une méthode de codage qui fonctionne dans un domaine auditif perceptuel. Les fenêtres d'entrée de dimensions élevés de la parole et de l'audio sont transformées au domaine de la puissance spectrale, en utilisant soit la DFT ou la MDCT. Les vectors spectraux logarithmiques sont alors transformés au domaine de l'excitation. Dans la section du quantificateur, les vecteurs sont transformś et décorrelés en utilisant la DCT. Cette opŕation donne la possibilité d'employer des covariances diagonales pour modeler les données. En conclusion, un VQ basé sur GMM est exécuté sur les vecteurs.

Dans la partie du décodeur, les opérations inverses sont faites. Cependant, afin d'empêcher les éléments négatifs du spectres de puissance dus à la transformation perceptuelle inverse dans le décodeur, au lieu d'utiliser l'inversion directe, un algorithme du *moindres carrés non-négatifs* a été employé pour commuter au domaine de la fréquence. À des fins de comparaison, un " codeur de déformation d'excitation " basé sur une référence sous-bande est implèmentè et ensuite comparè à l'algorithme proposé. Les fichiers codés résultants ont montré une meilleure performance pour le codeur proposé, basé sur les GMM .

Acknowledgments

I firstly, would like to express my thanks to my supervisor Professor Peter Kabal for his invaluable advice and support during my graduate studies at McGill. I have no doubts that finishing my degree in a proper and timely manner was impossible without his helps, suggestions and advices. I am also grateful to Professor Richard Rose for his help during writing my thesis, specially about the parts which were related to concepts of speech recognition.

I sincerely thank my fellow colleagues of the Telecommunication and Signal Processing Lab for their technical support and friendship. Specially, I have to mention Ricky Der for his indispensable help and suggestion concerning the concepts of perceptual audio coding. Not to mention Alex for his technical support in TSP Lab. Tania's French translation of the abstract is much appreciated.

Finally, I am grateful to my family for their love and support during my graduate studies at McGill. They have encouraged me, loved me and always supported me. Thank you.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Building coders with acceptable search and computational complexity that provide good performance for a given distortion measure has always been the goal of coder design. The ability to train a coder based on a set of data and to find a proper distortion measure are among the key issues in this design.

In this chapter the concepts of Audio Coding, Vector Quantization and Source Modelling are overviewed and different existing classes of audio coders are explained briefly. Finally the motivation for the work done in this thesis is discussed.

1.1 Source Coding

Source coding is a branch of Information Theory which has been first introduced by Claude Shannon in 1948. Source coding, tries to solve the problem of representing a source in the best way. In this theory a source of information is modelled using a stochastic process. The messages are the outcomes of this stochastic process and are produced according to a certain probability [6].

The basic block diagram of a source coder is depicted in Figure 1.1. The encoder maps

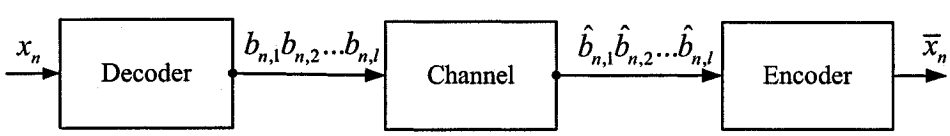

Fig. 1.1 Block Diagram of an Encoder Decoder scheme

each of the symbols x_n to a sequence of bits $b_n = b_{1,n}...b_{l,n}$. This sequence is then passed

through the communication channel and the resulting sequence $\hat{b}_n = \hat{b}_{1,n}...\hat{b}_{l,n}$ is decoded to $\hat{x}_n$ at the other end.

1.2 Audio Coding

Audio coders are generally divided in to three classes; *parametric coders*, *hybrid coders* and *waveform coders*. Parametric coders do not tend to reconstruct the waveform of the original signal. Instead, they code some parameters of the signal, send them and finally reconstruct the signal. This class of coders works at low bit rates at the expense of losing some performance precision. Waveform coders, try to generate a compressed signal which has a waveform similar to the original signal. They have better performance, but need higher bit rates.

1.2.1 Classes of Audio Coders

In the following the performance of these coders will be further briefly explained.

Parametric coders, *vocoders* or source coders model the input source with a few parameters. For speech signal, there has been a good representation, which models the vocal tract as a time varying filter. This filter is excited by a source which is modelled using a white noise source (for unvoiced signal) or a sequence of responses (for voiced signal).

Waveform coders, try to reconstruct the waveform of the original signal. These coders work well for coding of audio signals, because there are no good models which represent the audio source. In terms of the signal domain these coders lie in to two categories: Time domain and frequency domain.

- **Time domain coders:** The coding is done on the time samples of the signal. Some examples of this type of coding are [7]: Pulse Code Modulation (PCM), Adaptive Pulse Code Modulation (APCM), Differential Pulse Code Modulation (DPCM), Delta Modulation (DM), Adaptive Delta Modulation (ADM) and Adaptive Predictive Coding (APC).

- **Frequency Domain Coders:** The coding is done on the frequency samples of the signal. These coders produce coded speech with a higher quality than time domain coders. However, this is at the expense of a higher complexity [8]. One advantage of

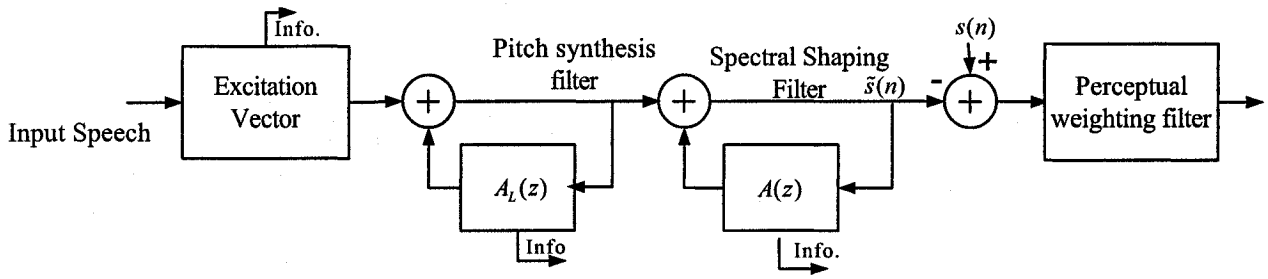

Fig. 1.2 Block Diagram of a CELP coder [1]

frequency domain coders over time domain coders is their ability to consider factors relating to human perception. Perceptual models of human hearing have been well modelled in the frequency domain. Therefore, if we use frequency domain coders we can benefit from this fact in the design of our coder.

Frequency domain coders fall in to two categories: *subband coders* and *transform coders*. Subband coders use bandpass filters to code the input signal in different frequency bands. Transform coders, use some specific transform on the frames of the input data. Perceptual audio coding is done using transform coders [9]. Some of the benefits of using this scheme are: compacting the signal energy into fewer coefficients, decorrelating the data coefficients and ease of using perceptual distortion measures.

Hybrid coders, tend to use both the benefits of waveform coders and parametric coders. They extract the parameters of the input signal and send it. Hybrid coders as well, transmit an error signal which is the difference between the input and output signal after the extraction of parameters.

An example of the most successful Hybrid coders for speech is Code Excited Linear Predictive (CELP) coder. Figure 1.2 depicts the block diagrams of a CELP coder. The two synthesis filters are used to model the long and short term parameters of speech. The parameters of these filters are determined using a perceptually weighted difference between the input and output signals.

1.2.2 Perceptual Audio Coding

Perceptual audio coders use the concept of *Auditory masking threshold* to quantize the signal. They use a transformation to represent the spectrum of the signal. Using this spectrum and models of human perception, the masking threshold is determined for different bands.

The bit assignment is then done using these thresholds and finally after multiplexing the quantized parameters with side information a bit stream is produced. Block diagram of a generic audio coder is given in Fig. 1.3. The lossless entropy coding system is able to perfectly reconstruct the samples of the original signal from the coded representation.

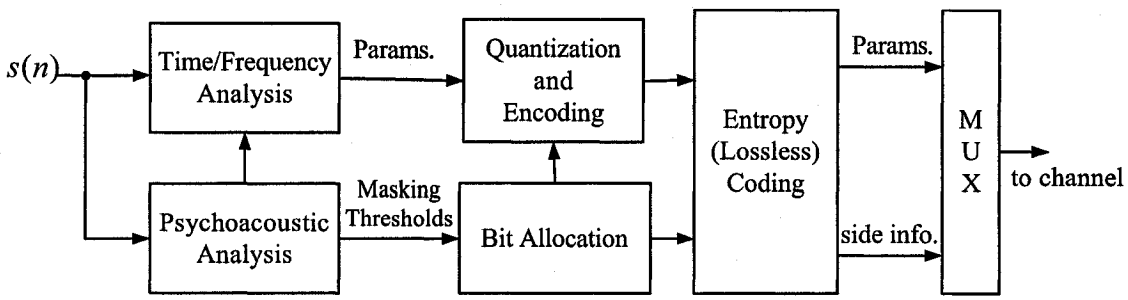

Fig. 1.3 Generic perceptual audio coder [1]

1.3 Vector Quantization

The problem of source coding refers to the ways of converting a source to a sequence of bits and sending them over a communication channel and then reproducing the sent signal. In speech communication distortion in the reproduced signal is tolerated to some extent. The amount of distortion is determined by the bit rate or the number of bits used to represent the source signal.

1.3.1 Quantization

Quantization is referred to as the process of approximating a continuous signal by a set of discrete values. Vector Quantization in contrast to scalar quantization, jointly quantizes the multiple signal values which are called vectors. In Vector Quantization (VQ) applications, the quantizer design is usually based on training with a set of representative training vectors [10]. VQs are described by a set of codewords. Codeword is a set of fixed vectors which is used to reproduce the quantized signal. The input vectors which are to be quantized are compared to the vectors in this set and matched to each one. The index of the codeword which minimizes the distortion measure is sent over the channel. *Vector Quantization* is one of the most efficient source coding schemes, which reduces the number of bits needed to represent the signal or in other words compresses the data.

1.3.2 Distortion Measures

A distortion measure d is the cost defined for quantizing a vector $\mathbf{x}$ with a vector $\hat{\mathbf{x}}$. The most widely distortion measure used between the input vector $\mathbf{x}$ and the output vector $\hat{\mathbf{x}}$ is the squared error or squared Euclidean distance between the two vectors. Weighted squared error is an example of another distortion measure which uses a matrix $\mathbf{W}$ as the weighting matrix for the distance and the distance between two vectors $\mathbf{x}$ and $\mathbf{y}$ is defined as $d(\mathbf{x},\mathbf{y}) = (\mathbf{x}-\mathbf{y})^{\mathrm{t}}\mathbf{W}(\mathbf{x}-\mathbf{y})$.

The number of vectors in a code book determine the size of that code book. The size of the code book also shows the number of partitions in the codebook. For a codebook of size M the d-dimensional space of the original vector $\mathbf{x}$ is divided to M cells, and each of the cells corresponds to a codeword vector. The representation of VQ as a Look-up table is shown in Figure 1.4.

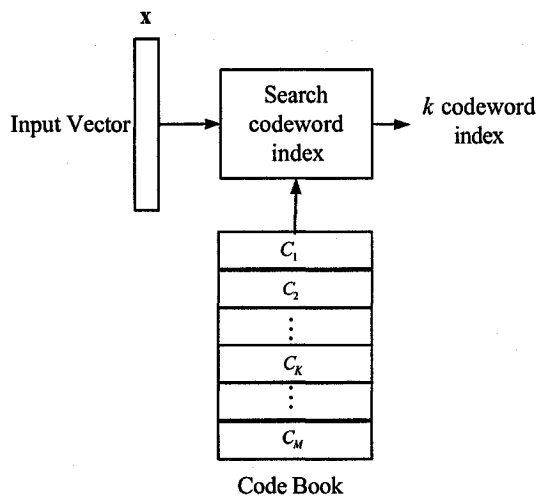

Fig. 1.4 Vector Quantization

The search for the nearest vector is done exhaustively, by finding the distance between the input vector and each of the codewords and the one with the smallest distance is coded as the output. This process is shown in Fig. 1.5.

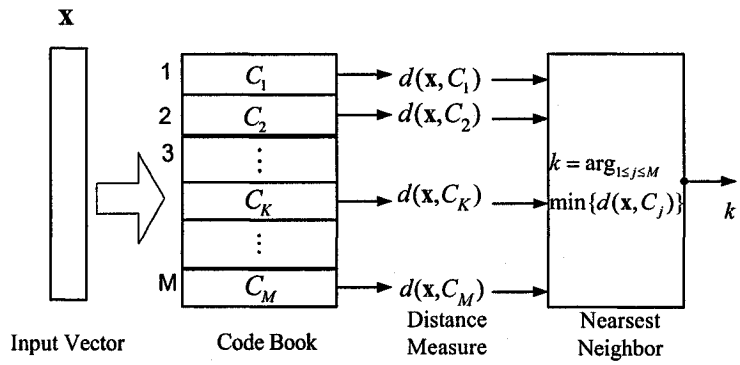

Fig. 1.5 VQ codeword selection

1.4 Statistical Source Modelling

VQs are usually designed using an empirical model. The goal in this design is training a codebook from a set of training vectors [10]. One problem happening here, is the size of the training set, which should be relatively large to the codebook size [11], otherwise the problem of over-fitting will be encountered, which is explained in Chapter 3. Therefore, the need to predict the possible *training vectors* arises here. Knowing the probability density function (*pdf*) of the vectors in this set, we can solve this problem.

1.4.1 Gaussian Mixture Modelling

It is usually impossible to model the distribution of speech or audio signal using a standard density function like a Gaussian. On the other hand, the family of finite mixture densities are one type of distributions, usually used for this parameter estimation. Recently a VQ design has been proposed in [4]. This scheme, in contrast to previous coders does not use the traditional training, instead it designs the VQ using the parameters obtained by modelling the *pdf* of the input speech signal. This transform coding scheme uses the first and second order statistics of the signal to normalize and decorrelate it. It finally uses scalar quantization for the individual decorrelated coefficients.

1.5 Description of Thesis Work

In this thesis we have implemented a new coder in the excitation domain. The aim is to minimize a perceptually defined distortion measure. Therefore, the human perception is considered in defining the distortion measure which will give a higher quality of regenerated signal at a lower bit rate.

Using proper transformation on the spectrum of the signal, a representation of the signal in the Excitation Domain will be given. This transformation includes the outer/middle ear filtering and frequency spreading. The transformed vectors are in the excitation domain.

The GMM-based quantization scheme used here, benefits from the coding scheme proposed in [4], however we have quantized high dimensional excitation patterns instead of using low dimensional LPC coefficients. The main contribution here, will be the use of a perceptual distortion measure and at the same time diminishing the quantizer complexity by the use of Gaussian Mixture Models in training the data.

Since high dimensional data vectors are used, we take advantage of using the GMM based VQ in order to exploit the statistical dependency between different elements of a frame. Use of high dimensional data vectors leads to need for building bigger training data bases. We also had to made slight modifications to the EM algorithm using the method proposed in [12].

Another difference between our scheme and the one proposed in [4] is that we have used both speech and narrow band audio signals in a data base in order to train a set of Gaussian Mixtures for implementation of the quantizer. Therefore, we could use the quantizer for the coding of both narrow band audio and speech files. However, we need a higher bit rate than speech for audio to have the same acceptable quality.

The majority of audio coders introduced in literature are doing the process in a sub-optimal way in the sense that, they divide a quantity in power spectrum domain by the masking which is defined in the excitation domain. The current coder avoids dealing with quantities in different domains by using the perceptual transformation which will be discussed in Chapter 2.

The other important issue to be considered is the invertibility of the perceptual transformation. Johnston has introduced this concept in [13]. In this work a perceptual transform coder is introduced. The power spectrum components are convolved with a spreading function to switch to the excitation domain. A noise masking threshold is then subtracted

from the spread critical band spectrum which gives the spread threshold. The necessity of inverting the spread threshold back to Bark domain is mentioned in Johnston's work. The trouble here is that after subtracting the masking threshold, the resulting components in the excitation domain are no further invertible to power spectrum domain, since an offset has been subtracted from the original value. Johnston proposes a renormalization process to compensate for the effect of spreading function. In this case, the deconvolution is very unstable and may result in negative power spectrum, therefore it can not directly be used as the inverse transformation. This concept has been discussed more in [14].

The same problem may occur in the design of a quantizer which works in the excitation domain. This means, using the inverse pereptual transform we may no longer wind up with positive power spectrum values, since the quantized values are different from the original values. Therefore, we may wind up with negative power spectrum elements. In order to avoid this problem we have used a *Non Negative Least Squares* algorithm to find the best approximation to the inverse function of the excitation transformation. The details of implementation are further discussed in Chapter 4.

Finally, an Excitation Domain (ED) reference coder has been implemented and its performance is evaluated and compared to the proposed coder.

The basic idea behind the GM quantizer given here is the use statistics of the vectors to build a transform coder, decorrelating the data using these statistics and benefitting from the perceptual properties of speech in defining the distortion measure.

1.6 Thesis Organization

This thesis consists of 5 chapters. Chapter 2 will be a discussion on the concepts of Perceptual audio coding. We will briefly review the concepts of frequency domain to excitation coding, critical bands, masking threshold, time frequency filter banks and other related ideas used in perceptual audio coding. We will then, explain how the perceptual transformations could be used in order to change the domain in which we are coding and switch to a domain called the Excitation Domain. The main contribution of this work is that, the coding is done in a different domain which considers human perception.

Chapter 3 will introduce the concept of statistical modelling, Gaussian Mixture Models and their use in the world of speech coding. GMM-VQs will be explained and we will go over a literature review of the works done using the GMMs in the world of speech coding

and pattern recognition.

Chapter 4 discusses the new speech/narrow band audio scheme that has been introduced in this work. The ideas behind using Gaussian mixtures to model the data and perceptual transformation to transform data vectors have been discussed. The results of our implementations have been included in this chapter. We have also implemented an Excitation Domain (ED) reference coder based on subband coding and iteratively changing the step size of the quantizer, in Appendix A.

Finally, Chapter 6 will be an overview and conclusion of the work that has been done.

Chapter 2

Perceptual Audio Coding

2.1 Principles of Psychoacoustics

In order to have an optimum fidelity for audio and speech coders there is a need for coders to optimize their performance relative to the human ear. Most current audio coders compress the signal using the fact that "irrelevant" signal information will not be perceived by listeners [2]. The irrelevant information is identified using psychoacoustic principles such as absolute thresholds of hearing, critical band frequency analysis and different types of masking. Using these psychoacoustical notions along with signal quantization theory, the concept of perceptual entropy has been introduced. Perceptual entropy can be used to determine a fundamental limit of transparent audio compression.

In order to have a better understanding of hearing threshold we have to define a metric to quantify the audio intensity. This metric which is called *sound pressure level* (SPL) gives the level of sound in (dB) compared to a reference level, i.e. $L_{\mathrm{SPL}} = 20\log_{10}(p/p_0)$ dB. L_{SPL} is the SPL of a stimulus [15], p is the sound pressure in Pascals and p_0 is the reference level which is 20 μPa.

2.1.1 Absolute Threshold of Hearing

The absolute threshold of hearing is the least amount of energy needed in a pure tone at a specific frequency such that a listener can detect the tone. It is reported in [2] that Fletcher has given a formula for the frequency dependence of this quantity. The approximation to

this quantity is given by a nonlinear function [16]

$$T_q(f) = 3.64(f/1000)^{-0.8} - 6.5e^{-0.6(f/1000-3.3)^2} + 10^{-3}(f/1000)^4. \quad (2.1)$$

Figure 2.1 is a graph of this threshold versus frequency.

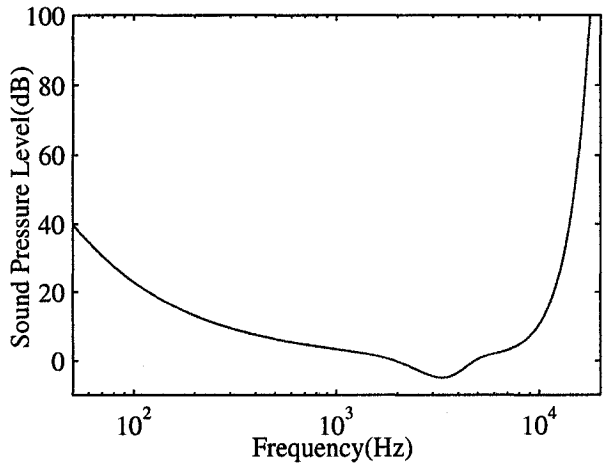

Fig. 2.1 The absolute threshold of hearing in quiet

2.1.2 Critical Bands

The absolute threshold of hearing is not the only factor that is considered in the coding context. When we have a spectrally complex quantization noise, the detection threshold will depend on the stimuli at any time. Therefore, this threshold is a time varying function of the input signal. The performance of the ear in spectral analysis is of considerable importance here. A frequency to place transformation occurs in the cochlea along the basilar membrane [15]. Mechanical movements are transformed to travelling wave in the cochlea. For sinusoidal stimuli, the travelling wave on the basilar membrane propagates from the oval window until a region which has the same resonant frequency [2]. At this point the magnitude increases to a peak and this location is called "characteristic frequency". From a signal processing perspective, in this transformation the cochlea can be viewed as a bank of overlapped bandpass filters. The magnitude responses are nonuniform functions of frequency. Besides that, the filter passbands have nonlinear bandwidths which increase by frequency. The "critical bandwidth" is a function of frequency that measures the width

of the cochlear filter passbands. There have been many interpretations of critical bands. One interpretation is the perceived level of a noise. For a narrowband noise the perceived loudness remains constant for a constant SPL even when the noise bandwidth is increased up to the critical band width. If we increase the bandwidth beyond the critical bandwidth, the loudness begins to increase. Another notion is the audibility threshold of a narrowband noise masked with two tones. The detection threshold for the noise remains constant as long as the frequency separation between the tones is within one critical bandwidth. Beyond this bandwidth the threshold rapidly decreases.

Critical bandwidths are constant up to 500 Hz. In practical systems it is useful to treat ear as a bank of discrete filters. Table 2.1 gives such a filter bank.

Table 2.1 Critical band filter bank [5]

Band No.	Center Freq. (Hz)	Bandwidth (Hz)	Band No.	Center Freq. (Hz)	Bandwidth (Hz)
1	50	0–100	14	2150	2000-2320
2	150	100–200	15	2500	2320–2700
3	250	200–300	16	2900	2700–3150
4	350	300–400	17	3400	3150–3700
5	450	400–510	18	4000	3700–4400
6	570	510–630	19	4800	4400–5300
7	700	630–770	20	5800	5300–6400
8	840	770–920	21	7000	6400–7700
9	1000	920–1080	22	8500	7700–9500
10	1175	1080–1270	23	10500	9500–12000
11	1370	1270–1480	24	13500	12000–15500
12	1600	1480–1720	25	19500	15500–
13	1850	1720–2000			

A distance of one critical band is called "a Bark". The function

$$g(f) = 7\sinh^{-1}(f/650) \tag{2.2}$$

is usually used to convert from frequency in Hertz to the Bark scale. In addition to the representation in Table 2.1 there are alternative expressions for critical bandwidth. Specifically, the equivalent rectangular bandwidth (ERB) which has been used after the results of research directed toward measurement of auditory filter shapes. The models used here are called *Rounded exponential models.* These models are obtained by fitting the masking data with parametric weighting functions which represent the spectral shape of the auditory system. We have implemented the smearing functions which are used for the frequency to excitation domain transforms using this model which is also called "roex". A single parameter "roex" is given as,

$$W(g) = (1+pg)e^{-pg} \tag{2.3}$$

where W is the smearing function, g is the normalized frequency and is equal to $\frac{|f_c-f|}{z}$, f_c is the center frequency, f is frequency in hertz and z is the bark frequency. This filter representation is in the symmetric form. The experimental ERB measurements for *roex* models have been given by Moore and Glasberg in [17] and [18]. The details of the parameters used in our implementation are discussed in Section 2.3.

2.1.3 Masking

Masking refers to a process where one sound becomes inaudible because of the presence of another sound. Masking threshold is used in the design of conventional audio coders. In this work we have used a transformation from power spectrum to excitation domain instead of using masking thresholds.

Simultaneous masking occurs when two or more stimuli enter the auditory system at the same time. The relative spectral magnitude of the masker and the maskee determines the effect of one spectral energy in masking another one. As examples of simultaneous masking we can mention: *Noise-Masking-tone* (NMT), *tone-masking-noise* (TMN) and *noise-masking-noise* (NMN). For a more thorough explanation the interested reader is referred to [19].

Non-simultaneous masking occurs whenever the masking in time extends beyond the window of simultaneous stimulus presentation. For a finite duration masker the masking effect occurs both before masker onset and after.

2.2 Time-Frequency Analysis Filter Banks

Audio codecs need some type of time-frequency analysis. In order to employ the characteristics needed in perceptual audio coding it is needed to extract a frequency representation of the input signal. The most widely used means to obtain this goal is the notion of filter bank, which is a parallel bank of bandpass filters. A filter bank, divides the signal spectrum in to frequency subbands and a time series in each band which demonstrates the frequency localized signal power within each band. If M filters are used each will be characterized by a time domain impulse response $h_k(n)$ as well as a frequency response $H_k(\omega)$. Filter banks can be described using analysis synthesis frame work (Fig. 2.2). The input signal $s(n)$

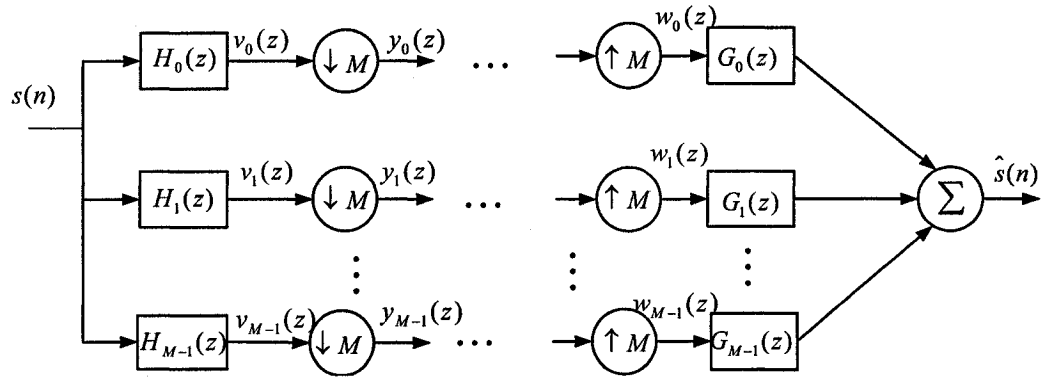

Fig. 2.2 Uniform M- band maximally decimated analysis-synthesis filter bank [2]

is processed by a band of $(L-1)$ th order finite impulse response (FIR) bandpass filters $H_k(z)$. The bandpass outputs are

$$v_k(n) = h_k(n) * s(n) = \sum_{m=0}^{L-1} s(n-m)h_k(m), \quad k = 0, 1, \ldots, M-1 \tag{2.4}$$

Decimation is done after this stage which gives $y_k(n) = v_k(Mn)$. It is impossible to implement ideal band pass filters. Therefore, aliasing is unavoidable. Quantization is done

on the subband sequences $y_k(n)$. The quantized version of the signal $\hat{y}_k(n)$ is up sampled by M. Finally, a bank of synthesis filters $G_k(z)$ is used in parallel and the overall output $\hat{s}_n$ is formed [2]. Perfect reconstruction occurs when $\hat{s}(n) = s(n)$. It is shown in [20] that having the analysis window $w_a(n)$ and the synthesis window $w_s(n)$ and assuming no quantization noise the condition of perfect reconstruction is

$$\sum_{k=-\infty}^{\infty} w_a(n - kL)w_s(n - kL) = 1. \tag{2.5}$$

The quantity L is the transform block advance of the windows. Defining the new window $w(n) = w_a(n)w_s(n)$, we can express the above expression in the frequency domain as

$$W(\frac{2\pi l}{L}) = \begin{cases} 1, & l = pL \\ 0, & \text{otherwise} \end{cases}$$

Since, usually quantization noise is introduced and $y_k(n) \neq \hat{y}_k(n)$ perfect reconstruction can not be obtained. *MDCT* and DFT are two specific filterbanks which have been considered. The details of design consideration and *MDCT* characteristics follow.

2.2.1 Design Considerations

In order to have an efficient coding performance we need our filter bank to be matched to signal properties. One of the difficulties encountered, is the trade off between time and frequency resolutions. Most audio source signals are stationary for a long period (a few frames) and then they will have an abrupt change [21]. Therefore, the optimal decision concerning the time frequency decomposition should be made adaptively. Some of the desired characteristics for filter banks are

- perfect reconstruction: A perfect reconstruction filter bank decomposes a signal by filtering and subsampling. It reconstructs it by inserting zeroes, filtering and summation.
- signal adaptive time-frequency tiling: Since speech is a non stationary signal, the frequency characteristics change over time. Therefore, a time-frequency representation of this signal is desired.
- good channel separation.

- strong stop band attenuation.
- critical sampling: Critical sampling means that the amount of data (samples per second) remains the same in the transformed domain.

In the next section MDCT, a Perfect Reconstruction cosine-modulated filter bank is discussed more.

2.2.2 Cosine Modulated PR Filter Banks and *MDCT*

Perfect reconstruction in filter banks has been popular. Works done by Malvar [22], Ramstad [23] and Koilpillai have proved that implementation of generalized PR cosine filter banks is possible. This section is more concentrated on a specific type of PR filter banks. Here the order "L" of each impulse response is twice the number of filter banks "M". This filter bank was formulated as a lapped orthogonal block transformation by Malvar. However, recently in the world of audio coding it has been known as modified discrete cosine transform or "MDCT". The MDCT analysis filter impulse responses are given by

$$h_k(n) = w(n)\sqrt{\frac{2}{M}}\cos\left[\frac{(2n+M+1)(2k+1)\pi}{4M}\right]. \tag{2.6}$$

The synthesis filters are obtained by a time reversal,

$$g_k(n) = h_k(2M-1-n). \tag{2.7}$$

In practice filter banks are realized as a block transform. Below, we have explained the process to obtain the MDCT coefficients from the input signal as well as the inverse transformation, this process has been depicted in Fig. 2.3 as well.

- *MDCT Realization*: The analysis filter bank is realized using a block transform of length $2M$ with an overlap of 50% between the blocks. Despite this overlap MDCT is critically sampled. Given the input block $x(n)$ the coefficients of MDCT transform $X(k)$, for $0 \leq k \leq M-1$ are

$$X(k) = \sum_{n=0}^{2M-1} x(n)h_k(n). \tag{2.8}$$

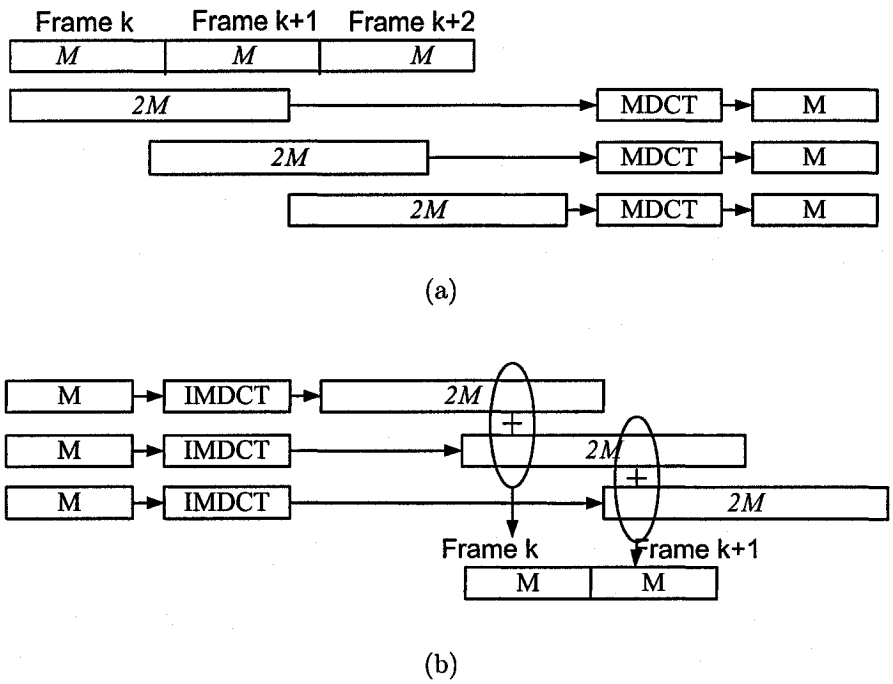

Fig. 2.3 $2M$ samples mapped to M coefficients a) Forward MDCT transform analysis with 50% overlap. b)The inverse transform [2]

This sum has the same form as the DFT of the signal $x(n)$, but the basis functions are changed and the summation expands over $2M$ coefficients. The inverse MDCT obtains the reconstruction by computing a sum of the basis vectors which are weighted by the transform coefficients of two blocks. The first M samples of $h_k(n)$ are multiplied by $X(k)$ (coefficients of the current block). However, the second M samples are weighted by $X^P(k)$ (the coefficients of the previous block). Therefore, the inverse transform which gives $x(n)$ for $0 \leq n \leq M-1$, is defined as

$$x(n) = \sum_{k=0}^{M-1} [X(k)h_k(n) + X^P(k)h_k(n+M)] \tag{2.9}$$

The next step is the design of proper window $w(n)$. As shown in [24] the linear phase and Nyquist conditions can be reduced to,

$$w(2M-1-n) = w(n) \tag{2.10}$$

and

$$w^2(n) + w^2(n+M) = 1 \tag{2.11}$$

- *Popular Windows*: MDCT is realized using a "sine" window. Malvar developed modulated lapped transform (MLT) and formulated the filterbank as a lapped orthogonal transform [25]. The window is defined as [1]

$$w(n) = \sin\left[(n+\frac{1}{2})\frac{\pi}{2M}\right] \tag{2.12}$$

for $0 \leq n \leq M-1$. This type of window has several unique properties. The dc energy is concentrated in just one coefficient. The filter bank channels have 24 dB sidelobe attenuation (The amplitude of the first sidelobe is 24 dB less than the main lobe). The MLT has been used in MPEG-1, Layer 3 hybrid filter bank MP3 and MPEG-2 filter bank [2].

2.3 Perceptual Domain Transforms

The auditory filters are representations of the frequency selectivity of the hearing system at a particular frequency. The excitation pattern refers to the distribution of the excitation evoked by that sound as a function of some internal variable related to place or frequency [17]. In terms of filter banks, these patterns can be seen as the outputs of each filter as a function of filter center frequency. In this thesis, the excitation patterns have been used as the elements to be quantized after some perceptual transformations. In this section we will discuss these transformations more and the quantization scheme will be explained in Chapter 4.

In most models for computation of the excitation patterns, power spectral components are divided to non-overlapping critical bands and then some form of spreading is applied to critical band intensities. The dividing step accounts for intra-band masking and the spreading accounts for masking effects between bands. The models used for the computation of excitation patterns is Patterson and Moore's. It is assumed here that the excitation patterns are derived using a continuous integral of power spectrum weighted by a spreading

[1] We can also use the prototype window introduced in [26]. This is a prototype window for MDCT which considers the frequency response of ear.

function as

$$E(f_c) = \int_0^\infty P(f)W(f, f_c)df, \tag{2.13}$$

The variables f_c and f are both frequency in Hertz, f_c is the frequency center of each smearing function and f is the range of frequency over which we take the integral. $P(f)$ is the input signal power spectrum and $W(f, f_c)$ is the frequency response of the smearing function centered at f_c. These functions have an exponential decaying shape which is multiplied by a linear term in f. The computation to find the inverse transform is intractable. However, simplifying the above equation to a form of discrete matrix operation, we will wind up to a transformation which can be inverted. The distortion measures used in the quantizers working in the power spectral domain is in the L^p norm. Mean square norm is a specific case which is defined as $D(\mathbf{X}, \hat{\mathbf{X}}) = ||\mathbf{X} - \hat{\mathbf{X}}||$ where X and $\hat{X}$ are the original and the quantized version of the power spectrum. Transforming the spectrum to excitation patterns using a transformation matrix T, the distortion measure is no longer defined in metric space. Besides that, we no longer have independent components of distortion in different dimensions. The latter, gives us an easy way of bit allocation in each dimension. The new distortion measure in the perceptual excitation domain is defined as [27]

$$D(\mathbf{X}, \hat{\mathbf{X}}) = ||T(\mathbf{X}) - T(\hat{\mathbf{X}})||. \tag{2.14}$$

Using this transformation we can define metric spaces on $T(\mathbf{X})$ instead of $\mathbf{X}$. This fact, accounts for the implementation of the quantization in the excitation domain so that:

- The distortion is defined as the norm of the difference between the quantized excitation vectors and the original excitation vectors. Therefore, we again have a metric space.
- After decorrelation we can use incremental bit allocation, since the effects of the quantization are local. In other words, we are quantizing in a new domain and we are able to implement the quantization schemes used in power spectrum domain.

2.3.1 Excitation to Power Spectrum Mapping

As mentioned in the previous section the excitation pattern can be extracted by integratiNG the power spectrum weighted by a spreading function. A common spreading function,

usually used is in the form of *roex* [17] parameterization of the auditory filter which is stated as

$$W(f, f_c) = (1 + p\frac{|f_c - f|}{z})e^{-p\frac{|f_c-f|}{z}} \tag{2.15}$$

the parameter p depends on the center frequency f_c and input power $P(f)$. This fact makes Eq. (2.13), an integral nonlinear equation. To simplify the computation the assumption of level independence has been made. Therefore, p is only a function of frequency. The value of p used for simulations is a good approximation for medium levels around 50 dB SPL. Patterson has showed that p can be approximated as $\frac{4f_c}{\text{ERB}(f_c)}$, where $\text{ERB}(f_c) = 24.7(\frac{4.37}{1000}f_c + 1)$. We can, therefore, express the spreading function as

$$W(f, f_c) = (1 + \frac{4}{\text{ERB}(f_c)}|f_c - f|)e^{-\frac{4}{\text{ERB}(f_c)}|f_c-f|} \tag{2.16}$$

While implementing, we only have power spectrum at discrete points. Therefore, we need an approximation to the integral defined in Eq.(2.13).

The frequency axis for one period of the Fourier transform spans over the interval $[0, F_s/2]$. Assuming that we have frequency domain representation vectors of dimension N the frequency difference between two adjacent components will be $\frac{F_s}{2N}$. This replaces the *df* element in the integral. The frequency samples over which the summation is done can be represented as $f_k = \frac{kF_s}{2N}$ and finally we will wind up with the following discrete representation for excitation components

$$E(f_{c_i}) = \frac{F_s}{2N}\sum_{k=0}^{N-1} P(\frac{kF_s}{2N})W(\frac{kF_s}{2N}, f_{c_i}), \qquad \text{for} \qquad 1 \le i \le N \tag{2.17}$$

where c_i is the ith center frequency and $f_{c_i} = \frac{iF_s}{2N}$. In order to simplify the implementation the above equation has been rewritten to matrix representation. We define the $N \times 1$ power spectrum vector $\mathbf{p}_k = P(f_k)$, the $N \times 1$ excitation vector $\mathbf{e}_{c_i} = E(f_{c_i})$ and the $N \times N$ auditory filter matrix $\mathbf{W}_{k,i} = W(f_k, f_{c_i})$, the matrix representation will be in the following form

$$\mathbf{e} = \frac{F_s}{2N}\mathbf{W}\mathbf{p}. \tag{2.18}$$

Since, the transformation matrix is invertible we can define the inverse transform from the

excitation domain to power spectrum as

$$\mathbf{p} = \frac{2N}{F_s}\mathbf{W}^{-1}\mathbf{e}. \tag{2.19}$$

This equation has been used to calculate the excitation patterns of an ideal sinusoid with frequency 1000 Hz. The sampling frequency is 8000 Hz and the dimension of the vector representing the spectrum is $N = 120$. The power spectrum which is represented using DFT has one non zero element. Figure 2.4 depicts the excitation patterns for an interval of 20 to 90 dB SPL of levels.

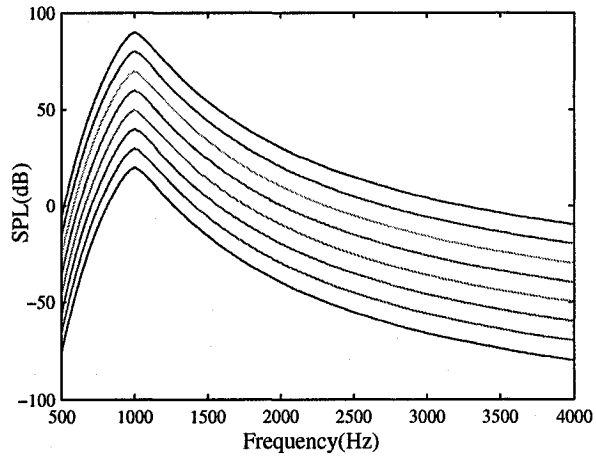

Fig. 2.4 Predicted Excitation Patterns for a 1 kHz tone with levels changing in the interval 20 dB–90 dB SPL

2.3.2 Inverse Transformation

In order to use the transformation from power spectrum to the excitation domain we have to make sure that the inverse transform exists. This assures the possibility of switching back to the power spectrum domain. As mentioned previously the matrix $\mathbf{W}$ is invertible. However, another concern here is to check whether the transformation matrix is ill conditioned or not. Using the following transformation

$$\mathbf{e} = \mathbf{W}^{-1}\mathbf{W}\mathbf{e} \tag{2.20}$$

where **e** is the power spectrum representation of an ideal sinusoid we expect to have an ideal sinusoid at the output for the ideal case. For an input of 30 dB SPL and the frequency 1 kHz the reconstructed power spectrum is depicted in Figure 2.5. The resultant spectrum has its peak at the sinusoid frequency. It can be seen that there is a suppression of at least 150 dB. The simulations have been done using Matlab and double precision with 64 bits has been used. This suppression is enough for our purpose in audio coding.

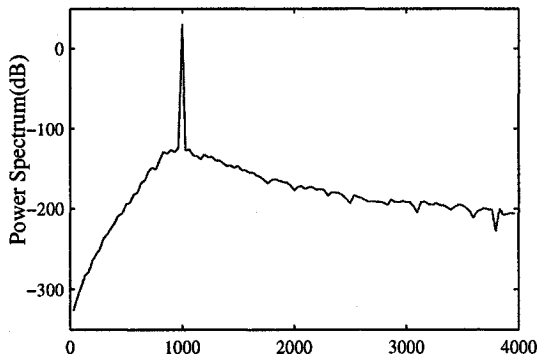

Fig. 2.5 Power spectrum for 30 dB SPL ideal sinusoid after direct and inverse transformations

2.3.3 Outer/Middle Ear Transformation

Another factor considered in the transformation to excitation domain is the outer/middle ear transformation. The ear can be modelled as a weighting filter. Its frequency response as stated in ITU-R B.S. 1387-1 [28] is,

$$A_{\mathrm{dB}}(f_{\mathrm{kHz}}) = -2.184(f/1000)^{-0.8} + 6.5e^{-0.6(f/1000-3.3)} - 0.001(f/1000)^{3.6}. \qquad (2.21)$$

We need to take this transformation into account in order to consider human perception fully in the construction of a perceptual based quantizer. Therefore, the inner outer ear model can be seen as a diagonal matrix **D** which consists of the values given by Eq. (2.21) as the diagonal elements.

2.3.4 Ear Internal Noise

While modelling the ear, we also have to consider an additive element ϵ_{TH} called the ear internal noise which is added to the excitation components. The noise excitation which is considered as an offset is expressed in terms of frequency in a closed form. The Kapust's [29] noise threshold s is related to frequency as,

$$\begin{aligned} s(f)_{\mathrm{dB}} &= (-2 - 2.05 \times \tan^{-1}(f/4000) - 0.75 \times \tan^{-1}(\frac{1}{2.56} \times (f/1000)^2)). \\ s &= 10^{s_{\mathrm{dB}}/10} \end{aligned} \tag{2.22}$$

The ear internal noise is given by Moore's model [30] as,

$$\begin{aligned} I(f) &= 10^{\frac{(1.466 \times (f/1000)^{-0.8})}{10}} \\ E_{\mathrm{th}}(f) &= \frac{I}{s}. \end{aligned} \tag{2.23}$$

The following graph depicts the relation between the internal noise and frequency.

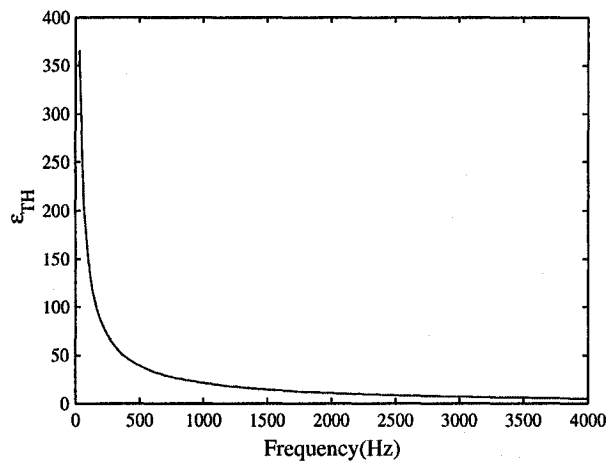

Fig. 2.6 Internal Noise Excitation Pattern given as a function of frequency.

Chapter 3

Gaussian Mixture Models and their use in Vector Quantization

In order to design quantizers it is needed to choose a set of training vectors which properly represents the vectors to be coded in the VQ. Therefore, the training data is chosen such that it has all the underlying data statistics. There are a lot of factors that need to be addressed in VQ design [4] such as: *Computational Complexity, Rate Dependence, Bit-Rate Scalability, Variable Rate Coding and Learning.* In [31], performance bounds for vector quantization (VQ) of speech spectrum source are obtained by combining concepts from high rate quantization and GM modelling and also GM model capabilities in VQ is demonstrated. The concept of GMM has been used in VQ design in [4] and [32]. Use of GMM in VQ design can help us obtain a lot of factors which are mentioned above. Below we describe the concept of Gaussian Mixture Models and mention some examples of its use in signal processing. Also, the design of a VQ based on GMM will be discussed further.

3.1 Density Estimation Using Mixtures

Density estimation for random sources has been done using two different approaches [4]: One is the maximum likelihood parameter estimation. This approach is called parametric data estimation. The *pdf* of the source $f(\mathbf{x})$ is assumed to belong to a family of parametric densities $f(\mathbf{x}|\boldsymbol{\theta})$. The goal here is to find the parameter $\boldsymbol{\theta}$ such that it best explains the source. This approach is computationally efficient, since the parameter can be estimated from a fairly small number of observations which linearly grows with the number of pa-

rameters [33]. However, the drawback of this method is the enforcement of an *a priori* structure on the observed data which may cause poor estimation of parameters.

Nonparametric density estimation on the other hand does not make any prior assumption about the unknown *pdf*. Therefore, the estimation is consistent which roughly means that if the number of samples is large enough, each estimated value is very close to true value irrespective of the unknown *pdf*. However, non parametric techniques need large number of observations. They also do not allow easy bit scalability and learning.

In order to solve this problem density estimation using mixture models [34], which lie in the class of quasi parametric estimator has been introduced. This method benefits from the advantages of both previously mentioned methods. The unknown *pdf* is modelled as a mixture of parametric *pdf*'s,

$$f(\mathbf{x}|\boldsymbol{\theta}) = \sum_{i=1}^{M} \alpha_i f_i(\mathbf{x}|\boldsymbol{\theta}_\mathbf{i}) \tag{3.1}$$

where

$$\Theta = [M, \alpha_1, ..., \alpha_M, \boldsymbol{\theta}_1, ..., \boldsymbol{\theta}_M]. \tag{3.2}$$

The parameter α_i are the weight constants which have to sum up to one. Parameterized by $\boldsymbol{\theta}_i$, $f_i(\mathbf{x}|\boldsymbol{\theta}_\mathbf{i})$ is an individual parametric density. Each of the parametric *pdf*'s $f_i(\mathbf{x}|\boldsymbol{\theta}_\mathbf{i})$ can also be called a *cluster*. Here M is the number of clusters is a design parameter. For a dimension d, the number of parameters will be in the order of $O(d^2)$. Equation (3.1) can be seen as a functional decomposition of the unknown *pdf* in terms of parametric *pdf*s which make the basis functions of this decomposition [35].

3.1.1 Missing Data Problem

Mixture models could be interpreted as missing data problems. The probability that an observation $\mathbf{x}$ belongs to cluster i is the mixture weight α_i. The missing data here is the information about to which cluster the observation belongs. Using the two steps of the Expectation Maximization algorithm (which will be discussed in the next section) we recursively substitute for the data using the expected value and the current estimate of the mixture parameters in the Expectation phase. We then update the parameters using the conditional mean in the maximization step. Due to the smooth nature of multivariate Gaussians, they serve well as good radial-basis functions (their characteristic feature is that their response decreases monotonically with distance from a central point). This means,

having an unknown function we can consider each of the Gaussians as basis function in space and they will model unknown functions properly. The problem of over–fitting can be solved using multivariate Gaussians. This problem occurs when the model is derived using a training data set and the model is over–fitted or to that training corpus, in other words it just represents a small set of training vectors and will not show the general statistics of source for a random test vector. Therefore, the quantizer may perform poorly for a test corpus. The smooth nature of Gaussians will help on modelling the form of the unknown statistics instead of the undesired details.

3.2 The EM algorithm

The expectation-maximization (EM) algorithm is a procedure for maximum-likelihood (ML) estimation in the cases where a closed form expression for the optimal parameters is hard to obtain. This iterative algorithm guarantees the monotonic increase in the likelihood L when the algorithm is run on the same training corpus. This approach for computing the ML estimate of *pdf* was first proposed by Dempester *et al* in [36]. We first look at the problem of the EM algorithm in general case and then the special case of Gaussian Mixtures will be considered.

3.2.1 Maximum Likelihood Parameter Estimation

Given a set of observation data in a matrix $\mathbf{X}$ and a set of observation parameters $\boldsymbol{\theta}$ the ML parameter estimation aims at maximizing the likelihood $L(\boldsymbol{\theta})$ or log likelihood of the observation data $\mathbf{X} = \{\mathbf{x_1}, ..., \mathbf{x_n}\}$

$$\hat{\boldsymbol{\theta}} = \arg\max_{\theta} L(\boldsymbol{\theta}). \tag{3.3}$$

Assuming that we have independent, identically distributed data we can write the above equations as

$$L(\boldsymbol{\theta}) = p(\mathbf{X}|\boldsymbol{\theta}) = p(\mathbf{x}_1, ..., \mathbf{x}_n|\boldsymbol{\theta}) = \prod_{i=1}^{n} p(\mathbf{x}_i|\boldsymbol{\theta}). \tag{3.4}$$

To find the maximum for this function, assuming an analytical function, we can take the derivative and set it equal to zero

$$\frac{\partial}{\partial \boldsymbol{\theta}} L(\boldsymbol{\theta}) = 0. \tag{3.5}$$

3.2.2 General Case

Before, further proceeding to the explanation of how this algorithm works we will give a few definitions.

- **Incomplete data** The observation sequence in the cases where it is not sufficient to describe the underlying model is called incomplete data.
- **Hidden Data** The data that can not be observed directly and gives information about the state of the underlying model.
- **Complete Data** Combination of hidden and observed data.

The two steps of the algorithms will run as followed

- Obtaining model parameters by maximizing the *Log Likelihood* of the incomplete data.
- Maximizing the *expected log likelihood* from the complete data.

The goal here is determining the optimum model parameters $\hat{\boldsymbol{\theta}}$ for our estimate. Assuming that the observed data is given in the matrix $\mathbf{X}$ we need to maximize the probability $p(\mathbf{X}|\hat{\boldsymbol{\theta}})$. $\mathbf{Y}$ is the hidden data. In the EM algorithm it is primarily assumed that we have an estimate $\boldsymbol{\theta}$ and want to estimate the probability that each y occurs.

In order to maximize the following log-likelihood function a function based on the expected value of the complete data is built

$$p(\mathbf{X}|\hat{\boldsymbol{\theta}}) \geq p(\mathbf{X}|\boldsymbol{\theta}) \qquad \text{or} \qquad \log \frac{p(\mathbf{X}|\hat{\boldsymbol{\theta}})}{p(\mathbf{X}|\boldsymbol{\theta})} \geq 0. \tag{3.6}$$

Maximizing the the following auxiliary function will lead to find the maximum likelihood answer,

$$Q(\boldsymbol{\theta}, \hat{\boldsymbol{\theta}}) = \sum_{Y} p(\mathbf{X}, \mathbf{Y}|\boldsymbol{\theta}) \log p(\mathbf{X}, \mathbf{Y}|\hat{\boldsymbol{\theta}}) = E_{\boldsymbol{\theta}}\{\log p(\mathbf{X}, \mathbf{Y}|\hat{\boldsymbol{\theta}})\}. \tag{3.7}$$

The four steps of the EM algorithm can be summarized as followed:

1. **Initialization**: Choosing and initial value for the model parameters $\boldsymbol{\theta}$
2. **Expectation step**: Computing the function Q based on $\boldsymbol{\theta}$
3. **Maximization step**: Computing $\hat{\boldsymbol{\theta}} = \arg\max_{\hat{\boldsymbol{\theta}}} Q(\boldsymbol{\theta}, \hat{\boldsymbol{\theta}})$ to maximize the function Q.
4. **Iteration**: Set $\boldsymbol{\theta} = \hat{\boldsymbol{\theta}}$ and repeating the steps 2 and 3 till the convergence of the algorithm.

This algorithm can be applied to many problems. Hidden Markov Models (HMMs) and GMMs are two specific cases. The explanation in the case of GMMs follows. The *pdf* of each observation vector at a specific time t and the parameter $\boldsymbol{\theta}$ are expressed as

$$P_{\boldsymbol{\theta}}(\mathbf{x}_t) = \sum_{m=1}^{M} p_m b_m(\mathbf{x}_t) \tag{3.8}$$

$$\boldsymbol{\theta} = \{p_m, \boldsymbol{\mu}_m, \mathbf{C}_m\}, m = 1, ..., M. \tag{3.9}$$

The parameter p_m is the mth mixture weight, $\boldsymbol{\mu}_m$ is the mth mixture mean and $\mathbf{C}_m$ is the mth mixture covariance. The observation matrix can be expressed in time as $\mathbf{X} = \{\mathbf{x}_1, ..., \mathbf{x}_T\}$ and the component class indices , $I = \{i_1, ..., i_t\}$ where $i_t \in [1, M]$.

The likelihood of $\mathbf{X}$ can be written as

$$p(\mathbf{X}|\boldsymbol{\theta}) = \sum_{I} p(\mathbf{X}, I|\boldsymbol{\theta}) \tag{3.10}$$

where

$$p(\mathbf{X}, I|\boldsymbol{\theta}) = \prod_{t=1}^{T} p_{i_t} b_{i_t}(\mathbf{x}_t). \tag{3.11}$$

Regrouping the terms we can finally write the function Q in the following closed form

$$Q(\boldsymbol{\theta}, \hat{\boldsymbol{\theta}}) = \sum_{t=1}^{T} \sum_{i=1}^{M} \log[\hat{p}_{i_t} \hat{b}_{i_t}(\mathbf{x}_t)] \gamma_t(i) \tag{3.12}$$

where

$$\gamma_t(i) = p(\mathbf{X}|\boldsymbol{\theta})p(i_t = i|\mathbf{x}_t, \boldsymbol{\theta}) \qquad \text{and} \qquad p(i_t = i|\mathbf{x}_t, \boldsymbol{\theta}) = \frac{p_i b_i(\mathbf{x}_t)}{\sum_{k=1}^{M} p_k b_k(\mathbf{x}_t)}. \tag{3.13}$$

Therefore the new model parameters $\hat{\boldsymbol{\theta}} = \{\hat{p}_m, \hat{\boldsymbol{\mu}}_m, \hat{\mathbf{C}}_m\}$ can be found by solving the following expression

$$\frac{\partial Q(\boldsymbol{\theta}, \hat{\boldsymbol{\theta}})}{\partial \hat{\boldsymbol{\theta}}} = 0. \tag{3.14}$$

3.2.3 Special Case: Gaussian Mixtures

In order to find the cluster weights, mean and covariance matrices we need to solve the above equation for this specific case. We will not explain the details here and just state the final results for the EM algorithms. The interested reader is referred to [37] for the detailed proof.

1. **Initialization**: In this step an initial estimate of the parameters is obtained. For all values of m the weight coefficients are initialized to $1/M$ the mean vectors are initialized either to a random vector or a vector randomly chosen from the training data base $\mathbf{X}$ and the covariance matrices $\mathbf{C}_i^{(1)}$ are initialized to a matrix with all of the elements equal to one.

2. **Likelihood Computation**: In each iteration k compute the posterior probabilities $P(\omega_m|\mathbf{x}_t, \boldsymbol{\theta})$ using

$$P^{(k)}(\omega_m|\mathbf{x}_t, \boldsymbol{\theta}) = \frac{p_i^{(k)} f_{\mathbf{X}|\boldsymbol{\mu}_i^{(k)}, \mathbf{C}_i^{(k)}}(\mathbf{x_t}|\boldsymbol{\mu}_\mathbf{i}^{(k)}, \mathbf{C}_\mathbf{i}^{(k)})}{\sum_{j=1}^{M} p_j^{(k)} f_{\mathbf{X}|\boldsymbol{\mu}_j^{(k)}, \mathbf{C}_j^{(k)}}(\mathbf{x_t}|\boldsymbol{\mu}_\mathbf{j}^{(k)}, \mathbf{C}_\mathbf{j}^{(k)})} \tag{3.15}$$

   where $P(\omega_m|\mathbf{x}_t, \boldsymbol{\theta})$ shows the posterior probability that given the observation vector $\mathbf{x}_t$ and the parameter $\boldsymbol{\theta}$ the data is generated by the mth cluster.

3. **Parameter Update**: Having the posterior probabilities compute the weights, mix-

ture means and covariances by

$$p_i^{(k+1)} = \frac{1}{M} \sum_{t=1}^{T} P^{(k)}(\omega_m | \mathbf{x}_t, \boldsymbol{\theta}) \tag{3.16}$$

$$\boldsymbol{\mu}_i^{(k+1)} = \frac{\sum_{t=1}^{T} P^{(k)}(\omega_m | \mathbf{x}_t, \boldsymbol{\theta}) \mathbf{x}_n}{\sum_{t=1}^{T} P^{(k)}(\omega_m | \mathbf{x}_t, \boldsymbol{\theta})} \tag{3.17}$$

$$\mathbf{C}_i^{(k+1)} = \frac{\sum_{t=1}^{T} P^{(k)}(\omega_m | \mathbf{x}_t, \boldsymbol{\theta})(\mathbf{x}_n - \boldsymbol{\mu}_i^{(k+1)})(\mathbf{x}_n - \boldsymbol{\mu}_i^{(k+1)})^T}{\sum_{t=1}^{T} P^{(k)}(\omega_m | \mathbf{x}_t, \boldsymbol{\theta})}. \tag{3.18}$$

A problem that usually arises in applications of mixture modelling is the implementation of the algorithm when the number of parameters in $\boldsymbol{\theta}$ increases. If we use more parameters, the obtained freedom may cause us the problem of over fitting, since the random properties of the training data may have been reflected in the model. On the other hand, a large number of parameters will cause undue complexity [31]. Therefore, there is need to find a compromise. Below one popular method for this goal is explained.

Use of Diagonal Covariance Matrices in GMMs

The number of parameters could be decreased by a great amount if diagonal covariance matrices are used instead,i.e. $\mathbf{C}_i = diag\{\lambda_{i,1}, \cdots, \lambda_{i,d}\}$ where d is the dimension of the input vector. It has been shown in [3] that in order to have the same precision in modelling the underlying *pdf* of $\mathbf{x}_t$ the use of diagonal covariance matrices will result in bigger number of Gaussians comparing to the original case. The update equation for the elements $\lambda_{i,j}^{(k+1)}$

then changes from Eq. (3.16) to

$$\lambda_{i,j}^{(k+1)} = \frac{\sum_{t=1}^{T} P^{(k)}(\omega_m|\mathbf{x}_t, \boldsymbol{\theta})(x_{t,j} - \mu_{i,j}^{(k+1)})^2}{\sum_{t=1}^{T} P^{(k)}(\omega_m|\mathbf{x}_t, \boldsymbol{\theta})} \tag{3.19}$$

where $x_{t,j}$ and $\mu_{i,j}^{(k+1)}$ are the jth component of the observation vector at time t and the mean vector at the $(k+1)$th iteration.

The performance of the EM algorithm depends on its initialization. Although, the overall shape of the density only varies slightly with different initializations, but the individual parameters may depend greatly on the initial value used for the algorithm, there have been different initialization methods proposed in the literature [38].

3.3 The Curse of Dimensionality

The "Curse of Dimensionality" is a phrase first used in [39]. This refers to the problems including high dimensional data. The problem is that in high number of dimensions the data will be very sparse. As an example looking at 10 points in unit interval and in a unit 10-dimensional hypercube, we can see that the points are much closer on the unit interval. Another related problem is the over parameterization of data processing algorithms. For instance, in the problem of gaussian mixtures a mixture of M Gaussians has $O(M \times n^2)$ [40] where n is the input dimension. The problem that may arise here is that if the number of parameters is too large compared to the number of data points, the model may *over–fit* to the training data. In order to constrain the number of parameters *regularization* algorithms are used [40]. One way of regularizing the covariance matrices is to assume they are all diagonal or spherical.

A problem in GMMs is that when one of the covariance matrices is singular the likelihood function has a maximum. In order to avoid these singularities we may impose artificial lower bounds on the volume elements (determinants of the covariance matrices) for each Gaussian. This is done by adding a small diagonal matric $\epsilon \mathbf{I}_{n \times n}$ to each covariance matrix in each iteration of the EM algorithm [12]. In the work done by Ormoneit *et al* [12] they note that ϵ introduces a bias in favor of large variances. The value of ϵ is changed over

a range of values and the one which minimized the classification error in their problem is chosen. This algorithm has been used in our implementation and is needed in the case of full covariance matrices with high number of dimensions. The bias term is added to the covariance matrix in each iteration of Maximization step and also is added to the posterior probabilities. Finally, after changing the values of ϵ the one which causes the smallest error in the model and still helps avoiding the problem of singularity is chosen.

3.4 Some of The Applications of Gaussian Mixture Models

Mixture models have been used for identification of outliers and tackling the problem of missing data. They have also been used as clustering techniques, supervised learning in the "mixture of experts" architecture [41], for classification using bayes classifiers and for regression. Below some of the applications of Gaussian Mixtures, prior to their use in speech quantization is discussed.

3.4.1 Classification

A classifier assigns vectors from an n dimensional feature space to one of K classes or categories and partitions the feature space to K disjoint regions. In [40] a new classifier called Gaussian Mixture Bayes classifier is introduced. This classifier also suffers from the problem of curse of dimensionality, since for K classes and Q mixtures the order of operations will be $O(K \times Q \times n^2)$. Kambhatla has explored ways of regularizing this GMB in [40]

3.4.2 Regression

The goal in regression is the prediction of m response variables y, given observed values of n predictor variables x. The joint density of the input and output can be modelled using Q Gaussians. The regression function can be written as $E[y|x]$ and the regression will be a weighted sum of linear models where the weights are component probabilities in the GMM. Thus a GM model for regression defines a *soft local linear* model for data [40]. In this work regularized GM regression algorithms have been proposed.

3.4.3 Gaussian Mixtures in Speech Recognition

Gaussian mixture models have been used in speech recognition in [3]. The goals in speech recognition can be categorized into two groups: verification and identification. In verification the goal is to determine whether a claimed identity is true or not. In speaker identification the goal is to find out which one of the voices in a given set, best matches the input voice. Reynolds and Rose [3] have used GMMs for robust text independent speaker identification. It has been shown that speech spectrum is effective for speaker identification [42]. This is due to the fact that, the spectrum reflects the vocal tract's structure. In [3] cepstral coefficients which are derived from a mel frequency filter bank to represent short time speech spectra (Fig. 3.1) have been used. In their model they have used mixtures of

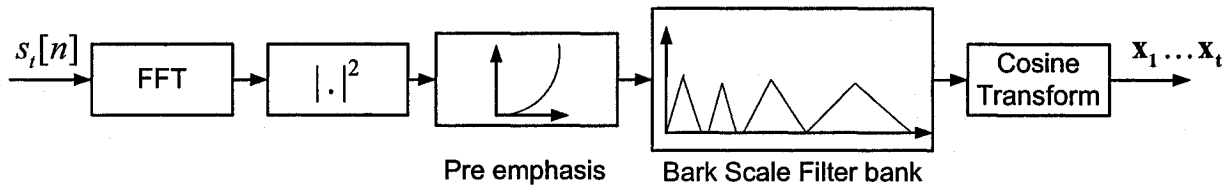

Fig. 3.1 Bark-Scale cepstral feature analysis [3]

Gaussians with M components. Each speaker is represented by a GMM and is referred to by his/her own parameters θ where,

$$\theta = \{p_i, \boldsymbol{\mu}_i, \mathbf{C}_i\} \qquad i = 1, \ldots, M \tag{3.20}$$

There have been two main reasons for choosing GMMs as the representations of speaker identity. First their ability to model different acoustic classes. The acoustic classes like vowels nasals or fricatives reflect some general speaker dependent vocal tract identifications. The spectral shape of each of the classes is represented by the corresponding cluster's mean and its average spectral shape is shown using the covariance matrix [3]. The second reason is that GMMs are good basis functions, since each of the components has a smooth nature. Therefore, they can model arbitrary shaped smooth densities very well. In this work, for a set of S speakers S GMMs have been used. The approach to detect the identity of a speaker is as followed. For the input vector set $\mathbf{X}$ the conditional probability $p(\mathbf{X}|\theta_\mathbf{k})$ is computed for $k = \{1, \ldots, S\}$. The parameter set which maximizes this probability will be chosen. The index of this parameter set determines which speaker has been chosen. In

other words this can be written as

$$\hat{S} = \arg\max_{1 \leq k \leq S} \sum_{t=1}^{T} \log p(\mathbf{x}_t | \theta_k) \tag{3.21}$$

$\mathbf{x}_t$ are the feature vectors. This is a maximum likelihood approach for user identification.

3.5 GMM Based Vector Quantization

Gaussian mixture models have become more popular recently in the field of image and source coding. Su and Mercereau have used GMMs and generalized gaussian mixtures [43] in image coding. The case that they have considered, concerns the problem of DC distribution. They have used block transform coding in which the image is divided to $N \times N$ blocks, and a transform of each block is performed. Prior to their work, DPCM has been used to exploit the correlation between DC samples (here DC refers to zero-frequency transform coefficients). However, the bit allocation for this scheme is complicated. They have used Gaussian mixtures to model the DC data. In [32] the information theoretic extremal property for source coding has been extended from Gaussians to mixtures of Gaussians using high rate quantization theory. Gray has also extended a method originally used for LPC speech VQ to provide a Lloyd clustering approach to the design of Gauss mixture models. GMMs have been used by Hedelin and Skoglund [31] for parameterz training in high rate VQ. They obtain expressions for high rate VQ in terms of lower and upper bounds. These expressions accurately give numerical high rate predictions even for small data sets. In practical usages the support of a GMM used for modelling the data might be bounded. This is called the problem of bounded support. Therefore, the *pdf* is defined only for a specific set of input values of the vector $\mathbf{x}$ to be quantized. Hedelin *et al* [31] present a *modified* EM algorithm for densities with bounded support which models the data with less number of parameters.

The recent work of Subramanian and Rao in GMM Based VQ, concentrates more on building structures for efficient VQ in a practical way rather than the theoretical high rate expressions derived by Hedelin *et al* [4].

The work in this thesis uses the same building blocks that have been used in [4]. Fig. (3.2) depicts the overall scheme. The scheme shown uses a set of training vectors $\mathbf{X}_n$ to find the parameters of the GM Model. Then quantization is done in a closed form

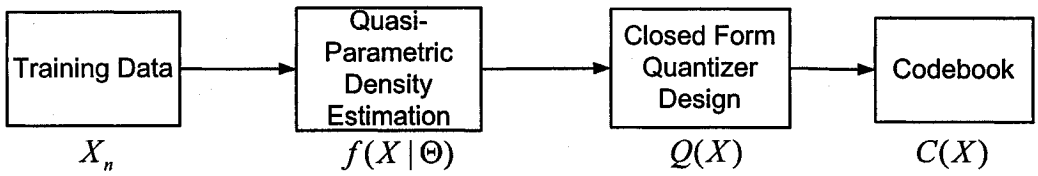

Fig. 3.2 Mel-Scale cepstral feature analysis [4]

formulation which will be discussed more thoroughly in the next chapter. The quantizer design is independent of the data. Therefore, the statistics of the data will affect the density estimation part only. These parameters could be updated adaptively while the data is changing. Therefore, the parametric density estimation given here followed by the closed form quantizer design will give the advantages of bit rate scalability, variable rate coding, interoperability (which means the ability of switching between memoryless quantizers and quantizers with memory) and learning. If we can build models which properly show the underlying statistics of the source, optimal finite rate quantizers can be built. In practice this optimality means having quantizers which provide us good distortion performances at low complexity. In the scheme proposed, the main idea is quantizing each of the clusters in the GM Model *independently.* Therefore, bit allocation should be done first between the clusters and then to each dimension in the cluster. The number of bits allocated to each cluster will be affected by the following facts.

- whether the system is fixed or variable rate
- the weight or probability given to each cluster in GM model
- the covariance matrix of the cluster

Transform coding is done within the clusters to allocate bits to each of the d components of cluster. Here we assume that the dimension of the input vector is d. The quantizer scheme used in each dimension uses a companding scheme (a compressor followed by a uniform scalar quantizer and then by an expander). After the quantization step, the quantized version which minimizes the relevant distortion measure will be chosen as the output. Although, this quantizer is not optimum for a given mixture model the gain made in search and memory complexity justifies the loss of optimality [4].

In the next chapter the concepts from Chapter 2 and 3 will be used to build a "perceptual domain" quantizer. The detailed transformations, quantizer design aspects, the bit

assignment scheme and results obtained will be discussed more thoroughly.

Chapter 4

Design of a Perceptual GMM Based VQ

As discussed in the previous chapters our design goal is building a quantizer which works in the perceptual domain and its corresponding distortion measure is perceptually weighted. The use of GM based VQ gives us the freedom of variable rate coding and bit rate scalability. The input speech or audio file is divided to frames of length 60, 80 or 120. These input vectors are then transformed to power spectral domain. This transformation is done using both the DFT and MDCT and the results are compared. Here, we assume the perfect reconstruction of the phase and the quantization is done on the magnitude.

4.1 The Overall Scheme of the Proposed Method

The overall proposed encoder scheme is depicted in Fig. 4.1. Frequency representation vectors of length 60, 80 and 120 have been used here for the purpose of quantization. Since we are using 50% overlap between successive input frames the corresponding frame lengths will be 120, 160 and 240. The sampling frequency for audio and speech files used is 8 kHz. Therefore the time spacing between two adjacent samples will be 0.125 msec. The reason that we have chosen 50% overlap is to avoid discontinuities and blocking effects. Each frame here has been windowed using the "sine" window discussed before in Chapter 2. The graph of a sine window for windowing a vector of length 240 is depicted in Fig. 4.2

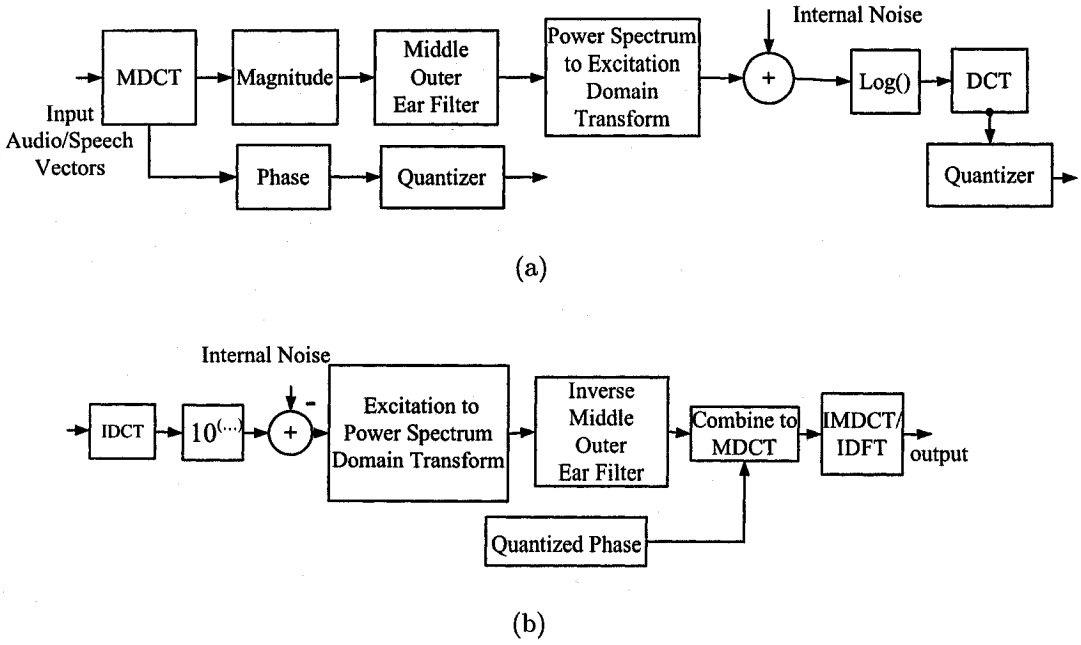

Fig. 4.1 The proposed quantization scheme a) encoder part b) decoder part

4.1.1 The effect of windowing with overlap

For input blocks of length N after windowing each of the blocks, a pointer will be shifted $N/2$ samples in our data file to specify the beginning of the next block. Assuming that we have L samples, the number of frames in the case where we do not have overlap is $[\frac{L}{N}]$, while the number of frames of length N for the same number of samples, in the case of 50% overlap is equal to $[\frac{L}{2N}]$ (after the overlap and add operation each block of length N is called a frame). This can be interpreted as having frames with twice the time duration of the previous case.

We can benefit from the conjugate symmetric property, which exists for both MDCT and DFT, in our power spectrum computations [44] for real signals. This can be expressed as

$$X(k) = X^*(N-K), \qquad 1 \leq k \leq N-1 \tag{4.1}$$

where $X(k)$ represents each DFT or MDCT coefficient. The DC component of speech/audio is not perceived by the human ear. Therefore, for a vector of length 240, removing the DC element and using the symmetry property, we only need to send 120 DFT/MDCT elements.

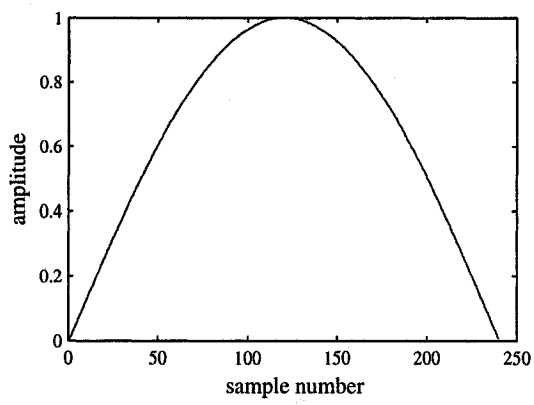

Fig. 4.2 Mal window for windowing a vector of length 240

4.1.2 Use of MDCT/DFT

We have used DFT and MDCT separately, as two frequency representations for our experimental results. As shown in Eq. (2.6) MDCT is a sequence of real numbers. Therefore, we need to quantize the amplitude and the sign. In our implementations we have assumed perfect reconstruction of the sign and only magnitude has been quantized. On the other hand, a DFT transform is represented by a magnitude and a phase. We still assume phase to be reconstructed and only quantize the amplitude.

Generally building a quantizer for sign of a real number can be considered as a comparator with threshold equal to zero. We need 1 bit/coefficient to build this quantizer. This will add up to the magnitude bit rate in a real implementation. In [11] an MDCT based coding of speech is introduced and sign quantization has been considered in more detail. They assume the phase to be a sequence of signs with equal symbol probability. to reduce the coding rate the psychoacoustics and statistics of this sequence are exploited. The sequence is divided to a masked and an unmasked set according to masking thresholds. The components that are considered masked which are known to the decoder are not coded explicitly. Using this scheme, on average 70% of the components belong to unmasked set and sign is coded with and average around 0.5 bits/dim. The coding scheme used is a Huffman code. However, DFT phase can be a continuous number in the interval $[0, \pi]$. Therefore, in general building a phase quantizer is more computationally complex and the number of bits needed is more than 1 bit/coefficient.

Finally the results for quantized files using both transformations are given separately

in the experimental section.

4.1.3 Middle/Outer Ear Transformation

The spectrum of each frame will be transformed to a frame of the same length using the transformations discussed in Chapter 2. Equation (2.21) is used here to implement the middle/outer ear transformation. In our implementations we have assumed f_0 to be in Hertz frequency instead of Bark frequency. For a real implementation we have used the matrix approximation to Eq. (2.13) as we described previously in Section 2.3. For an input vector of length N (by input vector we mean the vector in the frequency domain) $\mathbf{f_0}$ is a vector of length N, which is uniformly sampled in the frequency interval $[0, 4000\text{Hz}]$ and each element of this vector represents the frequency center for one of the smearing functions. The input vector is multiplied by the outer to middle ear transform $\mathbf{D}$ which is a diagonal matrix obtained from the samples of the continuous Eq. (2.21).

4.1.4 Moore's Excitation to Power Spectrum Transform

The Excitation domain transform has been implemented using the matrix given in Eq. (2.16). The vector $\mathbf{z_0}$ has been assumed to be in the Hertz frequency domain like the previous section. We have implemented the transformations in the Bark frequency domain as well. The details and the logic behind this choice are discussed below.

Equation (2.13) is a representation in Hertz frequency. If, instead, we prefer to have the transformation to Bark frequency we have to replace the smearing function $W(f, f_0)$ by $W'(f, z_0)$, where z_0 is the corresponding frequency element in Bark domain. The power spectrum is given by DFT vectors, uniformly distributed in Hertz domain. Therefore, to avoid the problem of finding the DFT coefficients at the Bark frequencies, it is convenient not to change the variable of integration in Eq. (2.13) from Hertz to Bark. Finally the transformation in Bark domain in the can be expressed as

$$E(z_k) = \int_0^\infty P(f)W(f, g^{-1}(z_k))df \tag{4.2}$$

which can be approximated by the following matrix representation

$$\mathbf{e} = \frac{F_s}{2N}\mathbf{W}\mathbf{p} \tag{4.3}$$

where $\mathbf{W}_{(i,k)} = W(\frac{iF_s}{2N}, g^{-1}(z_k))$. The transformation used here to switch from frequency to Bark is, $z = g(f) = 7\sinh^{-1}(f/650)$. We transformed the Hertz frequency range $[0, F_s/2]$ to the Bark frequency range $[0, 7\sinh^{-1}(\frac{F_s}{2\times 650})]$. The Bark frequency interval has been uniformly sampled, with N samples. Uniformly spaced bark frequency elements can be represented as

$$z_k = \frac{k}{N} g(\frac{F_s}{2}) = \frac{k}{N} 7\sinh^{-1}(\frac{F_s}{2\times 650}) \quad k = 1, \ldots, N \tag{4.4}$$

where we have replaced the frequency f by an index k. After the transformation back to frequency domain using $f_k = g^{-1}(z_k)$, we have nonuniform samples in Hertz frequency domain, which are used to find the elements of the matrix $\mathbf{W}$.

The implementation shows that for a 120×120 matrix, for this case where we assume uniform sampling in Bark domain using Eq. (4.4) the transformation matrix $\mathbf{W}$ is ill conditioned and the condition number is in the order of 10^{19}. Comparing to the condition number 10^5 in the case where we sampled the frequency interval $[0, 4000\text{Hz}$ uniformly, this justifies our rationale for using the Hertz frequency. Using the former transformation, will cause instabilities in the results of implementations.

A graph is included to show the difference between the excitation patterns between the Hertz and Bark frequency representations (Fig. 4.3). A sinusoid of frequency 2000 Hz with DFT amplitude of 40 has been chosen. This vector has been transformed using the matrix $\mathbf{W}$ which is implemented once in Hertz frequency and once in Bark frequency. The frequency vector here spans the interval $[0, 4000\text{Hz}]$, therefore since we are using 120 vectors the frequency separation is 33.3 Hz. In the case of the Hertz frequency the maximum has occurred at index 60 which is equivalent to 2000 Hz. The frequency interval mentioned is mapped to $[0, \ldots, 17.5]$ under the bark transformation. The maximum in the case of Bark frequency occurs at the index 90, which is equivalent to $90 \times 17.5/120 = 13.12$ Bark. This in term, is equivalent to 2000 Hz.

4.1.5 Calibration of Loudness

Calibration from input levels to loudness can be done in order to assume the perceptual properties of the transformed vectors. The full scale sine has been considered to have a Sound Pressure Level of 92 dB (called L_p). Integration of power density over frequency gives the signal energy, which is shown by Parseval's relation as well. Finally, this energy is divided by the full scale sound pressure level. Finally the scaling factor which depends

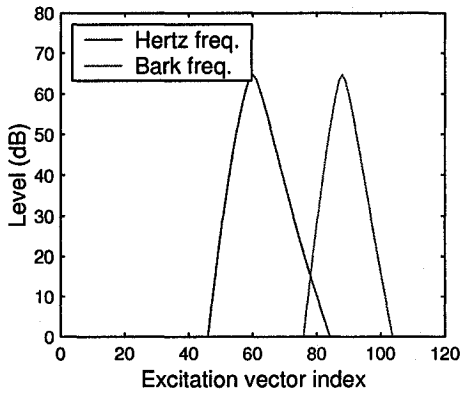

Fig. 4.3 Comparison of excitation vectors for a sinusoid of frequency 2000 Hz using Hertz frequency and Bark frequency transformation matrices

on the vector length, sampling frequency and full scale SPL is obtained. As an example input vectors of length 120 with a sampling frequency of 8000 Hz and L_p of 92 dB will give a normalization factor of 5.6×10^{-5}. Assuming level independence for the perceptual transform as we had in Section (2.3.1) we no longer need to consider this effect in for normalization of the signal. This factor is also needed in the computations of threshold of hearing.

4.2 Use of DCT for Decorrelating the Data

4.2.1 The Karhunen–Loeve Transformation versus DCT

The principal component analysis or Karhunen-Loeve transform is a mathematical way of determining a linear transformation of a sample of points in N dimensional space which exhibits the properties of the sample along the coordinate axes. Along the new axes the sample variances are extremes (maxima and minima), maximally aligned and uncorrelated. The name comes from the principal axes of an ellipsoid (e.g. the ellipsoid of inertia), which are the coordinate axes in question. It is designed to capture the variance in a data set in terms of principle components. In effect, one is trying to reduce the dimensionality of the data to summarize the most important (i.e. defining) parts while simultaneously filtering out noise. Below we review a mathematical representation of this transform.

This transform is an expansion of a random vector in the eigenvectors of the covariance matrix [45]. PCA linearly transforms an N dimensional vector $\mathbf{x}$ to N' dimensional vector

$\mathbf{y}$ using the orthonormal vector $\mathbf{W}$,

$$\mathbf{y} = \mathbf{W^t x} \qquad \mathbf{W} = [\mathbf{w_1}, \mathbf{w_2}, \cdots, \mathbf{w_{N'}}] \qquad \mathbf{W^t W} = \mathbf{I} \tag{4.5}$$

where $\mathbf{w_i}$ represents each of the eigenvectors of the covariance matrix $\mathbf{C}$ and the transformation $\mathbf{W}$ corresponds to the $N^{'}$ eigenvectors of the covariance matrix $\mathbf{C}$, corresponding to the first $N^{'}$ largest eigenvalues where

$$\mathbf{Q} = \{\mathbf{e}_1, \mathbf{e}_2, \cdots, \mathbf{e}_{N'}\} \quad \text{where} \quad \mathbf{C} = \mathbf{Q}\Lambda\mathbf{Q}^{'}. \tag{4.6}$$

This transform is optimum in the sense that it minimizes the variance of the axes in the orthonormal space, meaning that the quantity $D = \sum_{i=N'+1}^{N} \lambda_i$ is minimized. Here $\{\lambda_1, \cdots, \lambda_N\}$ are the eigenvalues of Λ . Also, the resulting components of each dimension are uncorrelated.

In our coder design, we need to model the underlying statistics of the data. Therefore, GMMs have been used to model the training data. However, using diagonal covariance matrices will reduce the computation complexity. The problem here is that if we have big off diagonal covariance components, diagonal covariance matrices will not be a good approximation. Decorrelating the data vectors, we will have the to use the diagonal covariance matrix model, which is simpler.

Another difficulty that arises here is the implementation of the Karhunen–Loeve transform. Since, this is a data driven transform it has to be updated adaptively based on the new data vectors. Huang and Zhao show that [46] for a frame of length N the cost of computing the eigenvalues of the signal covariance matrix is $O(N^3)$, which is large. The other trouble is the need to send side information to implement Karhunen–Loeve transform. However, we have the choice of using the Discrete Cosine transform (DCT) which is an approximation to KLT.

The discrete cosine transform (DCT) is used in image processing and helps separate the image into parts (or spectral sub-bands) of differing importance (with respect to the image's visual quality). The DCT is similar to the discrete Fourier transform: it transforms a signal or image from the spatial domain to the frequency domain. An AR(p) model has

a rational system function with p poles and for an input $w(n)$, the output $x(n)$ is

$$x(n) = -\sum_{k=1}^{p} a_k x(n-k) + d_0 w(n). \tag{4.7}$$

In [46], it is proved that for an AR(p) random process KLT can be approximated by DCT. In this case the order of computations reduces to $O(N^2)$ (If FFT is used in the implementations the computation order will decrease to $N \log_2 N$). We have used DCT in our implementations for decorrelating the data vectors. The one dimensional transform for a frame of length N is defined as

$$S(k) = c(k) \sum_{n=0}^{N-1} s(n) \cos \frac{\pi(2n+1)k}{2N}$$

$$\text{where} \qquad c(0) = \sqrt{\frac{1}{N}}, \qquad c(k) = \sqrt{\frac{2}{N}} \qquad k = 1, \ldots, N. \tag{4.8}$$

The inverse DCT for the one dimensional case is defined as,

$$s(n) = \sum_{k=0}^{N-1} c(k) S(k) \cos \frac{\pi(2n+1)k}{2N}. \tag{4.9}$$

4.2.2 Experimental Results Using DCT

In order to compare the statistics of the source vectors before and after the DCT transformation the normalized covariance coefficients are computed as followed:

$$\rho_{ij} = \frac{E[(\mathbf{x}_i - \mu_i)(\mathbf{x}_j - \mu_j)]}{\sqrt{E[(\mathbf{x}_i - \mu_i)]^2 E[(\mathbf{x}_j - \mu_j)]^2}}. \tag{4.10}$$

In this case the vector $\mathbf{x}_i$ represents the MDCT transformed speech and audio vectors, either before or after the DCT transform. These values have been computed for both the original excitation vectors and the DCT transformed version. The total of 200 000 vectors have been used for simulations and averaging has been done as an approximation to the expectation operation. As it can be seen in Fig. 4.4 the normalized covariance coefficients decrease more rapidly in the case of the DCT version which shows a less correlation between different vectors. In order to better visually understand the effect of DCT, we have plotted

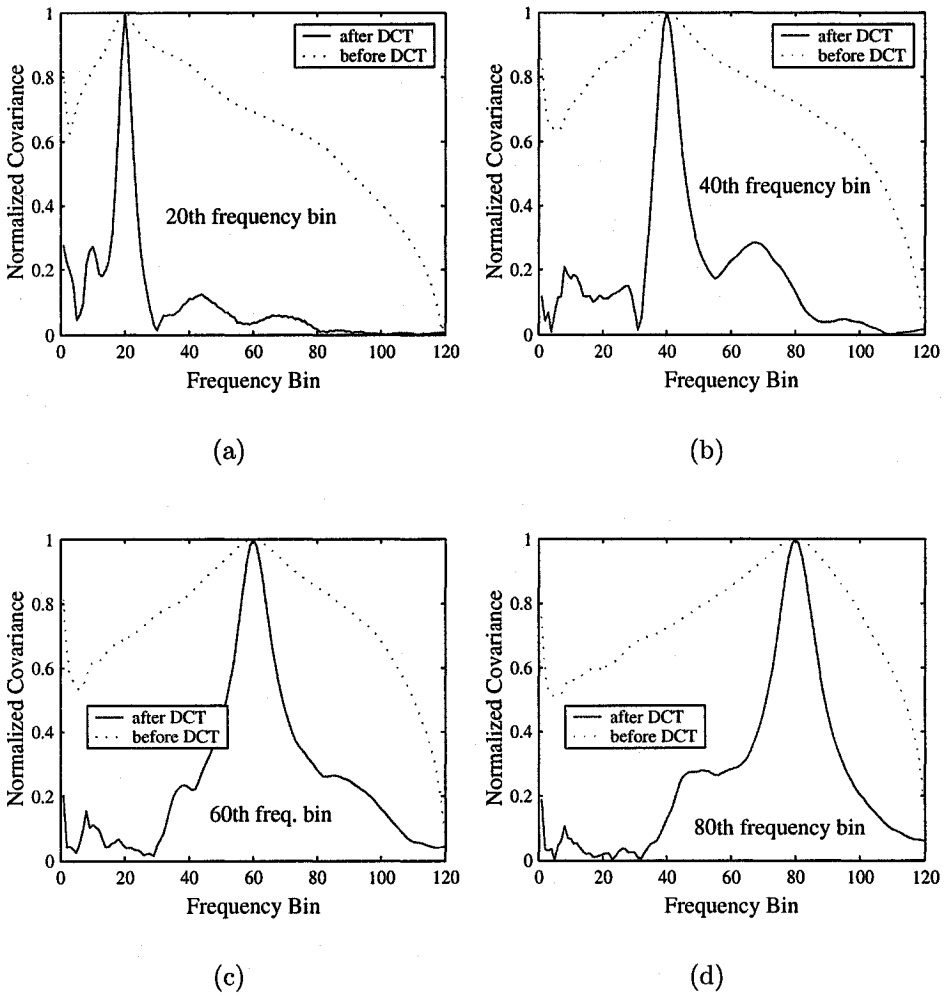

Fig. 4.4 Normalized Covariance Values for the a) 20th b) 40th c) 60th d) 80th frequency bin, before and after the DCT transform

the 2D histogram of data before and after the DCT transform. These histograms, show the two dimensional distribution of data for different frequency bins. As it is seen in Fig. 4.5 after the transform the data points are more symmetrically distributed around the origin. No mean removal has been done at this step. The colors in the histogram show the magnitude. The values are in the order of increasing from white to black. The energy compaction property of DFT can be clearly seen here. After the DCT transform the lower frequency bins occupy a larger space, while the higher frequency bins occupy a smaller space. In other words the energy has been more concentrated around the low frequency elements.

4.3 The Nonnegative Least Squares Problem

An important issue in the implementation of the Excitation domain transform is the invertibility of the transformation. This concept has been first introduced by Johnston in [13], where he mentions the necessity of inverting the spread threshold back to Bark domain after convolution of power spectrum with spreading function. In this case, the deconvolution is very unstable and may result in negative power spectrum. Therefore renormalization has been used as a solution. We have used a different approach which is explained below.

The space of an N dimensional power spectrum vector spans R^{+^N}, which is the N dimensional vectors with all nonnegative elements. However, transforming these vectors to the excitation domain using the transform matrix $\mathbf{T}$ which is equal to $\mathbf{DW}$, the new vectors no longer span R^{+^N}. The matrix $\mathbf{D}$ is introduced in section (2.3.3). The new space is a subset of the original space which is bounded by a polyhedron. The faces of this polyhydroxy are spanned by each of N combinations of $N-1$ columns of the transform matrix.

A problem that may arise here is that, we have limited the acceptable points to a subspace of R^{+^N} named S. The points which are near the boundaries of S are most likely to be quantized outside S. However, transforming the quantized excitation elements back to the power spectrum domain, we may wind up in power spectrum with negative entries. One way of avoiding this problem is finding the admissible excitation pattern $\mathbf{e}'$ which is closest to the quantized excitation pattern $\hat{\mathbf{e}}$. The correction algorithm involves successively projecting the quantized vector on the faces of the polyhedron and choosing the vector which minimizes the error. This works when the quantized excitation vectors all

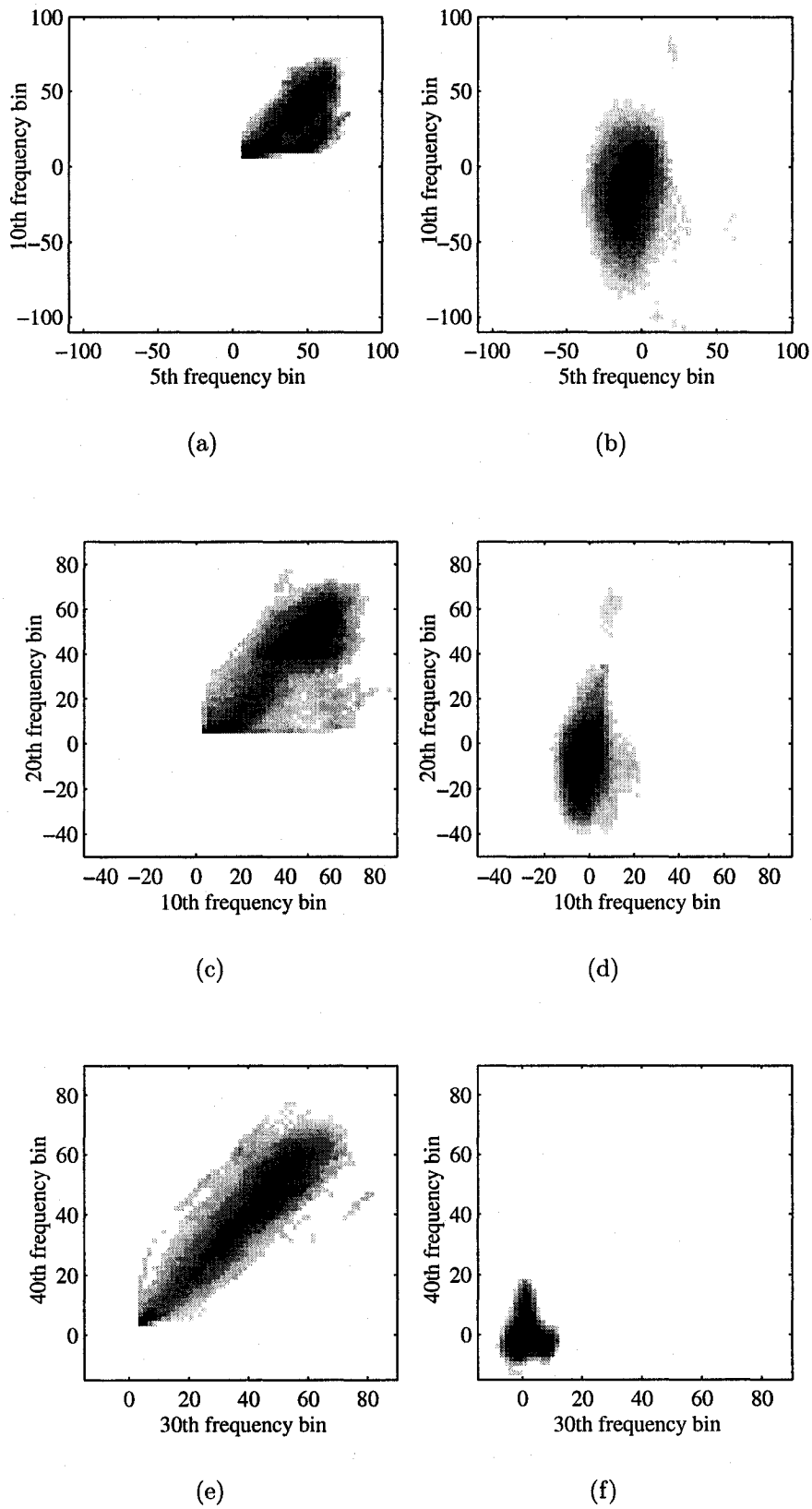

Fig. 4.5 2D histogram of speech vectors before DCT transformation: a)

have positive elements. This way it is ensured that the projection involves positive linear combination of basis vectors.

The general problem of *least squares* can be formulated as follows

$$\min ||\mathbf{Tp} - \hat{\mathbf{e}}||, \text{ subject to } \mathbf{Lp} \geq \mathbf{h} \tag{4.11}$$

where $\mathbf{h}$ is a desired vector and the comparison between the vectors is done element by element. We are interested in the case where the resulting power spectrum is positive. In other words $\mathbf{L}$ is the identity matrix and $\mathbf{h} = \mathbf{0}$, $\mathbf{T}$ is the excitation domain transformation and $\hat{\mathbf{e}}$ is the quantized excitation vector. If we do not use the least square algorithm to convert back to the power spectrum domain and simply use the inverse transform of the matrix $\mathbf{T}$ we may wind up in negative power spectral elements. This case has been studied in literature [47]. Characterizing conditions of Eq. (4.11) are subject to the following theorem [47].

Theorem 1 (Kuhn-Tucker conditions for LSI problem) *An n dimensional vector $\hat{\mathbf{p}}$ is a solution for LSI problem if and only if there exists an m dimensional vector $\hat{\mathbf{y}}$ and a partitioning of the integers 1 to m into subsets Ψ and Ω such that*

$$\mathbf{L}^T\hat{\mathbf{y}} = \mathbf{T}^\mathbf{T}(\mathbf{T}\hat{\mathbf{p}} - \hat{\mathbf{e}}) \tag{4.12}$$

$$\hat{\mathbf{r}}_i = 0 \quad for \quad i \in \Psi, \qquad \hat{\mathbf{r}}_i > 0 \quad for \quad i \in \Omega \tag{4.13}$$

$$\hat{\mathbf{y}}_i = 0 \quad for \quad i \in \Psi, \qquad \hat{\mathbf{y}}_i > 0 \quad for \quad i \in \Omega \tag{4.14}$$

where

$$\hat{\mathbf{r}} = \mathbf{L}\hat{\mathbf{p}} - \mathbf{h}. \tag{4.15}$$

This theorem can be interpreted as followed. Assuming that $\mathbf{l}_\mathbf{i}^\mathbf{T}$ denotes the ith row vector of $\mathbf{L}$, we define the space, $\{p : \mathbf{l}_i^T\mathbf{p} \geq \mathbf{h}_i\}$. The vector $\mathbf{l}_i$ is normal to the bounding hyperplane of this halfspace. Assume that the point $\hat{\mathbf{p}}$ is interior to the halfspace Ω and on the boundary of the half space Ψ. Our goal is to show that $\hat{\mathbf{p}}$ minimizes the constraint and it is the only unique answer to the problem.

The vector

$$\hat{\mathbf{a}} = \mathbf{T}^\mathbf{T}(\mathbf{T}\hat{\mathbf{p}} - \hat{\mathbf{e}}) \tag{4.16}$$

is the gradient of $\phi(\mathbf{p}) = \frac{1}{2}||\mathbf{Tp} - \hat{\mathbf{e}}||^2$ at $\mathbf{p} = \hat{\mathbf{p}}$. As stated previously, $\hat{\mathbf{y}}_\mathbf{i} = \mathbf{0}$ if $i \notin \Psi$, therefore Eq. (4.12) is simplified as followed

$$\sum_{i \in \psi} \hat{\mathbf{y}}_\mathbf{i}(-\mathbf{l}_\mathbf{i}) = -\mathbf{a}. \tag{4.17}$$

The negative gradient vector at $\hat{\mathbf{p}}$ is expressed as a linear combination of the normals $-\mathbf{g}_\mathbf{i}$ to the constraint hyperplane on which $\hat{\mathbf{p}}$ lies.

We should also prove that the perturbed versions of the vector $\hat{\mathbf{p}}$ do not satisfy the theorem's constraint. A perturbation $\mathbf{u}$ of $\hat{\mathbf{e}}$ such that $\mathbf{u} + \hat{\mathbf{e}}$ remains feasible must satisfy $\mathbf{u}^\mathbf{T}\mathbf{l}_\mathbf{i} \geq \mathbf{0}$ for all $i \in \Psi$. Using the facts that $\mathbf{y}_\mathbf{i} \geq \mathbf{0}$ and multiplying both sides of Eq. (4.17) by $\mathbf{u}^\mathbf{T}$, it is apparent that $\mathbf{u}^\mathbf{T}\mathbf{a} \geq \mathbf{0}$. Therefore, from the expression $\phi(\hat{\mathbf{x}} + \mathbf{u}) = \phi(\hat{\mathbf{x}}) + \mathbf{u}^\mathbf{T}\mathbf{a} + ||\mathbf{Eu}||^2/2$, we see that there does not exist any perturbation vector $\mathbf{u}$ such that it reduces the value of ϕ. Therefore, the obtained value is minimum and the answer is unique.

4.3.1 The Nonnegative Least Squares Algorithm

We have used the Matlab function *lsqnonneg* in our implementations of *NNLS* algorithm. This function works based on the algorithm stated in [47]. Below, we explain the used algorithm. The proof of convergence is included in [47], however we skip over it.

In the general case of *NNLS* algorithm we assume $\mathbf{T}$ to be a $k \times n$ dimensional matrix and $\hat{\mathbf{e}}$ a k dimensional matrix. The n dimensional vectors w and z provide working space. Two index sets are defined as $\mathcal{P}$ and $\mathcal{Z}$ are defined. On the termination of algorithm p is the solution vector and w is its dual.

NNLS Algorithm [47]

1. Set $\mathcal{P}$=NULL, $\mathcal{Z} = \{1, 2, \cdots, n\}$ and $\mathbf{p} = 0$.
2. Compute the n dimensional vector $\mathbf{w} = \mathbf{T}^\mathbf{T}(\hat{\mathbf{e}} - \mathbf{Tp})$.
3. If $\mathcal{Z}$ is empty or $w_j \leq 0$ for all $j \in \mathcal{Z}$ go to step 12.
4. Find the index $t \in \mathcal{Z}$ such that $w_t = \max\{w_j : j \in \mathcal{Z}\}$.
5. Move the index t from the set $\mathcal{Z}$ to $\mathcal{P}$.

6. Let $\mathbf{T}_{\mathcal{P}}$ denote the $k \times n$ matrix defined by

$$\text{Column } j \text{ of } \mathbf{T}_{\mathcal{P}} = \begin{cases} \text{column } j \text{ of } \mathbf{T} \text{ if } j \in \mathcal{P} \\ 0 \text{ if } j \in \mathcal{Z} \end{cases}$$

The n dimensional vector $\mathbf{z}$ is computed as a solution of the least squares problem $\mathbf{T}_{\mathcal{P}}\mathbf{z} = \hat{\mathbf{e}}$. For the values of j where $j \in \mathcal{Z}$, $\mathbf{z_j}$ is assumed to be zero.

7. If $z_j > 0$ for all $j \in \mathcal{P}$, set $\mathbf{p} = \mathbf{z}$ and go to step 2.

8. Find an index $d \in \mathcal{P}$ such that $\mathbf{p}_d/(\mathbf{p}_d - \mathbf{z}_d) = \min\{\mathbf{p}_j/(\mathbf{p}_j - \mathbf{z}_j) | \mathbf{z}_j \leq 0, j \in \mathcal{P}\}$.

9. Set $\beta = \mathbf{p}_d/(\mathbf{p}_d - \mathbf{z}_d)$.

10. Set $\mathbf{p} = \mathbf{p} + \beta(\mathbf{z} - \mathbf{p})$.

11. Move all the indices $j \in \mathcal{P}$, for which $\mathbf{p}_j = 0$, from $\mathcal{P}$ to $\mathcal{Z}$ and go to step 6.

12. Termination.

Finally we have the following properties for the answer

$$\begin{aligned} \mathbf{p_j} &> 0 \quad j \in \mathcal{P} \\ \mathbf{p_j} &= 0 \quad j \in \mathcal{Z}. \end{aligned} \tag{4.18}$$

The vector $\mathbf{p}$ is the solution to the least squares problem

$$\mathbf{T}_{\mathcal{P}}\mathbf{z} = \hat{\mathbf{e}}. \tag{4.19}$$

4.4 The Quantizer Design Details

As mentioned before in Chapter 3, our goal here is to have a quantizer which is matched to the *pdf* of source. We first describe the bit allocation scheme used based on [4]. The allocation algorithm has been changed slightly here to meet our specific conditions of high dimensionality. We then explain the quantizer scheme and finally discuss our GMM training with different number of clusters and in the cases of full and diagonal covariance matrices. The experimental results and comparisons of speech and audio quality after the quantization are presented in the next section.

4.4.1 Bit Allocation Algorithm

Cluster bit allocation

It is assumed that we have the total of b_{tot} bits to allocate to our quantizer. These bits are spread among the clusters and cluster i has b_i bits, where $1 \leq i \leq M$ and M is the number of Gaussians. Let $D_i(b_i)$ represent the mean square distortion of the optimal transform coder of cluster i. Using the high resolution expression for $D_i(b_i)$ as shown in [48], the Karhunen-Leove transform $\mathbf{Q}$ is the best transformation for minimizing the error. In [48] it has been shown that the best transformation, to construct from the quantized values the best estimate in a mean-square sense, is the inverse of the transformation matrix $\mathbf{Q}$.

In [4] both fixed rate and variable rate quantizers have been discussed. For our implementations, we have used a fixed rate scheme.

1. **Fixed rate**: In this scheme the total number of codebooks points is fixed. It is assumed that the total number of quantizer levels is the sum of the number of quantizer levels in each cluster. In other words, $2^{b_{\text{tot}}} = \sum_{i=1}^{M} 2^{b_i}$. Assuming that the clusters are well separated we can use the following upper bound on the quantizer distortion [4]

$$D_{\text{tot}} \leq \sum_{i=1}^{M} \alpha_i D_i(b_i). \tag{4.20}$$

The aim in the design is to minimize the above expression and find the corresponding bit allocation to each cluster. It is shown in [4] that

Theorem 2 (Optimized Fixed-Rate Cluster Bit Allocation) *Subject to the bit rate constraint* $2^{b_{\text{tot}}} = \sum_{i=1}^{M} 2^{b_i}$, *in order to minimize the distortion measure the best bit allocation is*

$$2^{b_i} = 2^{b_{\text{tot}}} \frac{(\alpha_i c_i)^{N/N+2}}{\sum_{j=1}^{M} (\alpha_j c_j)^{N/N+2}}, \quad 1 \leq i \leq M \tag{4.21}$$

where,

$$\mathbf{C}_i = \mathbf{Q}_i \mathbf{\Lambda}_i \mathbf{Q}_i^T \tag{4.22}$$

$$\mathbf{\Lambda}_i = \text{diag}(\lambda_{i,1}, \lambda_{i,2}, \cdots, \lambda_{i,N}) \tag{4.23}$$

$$c_i = \Big[\prod_{k=1}^{N} \lambda_{i,k}\Big]^{\frac{1}{N}}, \quad 1 \leq i \leq M \tag{4.24}$$

and N is the vector dimension.

2. **Variable rate**: In this case the average rate of the quantizer is fixed. Assuming that b_c is the number of bits that specifies the chosen cluster, and assuming that we use b_i bits to quantize a particular cluster then the variable rate constraint is $b_q = b_{\text{tot}} - b_c = \sum_{i=1}^{M} \alpha_i b_i$. The same expression for the total average distortion is used in this case. It is proven in [4] that the number of bits given to each cluster can be obtained by

$$b_i = b_q + \frac{N}{2}\Big[\log_2 c_i - \sum_{j=1}^{M} \alpha_j \log_2 c_j\Big]. \tag{4.25}$$

Bit allocation to each component in a cluster

The code book of each cluster is built using a transform coder [48],[49],[50]. The transformation is done using a unitary matrix $\mathbf{A}$ on the input block $\mathbf{p}$. In [49],[48] the case of basis restricted transform coding where the transform coefficients y_i, $i = 1, 2, \cdots, N$ are quantized independently is considered. They have proven that R_j bits/sample needed for the jth component y_j with variance σ_j^2 and average mean square distortion of D_j does not exceed

$$R_j = \bar{R} + \frac{1}{2}\log_2 \frac{\sigma_j^2}{\left[\prod_{k=1}^{N} \sigma_k^2\right]^{1/N}} \quad \text{bits/sample} \tag{4.26}$$

where $\bar{R}$ is the average bit rate. Using the results of [48] which give the optimum value of the above formula and rephrasing the equation for our case the number of bits allocated for quantizing the jth component of the ith cluster is given as

$$b_{i,j} = \frac{b_i}{N} + \frac{1}{2}\log_2\Big[\frac{\lambda_{i,j}}{c_i}\Big]. \tag{4.27}$$

One improvement to this bit allocation is to add a constraint that the resolution of each quantizer must be nonnegative. A problem that occurred in the implementations was that, for the cases where $\lambda_{i,j}$ is smaller than c_i the *log* term is negative. Since we are using vectors of high dimension the term $\frac{b_i}{N}$ is small. Therefore, we wind up with negative number of bits. In [10] it is shown that using Segall's solution we can guarantee a nonnegative bit allocation. In this approach it is assumed that the scalar components to be quantized have identical normalized pdf. The mean squared error incurred in optimally quantizing

each element is called $w(b)$. Let the inverse of $w(b)$ be $J(w)$. The distortion function is monotonically decreasing with increasing resolution b, therefore the inverse is well defined. Segall's solution gives the optimal bit allocation,

$$b_i = \begin{cases} J(\frac{\theta^*}{\sigma_i^2} w'(0)), & \text{if } 0 < \theta^* < \sigma_i^2 \\ 0, & \text{if } \theta^* \geq \sigma_i^2 \end{cases} \tag{4.28}$$

where σ_i^2 is the variance of each component and θ^* is the unique root of the equation

$$S(\theta) = \sum_{i:\sigma_i^2 \geq \theta^*} J(\frac{\theta}{\sigma_i^2} w'(0)) = b_{\text{tot}}. \tag{4.29}$$

However, for our implementations, in order to simplify the computations of inverse function needed above as well as finding the roots of Eq. (4.29) we simply changed the negative values to zero. However, we have to check the bit rate constraint. In other words we need to make sure that $b_i = \sum_{j=1}^{N} b_{i,j}$ or the summation is slightly less than the total bits allocated to a specific cluster. To reach this aim, we have changed the algorithm slightly. This procedure is discussed as follows:

1. Assume that dimension k has $-m$ bits.
2. Choose the jth dimension with the biggest value of variance λ_j. Then add m bits to the number of bits allocated to this dimension using Eq. (4.27) and set the number of bits in dimension k, equal to zero.
3. Discard both of the dimensions and repeat steps 1 and 2 until there is no more negative number of bits allocated to any dimension.
4. Round the fractional bits allocated to the dimensions for practical implementation.

Although, the optimality is lost here to avoid the problem of negative bit allocation this procedure is used. It is shown in experimental results section that the performance of the quantized version of speech is still acceptable.

4.4.2 Quantizer Scheme

The following steps are done in the quantization

- An input vector $\mathbf{z}$ is quantized using all the clusters.
- The quantization process in each cluster is shown in Fig. 4.6. In cluster i we subtract the cluster mean μ_i from the input vector $\mathbf{z}$. The next step is decorrelating the components of each input vector. As discussed in the previous section the transformation matrix $\mathbf{A}$ is a unitary matrix. It is known that [48], the best transform will be the Karhunen-Leove transform, therefore the eigenvector matrix $\mathbf{Q}_i^T$ of the covariance matrix $\mathbf{C}_i$ is used as the transformation (Section 4.2.1).
- At this step the quantization is simplified to scalar quantization which will be explained.
- The decorrelation process is done using the inverse of the transformation matrix $\mathbf{Q}_i^T$. The eigenvector matrices are unitary therefore this inverse matrix will be equal to $\mathbf{Q}_i$.
- Having M quantized versions of the input vector, we chose the vector, which minimizes the distortion measure. In [4] the Log spectral distortion has been chosen as this measure. In our implementations Minimum Mean Square Error (MMSE) has been used instead.

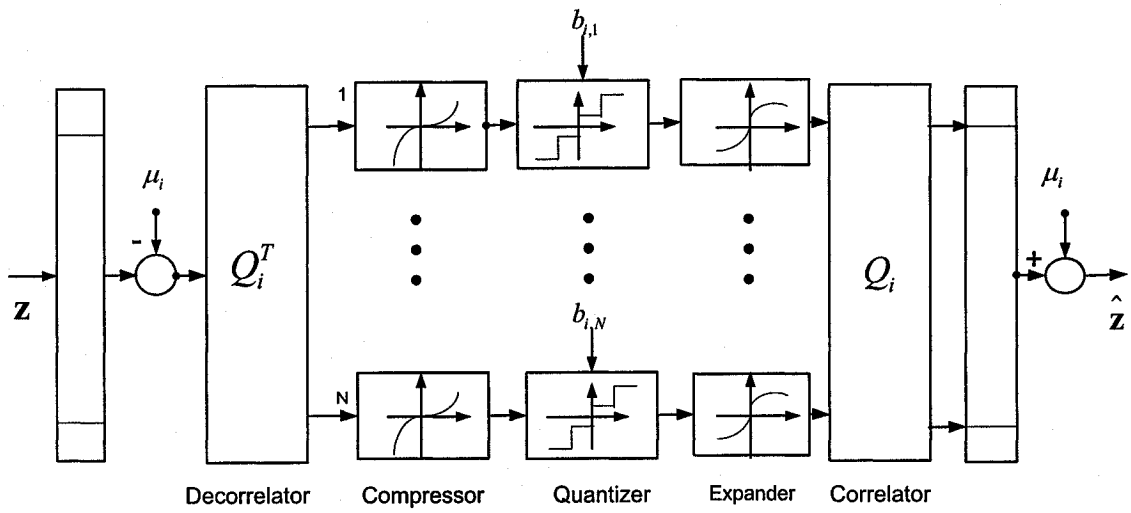

Fig. 4.6 Cluster quantization [4]

Scalar quantization scheme

1. **Compressor and expander** By taking the statistical properties of the signal in to consideration the distortion can be minimized by proper level distribution in the quantizer based on the *pdf* of the signal [51]. In practice this is done using a compressor function followed by a uniform quantizer and an expander which is the inverse of the compressor. The compressor function, has more number of levels around zero. It is shown that using this method the distortion is largely independent of the signal statistics.

   Assuming the probability density of the individual scalar components to be $f(y)$ we have used Panter and Dite's compressor function [51], which minimizes the mean square error of the scalar companding. It has the following form,

$$C(x) = \frac{\int_{-\infty}^{x} f(y)^{1/3} dy}{\int_{-\infty}^{\infty} f(y)^{1/3} dy}. \tag{4.30}$$

   The compressor function is a monotonic nondecreasing function [52]. In order to implement the compressor function we have simplified Eq. (4.30). The computations are described below.

   For the purpose of implementation, the scalar *pdf*s have been assumed to be Gaussians with mean zero and variance 1. The expression for $C(x)$ can be written as

$$C(x) = \frac{\int_{-\infty}^{x} \left(\frac{1}{\sqrt{2\pi}}\right)^{1/3} \exp\frac{-y^2}{2\times 3} dy}{\int_{-\infty}^{\infty} \left(\frac{1}{\sqrt{2\pi}}\right)^{1/3} \exp\frac{-y^2}{2\times 3} dy}. \tag{4.31}$$

   In order to implement this function we have used the Matlab *erf* function. *erf* function is defined as $\mathrm{erf}(x) = \frac{2}{\sqrt{\pi}} \int_0^x \exp -u^2 du = 2Q(\sqrt{2}x) - 1$ where $Q(x) = \int_{-\infty}^{x} \frac{1}{\sqrt{2\pi}} \exp -x^2/2 dx$. Changing the variable of integration in Eq. (4.31), from y

to $t = y/\sqrt{3}$ we can rewrite the equation as

$$\begin{aligned} C(x) &= \frac{\int_{-\infty}^{x/\sqrt{3}} \frac{1}{\sqrt{2\pi}}^{1/3} \exp\frac{-t^2}{2}\sqrt{3}dt}{\int_{-\infty}^{\infty} \frac{1}{\sqrt{2\pi}}^{1/3} \exp\frac{-t^2}{2}\sqrt{3}dt} \\ &= \frac{\int_{-\infty}^{x/\sqrt{3}} \frac{1}{\sqrt{2\pi}} \exp\frac{-t^2}{2}dt}{\int_{-\infty}^{\infty} \frac{1}{\sqrt{2\pi}} \exp\frac{-t^2}{2}dt} \\ &= 2Q(x/\sqrt{3}). \end{aligned} \tag{4.32}$$

The problem here, is that there is no direct expression for the inverse of $C(x)$. Therefore, we have to use a table–look up implementation. The transformed scalar components y_j are assumed to be Normalized Gaussians. Therefore,

$$p(y_j \in [-l, \cdots, l]) = \int_{-l}^{l} f(y_j)dy = \frac{1}{\sqrt{2\pi}} \int_{-l}^{l} \exp\frac{-y^2}{2}dy. \tag{4.33}$$

Let l be equal to 5. The value of the above probability will then be equal to 0.9996. Therefore, it is highly probable that all the Gaussian components lie in this interval. We have mapped the interval $k = [-5, 5]$ with steps of $\delta = 0.0001$, using the closed loop expression $k_{j,comp.} = 2Q(k_j/\sqrt{3})$, where k_j is each individual element in the above interval and $k_{j,comp.}$ is the corresponding compressed element.

In order to expand each element, the expander compares the quantized version of each component with all of the components in the interval $k_{comp.}$. It chooses the element with the smallest distance in this interval, i.e $k_{m,comp.}$. The corresponding component k_m in the interval k is considered as the result of the expansion operation. In order to have higher precision we have to decrease the step size δ, therefore we have a trade off between precision and computational complexity.

2. **Uniform scalar quantization** The uniform scalar quantizer has been implemented in a closed form here. Assuming y_j to be the jth component to be quantized using a uniform scalar quantizer $S_{i,j}$ of cluster i with $L_j = 2^{b_{i,j}}$ which is obtained using the bit allocation algorithm discussed in Section (4.4.1) on page (52), the quantized value

$\hat{y}_j$ is given as

$$\hat{y}_j = \frac{\text{round}(y_j L_j + 0.5) - 0.5}{L_j}. \tag{4.34}$$

4.4.3 Comparison of the performance of different GM models used in the implementations

For the purpose of parameterizing our data we had different choices for the mixture model. Although, using full covariance matrices, models the data in a more precise way, it is computationally inefficient to use full covariance matrices for high data dimensions. For a vector of dimension N the covariance matrix has $\frac{N(N-1)}{2} + N = \frac{N(N+1)}{2}$ elements. Therefore, each mixture is determined using $\frac{N(N+2)}{2}$ parameters, since it has a mean vector with dimension N and a weight number. This number will be equal to 7320 for vectors of dimension 120. For the case of 60 dimensional vectors this number is equal to 1860. This means doubling the number of dimensions will result in quadrupling the size of the training data, since the number of parameters is quadrupled and size of training data is directly proportional to the number of parameters in our model. It is shown in [35] that for vectors of dimension N with m training samples where normally $m \geq N$ the overall complexity of calculating the individual discriminant function, which is linearly proportional to the number of parameters (order of N^2), is dominated by $O(N^2 m)$ (here by discriminant function we mean each of the basis functions in the GMM).

We encountered the problem of over–fitting, while using vectors of dimension 120 for training the full covariance matrices. Another burden is the problem of singularity. The EM algorithm can result in singular covariance matrices and the algorithm terminates. To combat this problem, a minimum threshold is set for the covariance matrix. If, in any iteration, it goes below the threshold the covariance is set equal to the threshold. The other consideration is adding small matrices to the covariance matrix in each iteration of the maximization step of the EM algorithm in order to regularize them. This issue is more thoroughly discussed in [40] for the case of GM regression algorithms. We added these offset matrices in the form of $\epsilon \times \mathbf{I}_{N \times N}$. There is no specific theoretical rule describing how to find the value of ϵ. For 120 dimensional vectors the values of 10^{-12} and 10^{-14} seemed to be reasonable for full and diagonal covariance matrices.

For the purpose of illustration, the 1 and 2 dimensional histograms are given for data vectors of dimension 60 with 8 and 16 mixtures (Figs. 4.7, 4.8 and 4.9). The center of

each cluster in GM is determined by the cluster mean and the shape is determined by the cluster covariance. Therefore, the center of each 2D histogram is on the corresponding elements of the mixture mean vector. The mixture mean is defined as as the sum of the means of each cluster multiplied by the cluster weight. The GM mean vectors have been initialized to a random vector using the *randn* function in Matlab. The covariances have been initialized to $N \times N$ matrices with all elements equal to one. We have used 8 and 16 mixtures for the implementations. We have used both diagonal and full covariances for 8 mixture Gaussians. The GMM training is done using 400 000 vectors of dimension 60. The vectors used in this case are transformed to frequency domain using MDCT and then the training is done. Comparing the modelling power of full covariance matrices with diagonal ones, we see that we do not have a big gain, using full covariance matrices. The reason is the fact that we loose some degree of precision when we add the offsets to the covariances.

In Figs. 4.8, 4.9 it is seen that the full covariance matrices result in 2D histograms with lighter colors around origin (in comparison with the original histogram). Darker colors show denser areas. Our experiences showed that, in all cases of using full covariance matrices the offset element causes this change of distribution near origin. The reason is the need of adding bigger offset elements, to avoid singularity. The diagonal elements of the covariance are greater than expected. This causes a bigger density concentration near the mean of the specific frequency bin (the mean is approximately zero).

The resulting histograms for 120 dimensional vectors have been included in Fig. 4.10 as well. The frequency transformation used in this case is DFT. The case of 8 and 16 diagonal mixtures have been considered in the histograms. Comparing this figure with Figs. (4.8, 4.9), we can see the areas with lighter colors are bigger here, which shows that the offset added to the covariance in EM iteration is larger than the offset for lower dimensional data.

We previously discussed the use of DCT to uncorrelate the data vectors. This in addition to the facts mentioned above and the computational complexity of full covariance matrices accounts for choosing diagonal covariances as the basis of our quantizer design. The results are discussed in the next section.

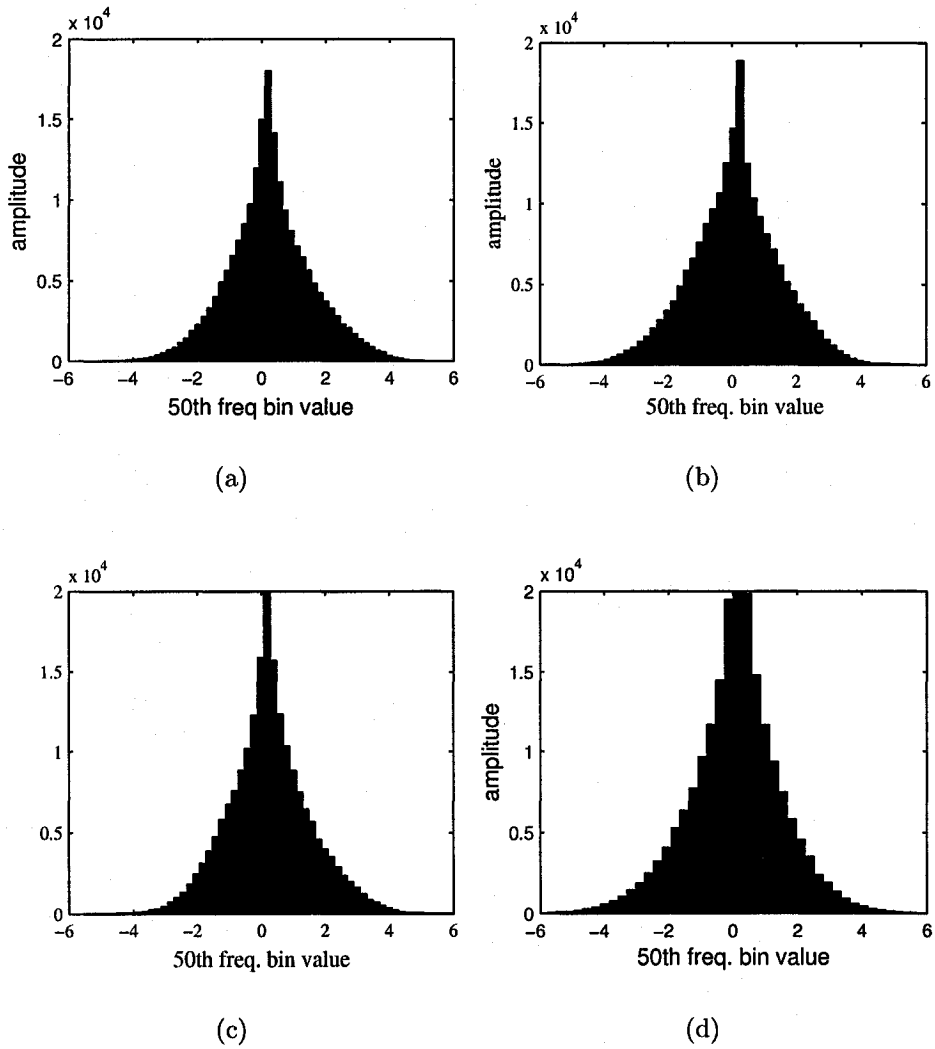

Fig. 4.7 1 dimensional histogram of the 50th freq bin for MDCT transformed vectors of dimension 60. a) Original b) Regenerated using 8 diagonal covariance matrix mixtures. c) 16 diagonal mixtures. d) 8 full mixtures.

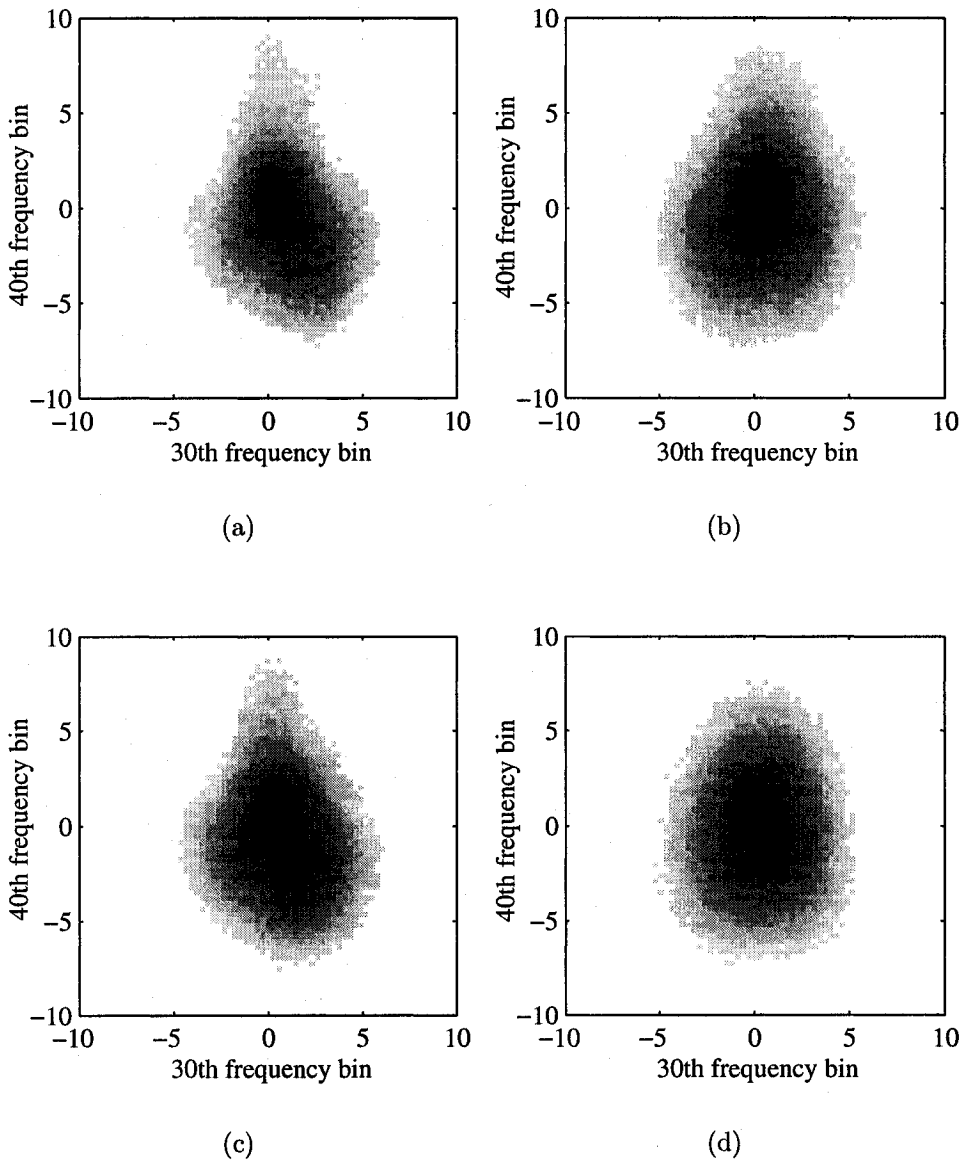

Fig. 4.8 2D Histogram of the 30th and 40th freq bins for MDCT transformed vectors of dimension 60. a) Original b) Regenerated using 8 diagonal covariance matrix mixtures. c) 16 diagonal mixtures. d) 8 full mixtures.

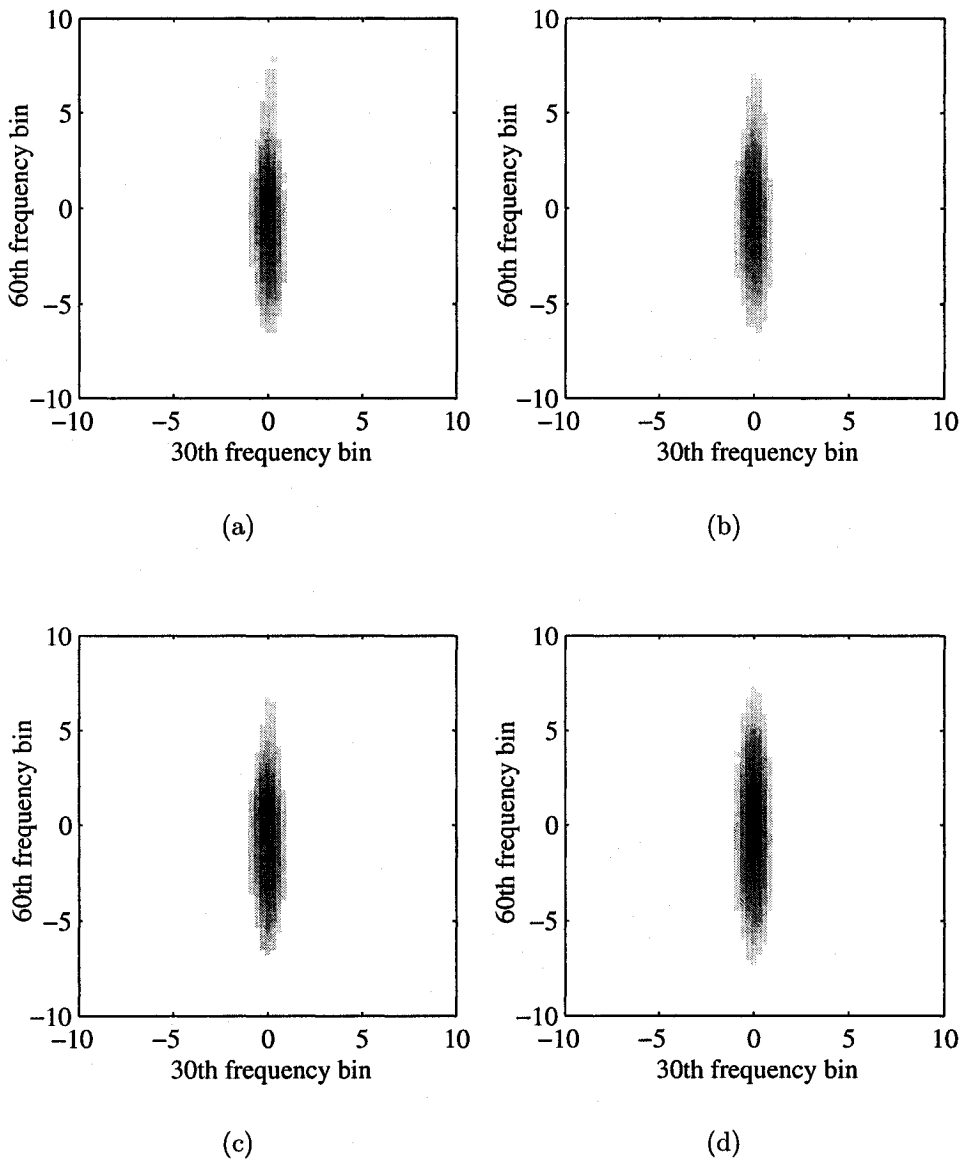

Fig. 4.9 2D Histogram of the 30th and 60th freq bins for MDCT transformed vectors of dimension 60. a) Original b) Regenerated using 8 diagonal covariance matrix mixtures. c) 16 diagonal mixtures. d) 8 full mixtures.

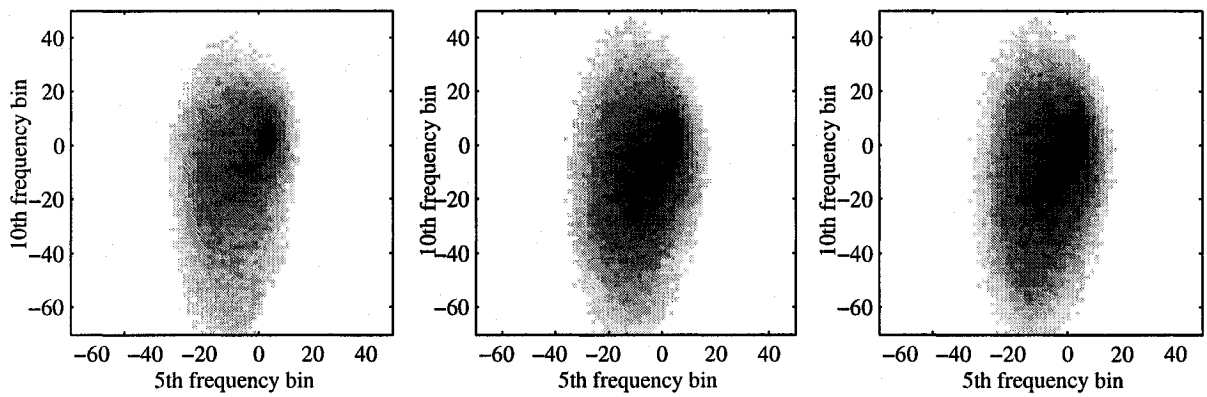

Fig. 4.10 2D Histogram of the 5th and 10th freq bins for DFT transformed vectors of dimension 120. a) Original b) Regenerated using 16 diagonal covariance matrix mixtures. c) Regenerated using 32 diagonal mixtures.

4.5 Experimental results

One concern in the design of the quantizer is the choice of dimension. We obtained slightly better performance for the 120 dimensional vectors. We evaluated the quality of the quantized speech using both subjective test as well as *PESQ* standard [53]. "Perceptual Evaluation of Speech Quality" (PESQ) is the recent winner of the ITU standard P.862 for measurement of speech quality. The PESQ measures the listening quality in intrusive (test call) applications and is accurate in predicting subjective quality (MOS) in a wide range of applications. This value is a number between 1 and 5. The MOS values are given in Table (4.1).

Table 4.1 MOS score descriptions.

Rating	Speech Quality	Level of distortion
1	Unsatisfactory	Very annoying
2	Poor	Annoying but not objectionable
3	Fair	Perceptible and slightly annoying
4	Good	Just perceptible but not annoying
5	Excellent	Imperceptible

We compared the original files with themselves and got 4.5 as the output. Therefore, this will be the highest possible score. The results of the implementations have been considered for both DFT and MDCT transforms. The MOS values are higher for DFT. We have assumed perfect phase reconstruction in our simulations. The MDCT transform of a signal is always a real number. Therefore, the phase reduces to positive and negative signs in this case. These signs carry much less amount of information comparing to the case of DFT, where we have real numbers as the phase. Therefore, the reconstruction effect will have different results in the two cases. Tables 4.2 and 4.3, give the MOS results from *PESQ* software, for different bit rates of 4 kBits/sec, 8 kBits/sec and 16 kBits/sec, using DFT for 120 dimensional vectors.

Table 4.2 MOS for quantized speech and audio using 16 diagonal mixtures (DFT transformed followed by the perceptual transformation and DCT)

	Speech file	Audio file
Bit Rate (kHz)	MOS	MOS
4	2.6	2.9
8	2.88	3.5
16	3	4.1

Table 4.3 MOS for quantized speech and audio using 32 diagonal mixtures (DFT transformed followed by theperceptual transformation and DCT)

	Speech file	Audio file
Bit Rate (kHz)	MOS	MOS
4	2.55	2.9
8	2.8	3.5
16	2.9	4.1

Table (4.4) gives the values of MOS for the case of using MDCT as the transform. We can see a great quality degradation in this case.

The bit allocation scheme using 16 diagonal mixtures, in both DFT transformed and MDCT transformed vectors, is given in Tables (4.6 and 4.7). In both cases the bit rate is 16 kBits/sec, the frame size is 120 samples. The bits are allocated to cluster and within each cluster to the components. The tables show the bit allocation within the cluster with

Table 4.4 MOS for quantized speech and audio using 16 diagonal mixtures (MDCT transformed followed by the perceptual transformation and DCT)

	Speech file	Audio file
Bit Rate (kHz)	MOS	MOS
4	2.4	2.6
8	2.7	3.1
16	2.9	3.7

the highest probability (weight number) as an example. The number of bits allocated to dimensions is in decreasing order. Therefore, after DCT transform we need more number of bits to code lower frequency bins.

4.5.1 Comparison of results with ED reference coder

In addition to the work presented so far, another coding scheme is implemented. This coder which is named the Excitation Distortion reference coder (ED), uses the same frequency to excitation domain transformation as the one discussed so far. It then divides the input vectors to a number of subbands. Scalar quantization with the same step size are applied in each subband. The quantizer's step sizes are iteratively decreased in each subband. The subband that minimizes the corresponding distortion measure in each iteration is chosen and its step size is decreased, other step sizes remain unchanged. This process is repeated till we meet a minimum required distortion. The design and implementation details are discussed in Appendix A.

The results for the reference coder are given in Table 4.5. Comparing the MOS in this case with the reference coder we see that for the same bit rate and frequency transformation, the GMM based quantizer has a better performance.

Table 4.5 MOS comparison for different entropies (20 subbands, DFT transformed followed by the perceptual transformation and DCT)

	Speech file	Audio file
Entropy (bit/sample)	MOS	MOS
0.5 (4 kHz)	1.98	2.10
0.75	2.15	2.25
1 (8 kHz)	2.50	2.91
1.2	2.60	3.50
2 (16 kHz)	2.80	4.00

Table 4.6 Bits allocated to each dimension of the cluster with the highest probability, using 16 mixtures (DFT transformed data, followed by the perceptual transformation and DCT)

dimension	No. of Bits	dim	No. of Bits	dim	No. of Bits	dim	No. of Bits
1	8	31	3	61	1	91	0
2	8	32	3	62	1	92	0
3	7	33	3	63	1	93	0
4	7	34	3	64	1	94	0
5	6	35	3	65	1	95	0
6	6	36	3	66	1	96	0
7	6	37	3	67	1	97	0
8	6	38	3	68	1	98	0
9	6	39	2	69	1	99	0
10	5	40	2	70	1	100	0
11	5	41	2	71	1	101	0
12	5	42	2	72	1	102	0
13	5	43	2	73	1	103	0
14	5	44	2	74	1	104	0
15	4	45	2	75	1	105	0
16	4	46	2	76	0	106	0
17	4	47	2	77	0	107	0
18	4	48	2	78	0	108	0
19	4	49	2	79	0	109	0
20	4	50	2	80	0	110	0
21	4	51	2	81	0	111	0
22	4	52	2	82	0	112	0
23	4	53	2	83	0	113	0
24	4	54	2	84	0	114	0
25	4	55	2	85	0	115	0
26	3	56	2	86	0	116	0
27	3	57	1	87	0	117	0
28	3	58	1	88	0	118	0
29	3	59	1	89	0	119	0
30	3	60	1	90	0	120	0

Table 4.7 Bits allocated to each dimension of the cluster with the highest probability cluster using 16 mixtures (MDCT transformed, followed by the perceptual transformation and DCT)

dimension	No. of Bits	dim	No. of Bits	dim	No. of Bits	dim	No. of Bits
1	6	31	4	61	1	91	0
2	6	32	4	62	1	92	0
3	6	33	4	63	1	93	0
4	7	34	4	64	1	94	0
5	6	35	4	65	1	95	0
6	6	36	4	66	1	96	0
7	6	37	4	67	1	97	0
8	6	38	4	68	1	98	0
9	5	39	4	69	1	99	0
10	5	40	4	70	1	100	0
11	5	41	3	71	1	101	0
12	5	42	3	72	1	102	0
13	4	43	3	73	1	103	0
14	4	44	3	74	0	104	0
15	4	45	2	75	0	105	0
16	4	46	2	76	0	106	0
17	4	47	2	77	0	107	0
18	4	48	2	78	0	108	0
19	4	49	2	79	0	109	0
20	4	50	2	80	0	110	0
21	4	51	2	81	0	111	0
22	4	52	2	82	0	112	0
23	4	53	2	83	0	113	0
24	4	54	2	84	0	114	0
25	4	55	2	85	0	115	0
26	4	56	2	86	0	116	0
27	4	57	1	87	0	117	0
28	4	58	1	88	0	118	0
29	4	59	1	89	0	119	0
30	4	60	1	90	0	120	0

Chapter 5

Conclusion

This thesis has focused on the design and implementation of a speech and audio coder. The main contribution is the use of perceptual transformations and gaussian mixture models in the design of the quantizer. However, we have quantized the excitation patterns to have more fidelity of the quantized speech.

5.1 Summary of Work

We discuss the work done and the idea behind the our domain transformation and use of mixture models for VQ. Also the Excitation Domain coder which is explained in Appendix A is discussed briefly here.

5.1.1 Perceptual Domain Transformation

In order to consider the properties of human auditory system in the design of the quantizer a perceptual transformation has been used. The speech and audio files have been divided to frames of size N, where N is equal to 60, 80 and 120. These frames are then transformed to frequency domain. The transformation to frequency domain is done using either DFT or MDCT. In the first case the vectors are complex numbers and in the second case we have real numbers. Therefore, the MDCT phase will reduce to positive or negative signs. Since, we assume perfect phase reconstruction in our quantizer design the resultant quantized files have a better performance when we use DFT as the transformation.

The perceptual transformation used here has been explained thoroughly in Section 2.3.

The smearing function used is based on Moore's model and is in the form of *roex*. This function depends on two variables. One is the central frequency at which we need to evaluate the excitation pattern. The other function is the range of frequencies for which the function is determined. The excitation patterns are obtained by weighting the power spectrum vectors using the smearing function and integrating this product over the frequency range.

The middle/outer ear transfer function's effect has been considered as a matrix which has been multiplied by the excitation domain transformation's matrix. Using the overall transformation we wind up with a set of new vectors which are in the form of excitation patterns. The idea behind this change of quantization domain (from frequency to excitation) is the concept of human auditory system. Since, the new vectors are weighted by functions which are based on auditory properties, the distortion measure used for quantization will be a weighted distortion. Minimizing this function, we will have more fidelity in the quality of the quantized speech and audio.

Any transformation of the space of power spectrum which spans R^{N+} will result in the vectors which no longer span R^{N+} (Section 4.3). The region over which the new vectors are defined is limited to a polyhedron. The problem that arises here is that, quantizing the points which are near the sides of this polyhedron may lead to points outside the acceptable region. Therefore, using the inverse perceptual transformation at the decoder end may result in spectral vectors with negative elements. To avoid this problem, we have implemented a Non Negative Least Square algorithm using Matlab. Therefore, the outlier points are projected to the nearest point on the sides of the polyhedron. The mean square of error is minimized in this algorithm.

5.1.2 Gaussian Mixture Model based VQ

The quantizer scheme is based on the quantizer proposed in [4]. This quantizer takes the speech vectors and after mean subtraction and decorrelation uses a simple scalar quantization which is based on assuming Gaussian *pdf*. The idea behind choosing GMMs is their flexibility, bit rate scalability, variable rate coding, interoperability (which means the ability of switching between memoryless quantizers and quantizers with memory) and learning (Section 3.4).

In order to decorrelate data vectors we need their statistics. However, speech is a random

non-stationary source. Therefore, in order to model its *pdf* we need a linear combination of a set of basis function. Gaussian mixtures have been used for this aim. It has been shown that Gaussians are good radial basis functions. In order to find the parameters of mixtures, a set of speech and audio vectors were used. For the case of 60 dimensional vectors we used 400 000 vectors. The training has been done using both full and diagonal covariance matrices. As stated before in Section (4.4.3) for high dimensional data (120 in this case) the problems of singularity and under estimation occur. Since we have used DCT as an approximation to Karhunen–Loeve which decorrelates data we can use diagonal covariances to combat these problems. The reason is that the decorrelated vectors have very small off–diagonal elements which can be neglected in modelling.

The quantization has been done using M quantizers, where M is equal to the number of mixtures. The quantizers all quantize the input vector separately and in parallel and the result of the quantization which minimizes the corresponding distortion measure is chosen.

The bit allocation is based on the fact that total number of levels should be equal to the sum of the number of levels of each quantizer (number of levels given to each cluster). The lower frequency bins are of more importance and more number of bits is allocated to quantize them.

5.1.3 Comparison of results

In order to evaluate the performance of the GM model, the data using the given statistics of the GM are regenerated. The histograms are compared with the histograms of the original data. It is seen that the GM model has a good performance, unless we have data with high dimensions and want to use full covariances. In this case we need a higher number of training vectors and offset values should be added to the covariance matrix in each step of the EM algorithm.

We have used the PESQ software to evaluate the resulting performance. The bit rates of 4, 8 and 16 kbits/sec have been considered. The Mean Opinion Score is given, out of a 5 scale using this software.

In order to compare, an ED coder has been implemented in Appendix A. This coder uses the same transformation of frequency to excitation domain. It then divides the input to a number of subbands. Scalar quantization is used in each subband and the step sizes are iteratively decreased. The algorithm is terminated when the distortion is below some

threshold. In order to compute the bit rate entropy has been used in this case. Finally the MOS results are given in this case (Table A.2).

Comparing the MOS results of Table (4.2) with Table (A.2), we see that the numbers are in the same range and the proposed GMM coder has a slightly better performance (on average 0.2 to 0.3 higher MOS).

5.2 Future Work

This section will detail the future possible directions to be considered for improving the current work. The major concepts behind the design of the current coder are two fold, first the concept of perceptual domain transformation and second the use of a GMM based vector quantizer.

5.2.1 Considering Phase quantization in Coder Design

We have assumed perfect phase reconstruction for DFT transform. Also, we assumed the sign values to be perfectly reconstructed for MDCT transformation. However, in a real coder the effect of phase and sign have to be considered. Therefore, a direction for further works can be design of proper phase and sign quantizers for coder, and compare the results of MDCT and DFT to see which transform works better. Pobloth and Kleijn have considered the effects of phase perception in speech coding and have defined a perceptual phase capacity in [54] as the size of a codebook of phase spectra in a perceptually accurate model. Kim has proposed a perceptual phase coding in [55], where more bits are allocated to phase components which are perceptually more important. In [11] the sign quantization for MDCT has been considered. This scheme is described more in Section (4.1.2).

5.2.2 Transformation to New domain

Although we have used the Moore's model here and defined a new domain for coding, one can consider other possible representations of data in frequency domain. Different perceptual domain matrices can be used for implementations. Also except MMSE, other distortion measures can be used to see whether there will be an improvement in the performance.

5.2.3 Mixture Modelling

An important issue is the choice of a proper mixture model for data vectors. We focused on the Gaussians with diagonal covariance matrices. The main problem with the full covariances here is the high dimensionality of the data vectors. However, using bigger data bases for training can help solve this problem. In the cases that we encounter singularity in the steps of the EM algorithm we used offset matrices to solve this problem. One can also consider different offset values and find an optimum value, since the values in our simulations were obtained empirically and are not optimum. The choice of the covariance matrix can be something between full and diagonal. This way we gain more precision and at the same time the computational complexity will still be less than the case of full covariance matrices a good question will be how to choose this matrix and which structure should be imposed.

Another issue is the number of mixtures used in modelling. The limit on the number of training vectors poses a restriction on the number of mixtures. Using bigger data bases, one can model data with more number of mixtures. Although, it seems that we will not gain a lot after some point, this can be a possible direction. In our case use of 16 and 32 mixture models gave almost the same result. However, if we increase the number of mixtures above some value we no longer can assume that clusters are well separated. Therefore, the bit allocation given for the quantizer design using high resolution expression for distortion is no longer valid. The choice of proper covariance matrix and number of mixtures is an issue which can be studied.

5.2.4 Bit allocation

This scheme uses a fixed rate bit allocation. Variable bit allocation as considered in [4] can be compared with this case. Because of the high data dimensions at some points we encountered negative bit allocation (Section 4.4.1). To solve this problem we set the number of bits in dimensions with negative bit allocation equal to zero and decreased the same number of bits from the dimensions with positive number of bits in order to satisfy the given bit rate constraint in each case . This algorithm is not optimal anymore. Therefore, one can look at different bit allocation algorithms like the one proposed in [10] for high dimensional VQ to improve the performance.

Finally, to evaluate the performance of the coder we used the ED coder which allocated

the bits based on a greedy algorithm. However, other coders may be implemented to give more comparison.

Appendix A

Excitation Distortion Reference Coder

The excitation distortion reference coder is a transform coder which works based on an iterative step size change (a greedy algorithm). We have implemented this coder in the excitation domain and used the same transform matrices which we previously used for the GMM based quantizer. In order to compute the bit rate, the entropies of coefficients in all dimensions have been computed. This coder benefits from some basic concepts used in MP3 coding. The quantization procedure and the results are discussed below.

A.1 The quantizer scheme

The reference coder is based on quantization of subbands in the same excitation domain as the original coder proposed in the thesis. Input audio and speech are divided to frames of length N and each frame is represented in the frequency domain using a DFT. We then transform the frames using the same power spectrum to excitation matrix used in Chapter 2. Each transformed frame in the Excitation domain undergoes a DCT transformation for decorrelation.

Frames are divided into a number of subbands. Scalar quantization is done for each coefficient and the quantizer step size is the same for all of the coefficients in a subband. The number of samples in different subbands are equal which is different from the Bark scale distribution used in MP3 coding.

The goal is to find the optimum step size for each subband such that the quantized

version of speech and audio satisfies the bit rate constraints. Uniform scalar quantizers with the same step size are used in each of the bands. The quantization is done frame by frame.

In the end, to compare the resulting quantized files with our proposed coder the bit rate has to be measured. Therefore, we calculate the entropy of each coefficient of the quantized file. The average entropy is considered as a measure for the number of bits needed to encode each coefficient.

Subbands are given initial step sizes which are decreased using a greedy algorithm. In each iteration of this algorithm the distortion is measured. Since we are decreasing the step size, the *average* MMSE distortion will decrease in each iteration.[1] The greedy algorithm for each frame terminates when the frame distortion is below some specific threshold value. The quantization is discussed below more thoroughly,

Step size initialization: In the first iteration it is desired for all components to be quantized to zero. Assuming the quantizer step size to be q it is known that for scalar quantization the error is always smaller than $\frac{q}{2}$ [56]. Therefore, in each subband the step size is chosen to be slightly more than twice the greatest coefficient in that subband.

Quantizer Initialization: Using the given step size, a uniform quantization is performed for each component. This scheme is implemented in the following symmetrical mid-tread closed form,

$$\hat{x}_j = \delta_i \times [\![\frac{x_j}{\delta_i}]\!] \tag{A.1}$$

where x_j is the jth component of the input vector after DCT transform, δ is the ith subband's step size, $[\![]\!]$ is the round operation and $\hat{x}_j$ is the quantized version of the input. In this representation of scalar quantizer round$(\frac{x_j}{\delta_i})$ represents the index of each quantized coefficient which may be positive or negative. In the first iteration, using the initial value of the step size, all of the quantized components of the subband are zero. At this step the distortion is computed. We have used the mean square distortion measure here which we used as well in the GMM quantizer. The initial distortion is equal to the signal itself.

Quantization Scheme: A threshold value is chosen empirically for the distortion. Decreasing this value will result in better quality of quantized speech and a higher bit rate and entropy. For a specific bit rate we have computed the average number of bits per sample or entropy. The threshold value given for distortion is related to entropy and has

[1]The actual MSE may not decrease monotonically with decreasing the step size.

been changed until we found a threshold which results in value which is a little bit smaller than the desired entropy for the quantized file. The distortion comparison is as follows. In each iteration of this algorithm the distortion is compared to the chosen threshold. If it is above that threshold the algorithm either continues, otherwise the required bit rate is attained and the frame quantization algorithm terminates.

The entropy is related to MMSE distortion as follows. Having smaller distortion on average means that we gain more fidelity using our quantizer. Therefore, we need larger number of levels and in other words larger number of bits in the quantizer. The number of bits required to code a sample is defined as entropy. The details about the entropy computation is explained in Section (A.2).

The initial step size is decreased by a factor α smaller than one in each iteration. If the factor is too small it will result in loss of precision. On the other hand, a big factor like $\alpha = 0.99$ will cause the algorithm to take a long time to converge. We chose $\alpha = 0.9$ for the purpose of simulation. Assuming that for a frame of length N we have m subbands the process for each iteration is explained below,

1. Initialize i to be equal to zero.
2. Decrease the ith frame's step size by α. Quantize the input vector using Eq. (A.1).
3. Compute the MMSE distortion. Subtract the resulting distortion from the distortion given in the previous iteration and call it Δ_i.
4. Set the ith frames step size back to its previous value.
5. Increase i by 1, if $i \leq m$ then go to step 2 otherwise go to step 6.
6. Chose the value of i for which the absolute value of Δ_i is maximum. Decrease the step size of ith subband by the factor α and let other subbands' step sizes to remain unchanged.

If we decrease the number of subbands we lose quality. Although, having N scale factors gives the best performance in this case, we need to send all these coefficients as side information to the decoder. Therefore, there is a trade off between the subband number quality of the quantized speech and the side information needed to be sent.

At the decoder part the sent indices are mapped back to a quantized value. The inverse DCT is followed by the inverse of the Excitation domain transformation. Finally, using the

inverse DFT we map the vectors back to time domain. The overal block diagram of this coder and decoder is depicted in Fig. A.1.

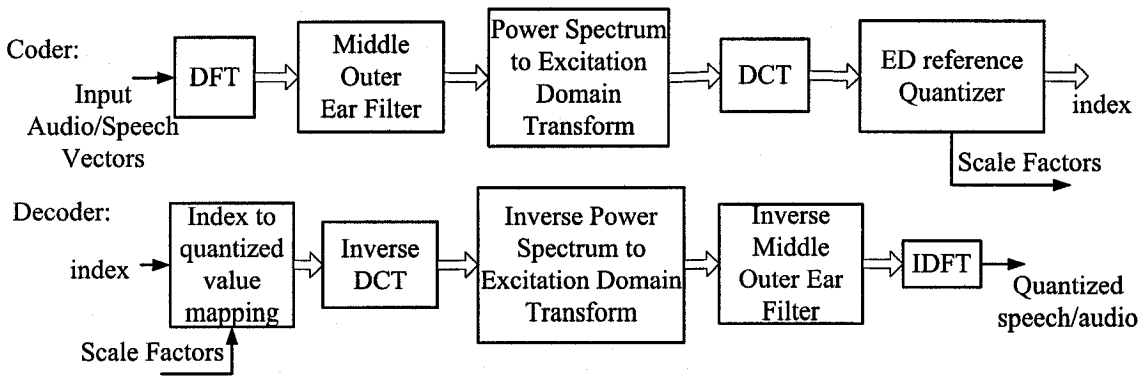

Fig. A.1 The overall scheme of the reference coder/decoder

A.2 Entropy Computation

Assuming a sample space $\mathbf{X} = \{a_1, \ldots, a_K\}$ the *self information of the event* $x = a_k$ *is defined as* [57]

$$I_{\mathbf{X}} = \log \frac{1}{P_{\mathbf{X}}(a_k)}. \tag{A.2}$$

This can be interpreted as the *a priori* uncertainty of the event $x = a_k$ or the amount of information needed to resolve this uncertainty. The entropy of an ensemble is defined to be the average value of the self information and is given by

$$H(\mathbf{X}) = -\sum_{k=1}^{K} P_{\mathbf{X}}(a_k) \log P_{\mathbf{X}}(a_k). \tag{A.3}$$

Entropy can be interpreted as the average amount of uncertainty in $\mathbf{X}$.

In the case of the ED reference coder each dimension of the quantized vector can be considered as a set of symbols. The entropy computation is explained below,

1. The kth coefficient of each frame is saved in a vector $\mathbf{V_k}$, where $1 \leq k \leq N$.

2. In order to compute the entropy of $\mathbf{V_k}$ we need a normalization process. The value of each level of the quantizer equals the level's index multiplied by the quantizer step

size. The step sizes for a specific subband are different for different frames. The quantized coefficient is divided by its corresponding step size to give the index.

3. The probability of occurrence $P_{\mathbf{X}}(a_k)$, of each index of the quantizer is computed.

4. The entropy $H(\mathbf{X})$ of the kth dimension using Eq. (A.3).

The average number of bits for each sample is computed by taking the average entropy over all of the dimensions.

A.3 Experimental Results

The above algorithm is implemented for 120 dimensional input vectors. A plot of entropy values for different dimensions is given in Fig. A.2.

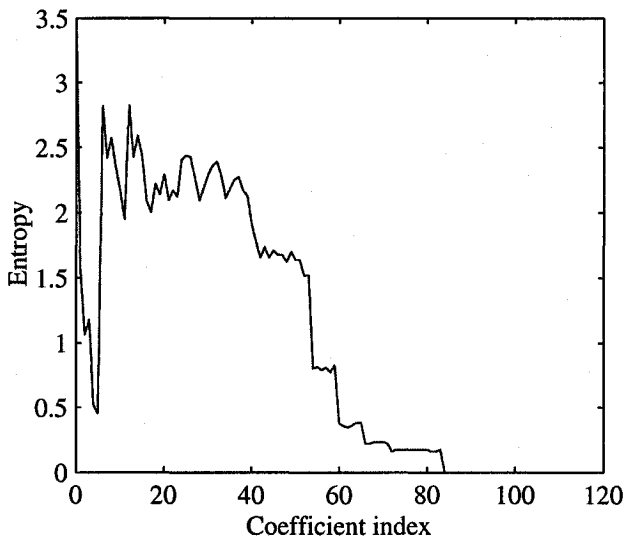

Fig. A.2 The entropy of each dimension for a 120 dimensional vector

This means the lower frequency bins need more number of bits to be quantized. Therefore, the average entropy has been used for the purpose of comparison. The entropy of 1 bit/sample is equal to a bit rate of 8 kbits/sec, since the sampling frequency is 8 kHz. Using MOS results, this entropy gives an acceptable quality of the quantized speech, but distortion can be felt to some extent.

Table A.1 MOS comparison for different number of subbands with the average entropy of 1 bit/sample

	Speech file
Number of Subbands	MOS
15	2.25
20	2.50
40	2.55

We have compared 3 different number of subbands in Table A.1. The results of MOS are obtained using PESQ software.

Table A.2 shows the performance of the quantizer for 20 subbands and different values of entropy. The test speech file belongs to a female speaker and its duration is 5 seconds. The test audio is a classical piano of duration 10 seconds.

The results of the two tables show that the quality of the quantized speech is acceptable. However, we can hear distortion.

Table A.2 MOS comparison for different entropies (20 subbands)

	Speech file	Audio file
Entropy (bit/sample)	MOS	MOS
0.5 (4 kHz)	1.98	2.10
0.75	2.15	2.25
1 (8 kHz)	2.50	2.91
1.2	2.60	3.50
2 (16 kHz)	2.80	4.00

References

[1] A. S. Spanias, "Speech coding: a tutorial review," *Proc. IEEE*, vol. 82, pp. 1541–1582, October 1994.

[2] T. Painter and A. Spanias, "Perceptual coding of digital audio," *Proc. IEEE*, vol. 88, pp. 451–513, April 2000.

[3] D. A. Reynolds and R. C. Rose, "Robust text independent speaker identification using gaussian mixture speaker models," *IEEE Trans. Speech, Audio Processing*, vol. 3, pp. 72–83, January 1995.

[4] A. D. Subramaniam and B. D. Rao, "PDF optimized parametric vector quantization of speech line spectral frequencies," *IEEE Trans. Speech, Audio Processing*, vol. 11, pp. 130–142, March 2003.

[5] B. Scharf, *Foundations of Modern Auditory Theory*, ch. Critical Bands. New York: Academic Press, 1970.

[6] T. Z. Shabestary, *Towards Low Complexity Vector Quantization.* PhD thesis, Chalmers University of Technology, Gothenburg, Sweden, May 2004.

[7] A. Gersho, "Advances in speech and audio compression," *Proc. IEEE*, vol. 82, pp. 900–918, June 1994.

[8] S. Furui and M. Sondhi, *Advances in Speech Signal Processing.* Marcel Dekker, Inc, February 1992.

[9] R. M. Gray, "Fundamental of data compression," *Invited Talk at the Int. Conf. on Inform. Comm., Signal Processing*, Singapore, September 1997.

[10] A. Gersho and R. M. Gray, *Vector Quantization and Signal Compression.* Kluwer, 1991.

[11] F. Norden, *Quantization, perception and speech coding.* PhD thesis, Chalmers University of Technology, Gothenburg, Sweden, May 2003.

[12] D. Ormoneit and V. Tresp, "Improved Gaussian mixture density estimates using Bayesian penalty terms and network averaging," Tech. Report FKI-205-95, TechnichalUniversity of Munich, 1995.

[13] J. D. Johnston, "Transform coding of audio signals using perceptual noise criteria," *IEEE Journal on Selected Areas in Comm.*, vol. 6, pp. 314–323, February 1988.

[14] C. R. Cave, "Perceptual Modelling for Low–Rate Audio Coding," Master's thesis, McGill University, Department of Electrical & Computer Engineering, June 2002.

[15] E. Zwicker and H. Fastl, *Psychoacoustics: Facts and Models.* Berlin, Germany: Springer-Verlag, 1990.

[16] E. Terhardt, "Calculating virtual pitch," *Hearing Research*, vol. 1, pp. 155–182, 1979.

[17] B. C. J. Moore and B. R. Glasberg, "Formulae describing frequency selectivity as a function of frequency and level, and their use in calculating excitation patterns," *Hearing Research*, vol. 28, no. 2-3, pp. 209–225, 1987.

[18] B. C. J. Moore and B. R. Glasberg, "Derivation of auditory filter shapes from notched-noise data," *Hearing Research*, vol. 47, no. 2-3, pp. 103–138, 1990.

[19] J. L. Hall, "Auditory psychoacoustics for coding applications," in *The Digital Signal Processing Handbook* (V. Madisetti and D. Williams, eds.), pp. 39.1–39.25, Boca Raton, FL: CRC Press, 1998.

[20] P. Kabal, "Time windows for linear prediction of speech," tech. rep., Department of Electrical and Computer Enginering, McGill University, 2003.

[21] J. D. Johnston, *Subband and Wavelet Transforms*, ch. Audio coding with filter banks, pp. 287–307. Kluwer Academic, 1996.

[22] H. Malvar, "Modulated QMF filter banks with perfect reconstruction," *Electron. Lett.*, vol. 26, pp. 906–907, June 1990.

[23] T. Ramstad, "Cosine modulated analysis-synthesis filter bank with critical sampling and perfect reconstruction," *Proc. IEEE Int. Conf. Acoustics, Speech, Signal Processing*, pp. 1789–1792, April 1991.

[24] P. P. Vaidyanathan, *Multirate Systems and Filter Banks.* Englewood Cliffs, N.J.: Prentice Hall, 1993.

[25] H. Malvar, "Lapped transforms for efficient transform/subband coding," *IEEE Trans. Speech, Audio Processing*, vol. 38, pp. 969–978, June 1990.

[26] H. Najafzadeh-Azghandi, *Perceptual Coding of Narrowband Audio Signals.* PhD thesis, Department of Electrical & Computer Engineering, McGill University, April 2000.

[27] R. Der, P. Kabal, and W.-Y. Chan, "Bit allocation algorithms for frequency and time spread perceptual coding," *Proc. IEEE Int. Conf. Acoustics, Speech, Signal Processing*, pp. IV-201– IV-204, May 2004.

[28] International Telecommunication UnionITU-R, Geneva, *Recommendation BS.1387-1,Method for objective measurements of perceived audio quality*, November 2001.

[29] T. Thiede, *Perceptual Audio Quality Assesment Using a Nonlinear Filterbank.* PhD thesis, Fachbereich Electrotechnik, Technical University of Berlin, 1999.

[30] E. Terhardt, "Calculating virtual pitch," *Hearing Research*, vol. 1, pp. 152–182, March 1979.

[31] P. Hedelin and J. Skoglund, "Vector quantization based on Gaussian mixture models," *IEEE Trans. Speech, Audio Processing*, vol. 8, pp. 385–401, July 2000.

[32] R. M. Gray, "Gauss mixture vector quantization," *Proc. IEEE Int. Conf. Acoustics, Speech, Signal Processing*, vol. 3, pp. 1769–1772, May 2001.

[33] D. W. Scott, *Multivariate Density Estimation: Theory Practice and Visualization.* New York: Wiley, 1992.

[34] G. McLachlan and K. E. Basford, *Mixture Models: Interface and Application to Clustering.* New York: Marcel Dekker, 1988.

[35] R. O. Duda, P. E. Hart, and D. G. Stork, *Pattern Classification*, ch. 2. New York: Wiley, 2001.

[36] A. Dempester, N. Laird, and D. Rubin, "Maximum likelihood from incomplete data via the EM algorithm," *J. R. Static.,Soc.*, vol. 39, pp. 1–38, 1977.

[37] J. Bilmes, "A gentle tutorial of the EM algorithm and its application for Gaussian mixture and hidden Markov models," Tech. Report TR 97-021, Computer Science Div., Univ. California, Berkeley, April 1998.

[38] P. McKenzie and M. Adler, "The EM algorithm for Gaussian mixture modelling and its intialization," Tech. Report, Univ. West. Austrailia, 1993.

[39] R. Bellman, *Adaptive Control Processes.* Princeton NJ: Princeton Univ. Press, 1961.

[40] N. Kambhatla, *Local models and Gaussian mixture models for statistical data processing.* Phd thesis, Oregon Graduate School of Science and Technology, January 1996.

[41] M. Jordan and R. Jacobs, "Hierarchical mixture of experts and the EM algorithm," *Neural Computation*, vol. 6, pp. 181–214, March 1994.

[42] B. Atal, "Automatic recognition of speakers from their voices," *Proc. IEEE*, vol. 64, pp. 460–475, April 1976.

[43] J. K. Su and R. M. Mersereau, "Coding using Gaussian mixture and generalized Gaussian models," *IEEE Int. Conf. Image processing*, pp. 217–220, September 1996.

[44] A. V. Oppenheim and A. S. Wilsky, *Signals and Systems.* Signal processing, Prentice Hall, 2nd ed., 1996.

[45] K. Fukunaga, *Introduction to Statistical Pattern Recognition.* New York: Academic Press, Inc., 1972.

[46] J. Huang and Y. Zhao, "A DCT based fast signal subspace technique for robust speech recognition," *IEEE Trans. Speech, Audio Processing*, vol. 8, pp. 747–751, November 2000.

[47] C. L. Lawson and R. J. Hanson, *Solving Least Squares Problems.* Prentice-Hall, 1974.

[48] J. J. Y. Huang and P. M. Schultheiss, "Block quantization of correlated Gaussian random variables," *IEEE Trans. Commun. Syst.*, vol. 11, pp. 289–296, September 1963.

[49] R. Zelinski and P. Noll, "Adaptive transform coding of speech signals," *IEEE Trans. Speech, Audio Processing*, vol. ASSP-25, pp. 299–309, August 1977.

[50] R. V. Cox and R. E. Crochiere, "Real time simulation of adaptive transform coding," *IEEE Trans. Speech, Audio Processing*, vol. ASSP-29, pp. 147–154, April 1981.

[51] P. F. Panter and W. Dite, "Quantization distortion in pulse count modulation with nonuniform spacing of levels," *Proc. IRE*, vol. 29, pp. 44–48, January 1951.

[52] P. W. Moo and D. L. Neuhoff, "Optimal compressor functions for multidimensional companding," *IEEE Int. Symp. Inform. Theory*, p. 515, June 1997.

[53] ITU-T standard, *P.862, Perceptual evaluation of speech quality (PESQ)*, February 2001.

[54] H. Pobloth and W. B. Kleijn, "On phase perception in speech," *ICASSP*, vol. 1, pp. 29–32, March 1999.

[55] D. S. Kim, "Perceptual phase quantization of speech," *IEEE Trans. Speech, Audio Processing*, vol. 11, pp. 355–364, July 2003.

[56] A. V. Oppenheim and R. W. Shafer, *Discrete-Time Signal Processing.* Englewood Cliffs, NJ: Prentice Hall, Inc., 1989.

[57] R. G. Gallager, *Information Theory and Reliable Communication.* John Wiley and Sons, 1968.