DIGITAL ENCODING OF SPEECH SIGNALS
AT 16 - 4.8 KBPS

Thesis submitted to the University of Surrey
for
the degree of Doctor of Philosophy

Ahmet M. Kondoz
Department of Electronic and Electrical Engineering
University of Surrey
Guildford, Surrey.
UK.

January 1988
SUMMARY

Speech coding at 64 and 32 Kb/s is well developed and standardized. The next bit rate of interest is at 16 Kb/s. Although, standardization has yet to be made, speech coding at 16 Kb/s is fairly well developed. The existing coders can produce good quality speech at rates as low as about 9.6 Kb/s. At present the major research area is at 8 to 4.8 Kb/s.

This work deals first of all with enhancing the quality and complexity of some of the most promising coders at 16 to 9.6 Kb/s as well as proposing new alternative coders. For this purpose coders operating at 16 Kb/s and 12 to 9.6 Kb/s have been grouped together and optimized for their corresponding bit rates. The second part of the work deals with the possibilities of coding the speech signals at lower rates than 9.6 Kb/s. Therefore, coders which produce good quality speech at bit rates 8 to 4.8 Kb/s have been designed and simulated.

As well as designing coders to operate at rates below 32 Kb/s, it is very important to test them. Coders operating at 32 Kb/s and above contain only quantization noise and usually have large signal to noise ratios (SNR). For this reason their SNR's may be used for comparison of the coders. However, for the coders operating at 16 Kb/s and below this is not so and hence subjective testing is necessary for true comparison of the coders. The final part of this work deals with the subjective testing of 6 coders, three at 16 Kb/s and the other three at 9.6 Kb/s.
I would like to express my thanks and gratitude to my supervisor Professor B.G. Evans for the guidance, help and encouragement he provided during this work.

I would like to thank the staff of the subjective testing division in British Telecom Research Labs, for kindly providing the IRS equipment for the subjective tests.

To my mother Fatma, my wife Munuse and my son Mustafa I present my thanks for their encouragement, support and love.
CONTENTS

CHAPTER 1 - INTRODUCTION

CHAPTER 2 - DIGITAL SPEECH CODING AND ITS APPLICATIONS

2.1 Introduction 4
2.2 Digital coding of speech 4
2.3 Applications of digital speech coding 7
   2.3.1 Satellite applications 9
   2.3.2 Public Switch Telephone Network (PSTN) 10
2.4 References 11

CHAPTER 3 - FREQUENCY DOMAIN SPEECH CODING

3.1 Basic system concepts 12
3.2 Sub-band coding 14
   3.2.1 Band splitting 14
   3.2.2 Encoding the sub-band signals 27
3.3 Adaptive Transform Coding (ATC) 30
   3.3.1 The block transformation 31
   3.3.2 Quantization of the transform coefficients 33
   3.3.3 Bit allocation 34
   3.3.4 Noise shaping 34
   3.3.5 Adaptation strategy 35
3.4 References 40

CHAPTER 4 - TIME DOMAIN SPEECH CODING

4.1 Basic system concepts 42
   4.1.1 Linear Predictive Coding (LPC) of speech 42
4.1.2 Pitch predictive coding of speech 44
4.2 Adaptive Predictive Coding (APC) 46
4.3 Base-Band coding 49
4.4 Multi-Pulse Excited Linear Predictive Coder (MPLPC) 52
4.5 Code Excited Linear Prediction (CELP) 55
4.6 Harmonic scaling of speech 57
4.7 References 61

CHAPTER 5 - VECTOR QUANTIZATION OF SPEECH SIGNAL
5.1 Basic system concepts 64
5.1.1 Distortion measures 67
5.1.2 Code-book design 69
5.1.3 Computational and storage costs 70
5.2 Code-book design and search 71
5.2.1 Binary search 71
5.2.2 Cascaded quantization 74
5.2.3 Random code-books 74
5.2.4 Training testing and code-book robustness 75
5.3 References 76

CHAPTER 6 - 16 KB/S CODERS
6.1 Introduction 79
6.2 16 Kb/s sub-band coder 79
6.2.1 Band splitting 79
6.2.2 Encoding the sub-bands 83
6.2.3 Bit allocation and noise shaping 87
6.2.4 Simulations 89
6.2.5 Further considerations on bit allocation and quantization 102
6.3 16 Kb/s Transform coder 105
6.3.1 Simulations 106
6.4 Discussions 113
6.5 References 114

CHAPTER 7 - 12KB/S TO 9.6 KB/S CODERS

7.1 Introduction 116
7.2 Sub-Band coder 116
  7.2.1 SBC with vector quantized side information 119
  7.2.2 Fully vector quantized SBC 129
7.3 Transform Coder 141
  7.3.1 Zelinsky and Noll's approach 142
  7.3.2 Vocoder driven ATC 145
  7.3.3 Hybrid Transform Coder 150
7.4 LPC of speech with VQ and frequency domain noise shaping 159
  7.4.1 Coder description 161
  7.4.2 Simulations 164
  7.4.3 Discussions 168
7.5 Linear Predictive BBC and High Frequency Regeneration of speech 169
  7.5.1 Coder description 170
  7.5.2 Discussions 172
7.6 Discussions 174
7.7 References 175

CHAPTER 8 - 8 KB/S TO 4.8 KB/S CODERS

8.1 Introduction 176
8.2 Code Excited Linear Prediction (CELP) 177
  8.2.1 8000 bits/sec CELP 178
  8.2.2 4800 bits/sec CELP 183
  8.2.3 Complexity consideration of CELP 194
10.3 Future work

10.4 References

APPENDICES

A  Parallel filter coefficients for a 16 band SBC  251
B  Parallel filter coefficients for a 16 band SBC with two point FFT  265
C  List of published papers  267
D  Source code (in C) of important algorithms  289
CHAPTER 1

INTRODUCTION

When human beings converse, they do so via sound waves. These sound waves cannot travel more than 100 to 200 meters without disturbing others and losing privacy. Also, over larger distances, the human voice transmitted in free space becomes inadequate and acoustical amplification of the speech would generally be unacceptable in our modern society. Even if shouting was acceptable, practical limitations would not allow it, i.e., when everybody talks loudly nobody understands anything. As a result, to communicate over long distances we must resort to electrical techniques, with the use of acousto-electrical and electro-acoustical transducers. Before transmission speech is coded into an analogue or digital format. In the past analogue representation of speech has been widely used. Although, digital coding of speech was proposed more than three decades ago, its realization and the exploitation for the benefit of society has taken place within the last 5 to 10 years. Since then there has been a great emphasis on producing completely digital speech networks. There are a number of reasons for digital coding of speech signals.

Transmission of speech over long distances requires repeaters and amplifiers. In analogue transmission, noise cannot be eliminated when amplification is employed. Therefore, long distances mean greater noise accumulation. Digital coding achieves transmission of information over long distances without degradation of speech quality. This occurs because digital signals are regenerated, i.e., retimed and reshaped at the repeaters. The transmission quality therefore, is almost independent of distance and network topology in an all digital environment.

In comparison with the frequency division multiplexing (FDM) techniques in analogue transmission systems, where complex filters are required, the multiplexing function in digital systems is and can be achieved with economic digital circuitry. Furthermore, switching of digital information is easily performed with digital building blocks leading to all-electronic exchanges which obviate the problems of analogue cross-talk and mechanical switching.

Interconnection of various transmission media and switching equipment is realized by relatively cheap interface equipment with little or no signal impairment. Also by
multiplexing digital signals (TDM), the channel capacity in an existing media may be increased.

Using a uniform digital format digital signals can be transmitted over the same communication system. Consequently, speech signals can be handled together with other signals such as video, computer data, facsimile etc.

Nowadays complex signal processing can easily be achieved by digital computers. Digital signals can easily be encrypted to provide secrecy in secure communication channels such as the military. The power requirements for digital systems transmission is much less than analogue systems and also in digital systems transmission reliability is much higher. These factors have extra importance in satellite and computer controlled communications.

Digital transmission is more robust to noise in the transmission path. Using forward error correction (FEC) [1], digital systems can extract the information even in the presence of noise which is higher than the signal level. Adaptive digital processing methods based on the signal statistics [2] can also be applied to recover signals in severe conditions. These cannot be achieved in real time without the use of large scale integration techniques (LSI). LSI employed in the realization of digital circuits can result in cheap and very compact equipment. As a final application, digitization of speech offers the possibility of voice communication with computers.

Although, digitization of speech is necessary for speech recognition processing as well as for transmission, we are here only interested in the coding of speech signals for transmission purposes. Digitization of speech for transmission over a communication channel has one very significant disadvantage. Digital speech transmission requires very much larger transmission bandwidth, in order to maintain the quality of a 4 KHz analogue speech channel. Unless the bandwidth of the digital speech transmission is reduced whilst maintaining its analogue equivalent quality, the advantages of digital speech coding, listed above will not be fully exploited and may be very costly. Spectral efficiency is extremely important in many radio communication systems, e.g. mobile satellite and cellular systems. However, for digital transmission reducing the bandwidth could mean the reduction of the number of bits to be used to code the speech samples, and hence, a reduction in speech quality. High digital speech quality can be obtained at 64 Kb/s and 32 Kb/s by PCM [3] and ADPCM [4][5] respectively, but the required transmission bandwidth is still too much greater to be practical for use in satellite cellular communication systems. It is therefore, very important to reduce the bit rate of
coded speech down to 16 Kb/s and below if digital speech is to be introduced economically to the communication systems. There are two other important parameters that should be taken into consideration for digital speech coding. These are the coding delay and the cost. The major factors: high quality, reduced bit rate, small delay and low cost are all in opposition to each other. For high quality and low bit rates may be achieved with long coding delays and high cost. During the course of this research work we have investigated various methods of reducing the speech bit rate whilst maintaining high quality, low delay and cost. The research work was split into three major areas. speech coding at 16 Kb/s, 12 to 9.6 Kb/s and 8 to 4.8 Kb/s, which are discussed in chapters 6, 7 and 8 respectively. In chapter 2 we briefly discuss various speech coding schemes and applications. In chapter 3, 4, and 5 basic principles of the most promising low bit rate speech coding algorithms are discussed. Finally, in chapter 9 we present the results of a small subjective test, and to conclude in chapter 10 we discuss the major conclusions obtained from the work and suggest possible future areas.

References


CHAPTER 2

DIGITAL SPEECH CODING AND ITS APPLICATIONS

2.1 Introduction

Here, we briefly discuss digital coding of speech signals and its applications.

2.2 Digital Coding Of Speech

Digital coding of speech signals can be broadly classified into three categories, namely: Analysis - synthesis (vocoder) coding, waveform coding and hybrid coding as shown in Figure 2.1. The concepts used in the first two methods are very different, and the third method is a mixture of the first two coding systems.

In the vocoding systems, only the theoretical model of the speech production mechanism is considered and its parameters are derived from the actual speech signal and coded for transmission. At the receiver these model parameters are decoded and used to control a speech synthesizer which corresponds to the model assumed in the analyser. Provided that the perceptually significant parameters of the speech are extracted and transmitted, the synthesized signal perceived by the human ear approximately resembles the original speech signal. Therefore, during the analysis procedure the speech is reduced to its essential features and all of the redundancies are removed. Consequently, a great saving in transmission bandwidth is achieved. However, when compared with the waveform coding methods, analysis - synthesis processing operations are complex, resulting in expensive equipment.

In waveform coding systems, an attempt is made to preserve the waveform of the original speech signal. In such a coding system the speech waveform is sampled and each sample is coded and transmitted. At the receiver the speech signal is reproduced from the decoded samples. The way in which the input samples are coded at the transmitter may depend upon the previous samples or parameters derived from the previous samples, so that advantage of the speech waveform characteristics can be taken. Waveform coding systems tend to be much simpler and therefore inexpensive compared to the vocoder type systems. Because of this, they are of considerable interest and importance and their
applications may vary from mobile radio to commercial line systems.

Hybrid coding of speech, as the name suggests, combines the principles of both vocoders and waveform coders. Using suitable modelling, redundancies in speech are removed leaving a small energy residual signal to be coded by a waveform coder. Therefore, the difference between a pure waveform coder and a hybrid coder is that in the hybrid coder, the energy in the signal to be coded is minimized before quantization, hence, the quantization error which is proportional to the energy in the input signal is reduced. On the other hand the difference between a vocoder and a hybrid coder is that in a hybrid coder the excitation signal is transmitted to the decoder, however, in a vocoder a theoretical excitation source is used. Therefore, hybrid coders try to bridge the gap between high quality waveform coders and synthetic quality vocoders.

<table>
<thead>
<tr>
<th>Speech Coding</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Waveform Coding</td>
<td>Hybrid Coding</td>
<td>Vocoding</td>
</tr>
<tr>
<td>PCM</td>
<td>APC</td>
<td>CV</td>
</tr>
<tr>
<td>APCM</td>
<td>SBC</td>
<td>FV</td>
</tr>
<tr>
<td>DPCM</td>
<td>ATC</td>
<td>LPC</td>
</tr>
<tr>
<td>ADPCM</td>
<td>RELP</td>
<td></td>
</tr>
<tr>
<td>ADPCM</td>
<td>VELP</td>
<td></td>
</tr>
<tr>
<td>TDHS</td>
<td>MPLPC</td>
<td></td>
</tr>
<tr>
<td>CELP</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Figure 2.1: A broad classifications of speech coders.

Hybrid coders may use various speech specific principles to reduce the speech residual energy before quantization. Therefore, hybrid coders can be further classified according to modelling principles as shown in Figure 2.2.
Modelling of short term amplitude spectrum using vocoding techniques

<table>
<thead>
<tr>
<th>Hybrid Coding</th>
<th>Pre-processing and waveform coding</th>
<th>Residual excited vocoders</th>
<th>Definition of excitation sequence using analysis by synthesis</th>
</tr>
</thead>
<tbody>
<tr>
<td>VDATC HC</td>
<td>TDHS-ADPCM</td>
<td>RELP</td>
<td>MPLPC</td>
</tr>
<tr>
<td></td>
<td>TDHS-ATC</td>
<td>VELP</td>
<td></td>
</tr>
<tr>
<td></td>
<td>TDHS-SBC</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>TDHS-APC</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Figure 2.2: Principles classification of hybrid coders.

The coders listed under the headings of waveform coding, hybrid coding and vocoding in Figure 2.1 operate at various bit rates. However, assuming an average range of operation for each class, we can represent their quality against bit rate performance as shown in Figure 2.3.

![Diagram](image)

Figure 2.3: Speech quality versus bit rate for different types of coders.
Similar plots to those in Figure 2.3 may be drawn to represent the complexity of waveform coders and vocoders. However, it is extremely difficult to represent the complexities of hybrid coders on a single scale, because the relative complexity of the coders (e.g., RELP and CELP) are very different. However, one can say that hybrid coders are the most complex of all. Some hybrid coders such as CELP cannot be implemented without some simplifications.

From Figure 2.3 it can be seen that no matter what the bit rate is, the quality of recovered speech for vocoding techniques cannot reach 'good' or 'excellent' quality. They have 'poor' to 'fair' quality. Waveform coders on the other hand have 'excellent' quality at bit rates of 32 Kb/s and above. However, their speech quality deteriorates rapidly below about 24 Kb/s. Therefore, hybrid coders have their best operation range from 4 Kb/s to 16 Kb/s. In the following three chapters we explain the principles of the most promising hybrid coding techniques under the headings of frequency domain speech coding, time domain speech coding and vector quantization.

2.3 Applications Of Digital Speech Coding

Digital speech coding is rapidly becoming an attractive and viable technology for communications and man-machine interaction. This technology is being encouraged by advances in several fields. New algorithms are being developed for efficiently coded speech signals in digital form at reduced bit rates by taking advantage of the properties of speech production and perception. Simultaneously, device technology is evolving to a point where substantial amounts of real-time digital signal processing and digital data handling can be performed within single integrated circuits. Finally, new systems concepts in digital communications, computing, and switching are evolving which offer more flexible opportunities for storage and transfer of digital information.

There are various applications of digital speech coding which require system specific parameters and complexity requirements. These may be listed as follows:

- Delay
- Complexity
- Quality
- Compatibility with the existing systems
- Performance in specific channel conditions
Data handling

Delay

Delay is very important in real-time telephone systems. The importance of delay becomes more pointed for satellite applications where already large delays exist because of the long distance propagation. However, in some non-real-time applications and computer to computer message transmission and in some one way store and forward systems delay may not be so important.

Delay in digital coding schemes is introduced due to two reasons. One is that if the algorithm is complex, delay is necessary for the computation of the major complexity blocks. The other reason for delay is the theoretical algorithmic delay which is necessary for speech specific parameter calculations.

Complexity

The complexity and hence the cost of speech coding systems is extremely important if it to be widely used. For this purpose the cost of the terminal equipment should be kept as low as possible.

Quality

Most important of all is the quality of the received or recovered speech. Under all circumstances the quality of recovered speech should be kept at a level which will be acceptable by customers. The major speech quality degradations are introduced during the digital coding process of the analogue speech signals. Therefore, the chosen speech coding algorithm should maintain the quality of speech at an acceptable level.

Compatibility With Existing Systems

Any new digital speech coding system should be easily integrated into the existing network without causing extra delay, reduced performance or additional cost.

Performance Under Specific Channel Conditions

The quality of the recovered speech may be affected by the various channel conditions. This is especially important in various satellite applications. Therefore, speech coding techniques should either be robust under channel errors or allow some of the channel
capacity to be used for forward error detection and correction.

Data Handling

Some applications may require the transmission of data using the speech channel. Therefore, for certain applications speech coding systems should handle data as well as speech.

2.3.1 Satellite Applications

The choice of the speech coding technique is one of the most important technologies for the development of low carrier to noise (C/N) ratio digital radio satellite communication systems for land, maritime and aeronautical mobile communications and also for thin-route communications. A comprehensive study quantifying the subjective performance of various encoding techniques in a telephone network environment was reported in reference [1]. Also as intensive study on various candidate speech coding techniques was conducted to choose the most suitable coding techniques for use in satellite communications [2].

In low C/N digital satellite communication systems, speech coding at a low bit rate up to 16 Kb/s is attractive to economically meet the growing demand for telephone service and also to effectively provide ISDN services by speech and data integration.

The international maritime satellite organization (INMARSAT) has a concrete plan to introduce a new digital maritime satellite communication system in which the telephone channel is digitized at 16 Kb/s instead of the companded FM currently in use. The 16 Kb/s digital channel provides increased availability maritime channel capacity, savings of limited satellite power, and also provides capability to offer a wide variety of new services. Adaptive predictive coding with maximum likelihood quantization (APC-MLQ) [3] has been chosen for use in the INMARSAT system. The APC has a new adaptive quantizer in which the step sizes are controlled to minimize the power of the difference between an input signal and the reconstructed signal. Performance indicates that the APC-MLQ is one of the most suitable low rate speech coding techniques for the low C/N satellite communication systems at 16 Kb/s [3][4].

INMARSAT plans to introduce a new digital maritime satellite communication system called the 'standard-B system' adopting 16 Kb/s speech coding. In low C/N satellite communication systems including thin-route systems, companded FM has generally been
used for public telephone services. In the smooth transition from the existing analogue system to the new digital system, the main performance requirements for the 16 Kb/s speech coding are [4].

a) Subjective speech quality comparable to or better than that of companded FM in the existing analogue system.

b) Robustness to bit errors in a range of $10^{-3}$ and $10^{-2}$ error rates.

c) Transparency of voice-band data up to 2400 bits/sec.

d) Immunity to ambient noise.

A recent speech coding activity has been the common European mobile telephony standardization. Amongst the major coding candidates there were four sub-band coders, one multi-pulse LPC and a regular pulse excited LPC which were submitted by Norway, Sweden, Italy, France and Germany respectively. Although, final test results have not been published regular pulse excited LPC combined with the pitch filter used in the French multi-pulse LPC (RPE-LTP) has been selected. RPE-LTP is a new approach to multi-pulse coding [5] which produce high quality speech at around 13 Kb/s, allowing some capacity for FEC in a 16 Kb/s channel. RPE-LTP is a base-band type coder which uses a weighting filter and grid selector to approximate the decimated sequence to the optimized multi-pulse sequence.

2.3.2 Public Switch Telephone Network (PSTN)

For the PSTN applications the transmission power (bandwidth) is not as critical as it is in the satellite applications. However, still great savings can be made if the reduced bit rate speech coding techniques are used. The standard channel is designed for 64 Kb/s (PCM) but if the bit rate is reduced by a factor of 2 or more then 2 or more sub-channels could be multiplexed in to the standard 64 Kb/s. By digitizing PSTN the following advantages can be gained.

(i) Digital speech signals can be regenerated at stations along the transmission path, hence transmission can be achieved over long distances with immunity to cross talk and random noise.

(ii) Easy signalling, multiplexing, switching and improved end to end quality.

(iii) Flexible processing, echo cancellation, equalization and filtering and other processing such as encryption.
At present there are two standardized digital speech coding algorithms. First one is the Pulse Code Modulation (PCM), A or μ law, which was standardized in 1972. The second is the Adaptive Differential Pulse Code Modulation (ADPCM), which was standardized in 1985 to operate at 32 Kb/s for speech and voice-band data.

Since the standardization of ADPCM at 32 Kb/s in 1985, there have been many high quality lower bit rate speech coding algorithms developed (SBC, APC, ATC, RELP). However, officially none of these high quality lower bit rates has been standardized. Amongst these high quality low bit rate speech coders two have been adopted by INMARSAT and GSM (APC-MLQ and RPE-LTP at 16 Kb/s respectively).

Although, there is no other standard algorithm for commercial use, there is a military standard. LPC-10 has been used by the military at 2.4 Kb/s which is a vocoder and produces synthetic quality speech.

2.4 References

3.1 Basic System Concepts.

The basic concept in frequency domain coding is to divide the speech spectrum into frequency bands or components using either a filter bank or a block transform analysis. After encoding and decoding, these frequency components are used to resynthesize a replica of the input waveform by either filter bank summation or inverse transform means. A primary assumption in frequency domain coding is that the signal to be coded is slowly time varying which can be locally modelled with a short-time spectrum. Also, for most applications involving real-time constraints, only a short time segment of input signal is available at any given time instant. Within the context of the above explanations, a block of speech can be represented by a filter bank or block transformation as follows.

(i) In the filter bank interpretation $\omega$ is fixed at $\omega = \omega_0$ and $X_n(e^{j\omega_0})$ is viewed as the output of a linear time invariant filter with impulse response $h(n)$ excited by the modulated signal $x(n) e^{-j\omega_0 n}$.

$$X_n(e^{j\omega_0}) = h(n) * [x(n) e^{-j\omega_0 n}]$$  \hfill (3.1)

Here $h(n)$ determines the bandwidth of the analysis around the centre frequency $\omega_0$ of the signal $x(n)$ and is referred to as the analysis filter [1][2][3][4].

(ii) In the block Fourier transform interpretation the time index $n$ is fixed at $n = n_0$ and $X_{n_0}(e^{j\omega})$ is viewed as the normal Fourier transform of the windowed sequence $h(n_0-m) x(m)$.

$$X_{n_0}(e^{j\omega}) = F[h(n_0-m) x(m)]$$  \hfill (3.2)
where $F[]$ denotes the Fourier transform. Here, $h(n_0-n)$ determines the time width of the analysis around the time instant $n=n_0$ and is referred to as the analysis window [1][2][3][4].

Portnoff [5] shows that the synthesis equation for the filter bank or the block transformations are as follows. For the filter bank synthesis,

$$\hat{x}(n) = \frac{1}{2\pi h(0)} \int_{-\pi}^{\pi} X_n(e^{j\omega}) e^{j\omega n} d\omega$$  \hspace{1cm} (3.3)

which can be interpreted as the integral (or incremental sum) of short time spectral components $X_n(e^{j\omega n})$ modulated back to their centre frequencies $\omega_0$.

For the block transformation synthesis, synthesis equation takes the form,

$$\hat{x}(n) = \frac{1}{H(e^{j\theta})} \sum_{r=-\infty}^{\infty} F^{-1}[X_r(e^{j\omega})]$$  \hspace{1cm} (3.4)

which can be interpreted as summing the inverse Fourier transformed blocks corresponding to the time signals $h(r-n)x(n)$.

Although, the theory shown above may appear too complex to be implemented in real time, recent advances in digital technology make economic implementation possible. The two well known speech coding techniques which belong to the class of frequency domain coders are Sub-Band coding (SBC) [6][7][8], and Adaptive Transform coding (ATC) [9][10][11]. The basic principles in both schemes are the division of the input speech spectrum into a number of frequency bands which are then separately encoded. Separate encoding offers two advantages. Firstly, the quantization noise can be contained within bands, and prevented from creating out-of-band harmonic distortion. Secondly, the number of bits allocated for coding of each band can be optimized to obtain the best overall performance.

In SBC a filter bank is employed to split the input speech signal typically into 4 to 16 broad frequency bands (wide band analysis). In ATC on the other hand a block transformation method with a typical transform size of 128 to 256 is used to provide much finer frequency resolution (narrow band analysis). In the following sections these two main frequency domain coding techniques will be discussed in greater details.
3.2 Sub-Band Coding

Sub-band coding is a waveform coding method which uses the wide band short time analysis/synthesis. The speech spectrum is partitioned into a number of bands and each band is low-pass translated to zero frequency. The resulting signals in each band are then sampled at the Nyquist rate, encoded, multiplexed and transmitted. At the receiver, the sub-bands are de-multiplexed, decoded and translated back to their original positions. The resulting sub-band signals are then summed together to give an approximation of the original speech signal.

The partitioning of the speech spectrum into bands and the coding of the signals related to these bands has a number of advantages when compared to single full band coding methods. In particular, by encoding the sub-bands, the short-time formant structure of the speech spectrum can be exploited. In this way the number of quantization levels can vary independently from one band to another as well as the characteristics of the quantizers. Also the quantization noise in a given band is confined to that band and there is no spill over into the adjacent frequency ranges. In addition, when employing a fixed or an adaptive bit allocation scheme to operate as part of the coding strategy, the spectrum of the noise found in the reconstructed signal can also be shaped in a perceptually advantageous way.

In practice the sub-band signals are produced in a slightly different way than that discussed above in terms of the short time Fourier transform. In order to produce real sub-band signals as opposed to the complex signals (using Fourier transforms), the speech spectrum can be split into a desired number of bands using several techniques. There are four techniques which have been used. These are Integer Band Sampling (IBS), Tree structure Quadrature Mirror Filters (TQMF), Discrete Cosine Transform (DCT), and Parallel Filter Banks (PFB).

3.2.1 Band Splitting

3.2.1.1 Integer Band Sampling (IBS).

Crochiere, one of the pioneers of sub-band coding, proposed an IBS technique for performing the low-pass to band-pass translations which eliminates the need for modulators and is therefore easily realized in hardware [7]. This is illustrated in Figure 3.1. The speech band is partitioned into $b$ sub-bands by band-pass filters $BP_i$ to $BP_b$. The output of each filter in the transmitter is re-sampled at a rate of $2f_i$, where $f_i$ is the
bandwidth of the \( i^{th} \) sub-band. These decimated signals are then digitally encoded and multiplexed for transmission. At the receiver, the decoded sub-band signals are up-sampled to their original sampling rates by inserting zero valued samples. These signals are then filtered by another set of band-pass filters, identical to those at the transmitter. Finally, the outputs of these filters are summed to give a reconstructed replica of the original input signal.

As shown in Figure 3.1, the IBS method imposes certain constraints on the choice of sub-bands. Sub-bands are required to have a frequency range between \( m_1 f_s \) and \( m_1 f_s \), where \( m_1 \) is an integer to avoid aliasing in the sampling process.

\[ S_i(\omega) \]

As shown in Figure 3.1, the IBS method imposes certain constraints on the choice of sub-bands. Sub-bands are required to have a frequency range between \( m_1 f_s \) and \( m_1 f_s \), where \( m_1 \) is an integer to avoid aliasing in the sampling process.

3.2.1.2 Tree Structure Quadrature Mirror Filter (TQMF)

Although the integer band sampling method has produced encouraging results, very long filters (175–200 taps) are necessary to provide the sharp cut-off characteristics.
required in order to reduce aliasing or inter-band leakage arising from the sampling process. A more elegant design [12][13], allows for almost perfect cancellation of this aliasing effect by utilising a set of low and high-pass filters which possess quadrature relationship. Consider the design of a 2 (equal) band sub-band coder, which uses a low-pass and a high-pass filters to split the signal, as shown in Figure 3.2.

![Diagram](image)

Figure 3.2: A 2 band sub-band coder, (a) coder structure, (b) spectral description of sub-bands.

The down sampling process in both upper and lower bands introduce aliasing terms in each of the sub-band signals. In the lower band, the signal frequency above $f_s/4$ is folded down into the range 0 to $f_s/4$ and appears as aliasing in this signal as shown by the shaded area in Figure 3.2b. Similarly, for the upper band any signal energy below
\( f_s/4 \) is folded upward into its Nyquist band \( f_s/4 \) to \( f_s/2 \). The amount of aliasing of energy or inter-band leakage is directly dependent on the degree to which the filters \( h_1(n) \) and \( h_2(n) \) approximate ideal low-pass and high-pass filters respectively.

In the re-construction process, the sub-band sampling rates are increased by inserting zeros between each sub-band sample. This introduces a periodic repetition of the signal spectra in the sub-bands. For example, in the lower band the signal energy from 0 to \( f_s/4 \) is symmetrically folded around \( f_s/4 \) into the range of the upper band. This unwanted signal energy or image is filtered out by the low-pass filter \( h_1(n) \) at the receiver. The filtering operation effectively interpolates the zero valued samples that have been inserted between the sub-band signals. In the same way the image from the upper band is reflected to the lower sub-band and filtered out by the filter \(-h_2(n)\).

Because of the quadrature relationship of the sub-band signals in the QMF, the remaining components of the images can be exactly cancelled by the aliasing terms introduced in the analysis (in the absence of transmission errors and quantization noise). In practice, this cancellation is obtained down to the level of quantization noise of the coders.

To obtain this cancellation property in the QMF, the filters \( h_1(n) \) and \( h_2(n) \) must be symmetrical filter designs.

\[
\begin{align*}
  h_1(n) &= h_2(n) = 0 \quad \text{for } n < 0 \text{ and } n \geq T \\
  \text{where } T & \text{ is the number of taps in the filters. The symmetrical property implies that,}
\end{align*}
\]

\[
\begin{align*}
  h_1(n) &= h_1(T-1-n) \\
  \text{and}
  h_2(n) &= -h_2(T-1-n) \quad \text{for } n = 0, 1, ..., (T/2)-1
\end{align*}
\]

The QMF further requires that the filters satisfy the condition.
\[ h_2(n) = (-1)^n h_1(n) \quad \text{for } n = 0, \ldots, T-1 \] (3.8)

which shows the mirror image relationship of the filters. The filters must also satisfy the condition.

\[ |H_1(e^{j\omega})|^2 + |H_2(e^{j\omega})|^2 = 1 \] (3.9)

where \( H_1(e^{j\omega}) \) and \( H_2(e^{j\omega}) \) denote the Fourier transforms of \( h_1(n) \) and \( h_2(n) \) respectively. Figure 3.3 shows the frequency response for a 32 tap filter design obtained by Johnston (32D design), [12].

![Figure 3.3: Frequency Response of a 32-tap Quadrature Mirror Filter Design for a Two-band Sub-band Coder](image)

(a) Magnitude Response of Individual High and Low pass Filters

(b) Magnitude Response of the Composite System
For band splitting into more than two bands, the basic QMF can be repeated in a tree structure. Figure 3.4 shows the use of QMF in an 8 band sub-band coder.

Figure 3.4: Tree structure QMF in an 8 equal band SBC.
Sub-band coders with nonuniform bands may also be obtained using the QMF approach subject to some limitations. This is done by truncating certain sections of the tree as shown in Figure 3.5, for a 5 band sub-band coder.

![Diagram of 5 band sub-band coder with non-uniform spacing of bands.](image)

Figure 3.5: 5 band sub-band coder with non-uniform spacing of bands.

The use of symmetrical FIR filters in the TQMF introduces a delay in the system equal to \((T-1)/2\) samples at each stage. i.e. \(f_s = 8\) KHz, \(T = 32\), delay = \((32-1) / 2\) = 15.5 samples and delay in time = 15.5 / 8000 ≈ 2 milliseconds. However, because the sampling rate of the sub-bands is halved at each stage, the actual amount of delay (referred to the original sampling rate) increases up the tree. Considering the delay at both analysis and synthesis stages, the total delay introduced by the tree structured \(b\) band TQMF is \((T-1)(b-1)\) samples, assuming the use of uniform filters at each stage.
3.2.1.3 A Transform Approach for band splitting

A recent attempt to split the speech spectrum into sub-bands has been made by F.S.Yeoh and C.S.Xydeas [14][15]. The generalized structure of the transform approach to sub-band coding is shown in Figure 3.6.

![Figure 3.6: Sub-band coder structure using the DCT transform approach.](image)

Here a block transformation is used to perform the band splitting into a number of equally or unequally spaced bands. The time signals corresponding to these bands can be coded in the same way in SBC with TQMF, using fixed or adaptive bit allocation with forward or backward adaptive quantization. This technique allows for more flexible design approach to frequency domain coding, as a whole range of trade-offs between performance, delay and complexity is possible, to suit specific applications. More
importantly the delay and complexity of the transform approach (in terms of signal processing operations) is substantially reduced compared to sub-band coders employing filter banks with long impulse responses.

The sequence of input samples \( x_n \) is segmented into blocks \( X_n \) of \( N \) samples. Each block \( X_n \) is transformed via an \( N \) point Discrete Cosine Transform (DCT) to yield a block \( Y_n \) of \( N \) transform coefficients. \( Y_n \) is then divided into contiguous blocks \( W_n(1), W_n(2), \ldots, W_n(b) \) each containing \( N/b \) samples (equal bands) where \( b \) is the number of frequency bands. Each of these smaller blocks \( W_n(i) \) is separately inverse transformed via an \( N/b \) point inverse DCT to give the sub-band signals \( Z_n(1), Z_n(2), \ldots, Z_n(b) \). At the receiver the reverse process is performed, the decoded sub-band signals \( Z_n(i) \) are forward transformed with an \( N/b \) point DCT to give the signals \( W_n(i) \). They are then combined in the correct order to form \( Y_n \). A final \( N \) point inverse DCT on \( Y_n \) yields the recovered signal \( x_n \).

The value of \( b \) determines the spectral resolution (number of bands) of the system, which can vary from the fine resolution provided by the ATC to the one band case of waveform coding schemes. Specifically, three cases arise.

(i) \( b = N \)

ie. the number of frequency bands is equal to the transform block size. The transform coefficients \( Y_n \) are coded individually, and the system becomes an adaptive transform coder (ATC), (see section 3.3).

(ii) \( b = 1 \)

ie. no splitting of the signal is performed and the full band signal is directly coded.

(iii) \( 1 < b < N \)

A range of different degrees of spectral resolution can be achieved, with \( b \) defining the fineness of resolution.

Non-uniform splitting of bands can be realised simply by dividing the transform coefficients \( Y_n \) into unequal parts before carrying out the second stage transformation.

3.2.1.4 Parallel Filter Bank (PFB)

Very recently, there has been a growing interest into use of parallel filter banks. If the aliasing cancellation properties of the tree QMF filters are achieved in a parallel filter bank process, with short impulse responses, the system will be less complex and the delay will be substantially reduced compared to the tree QMF.
Two approaches to PFB implementation have been made. The first approach uses band-pass FIR filters of about 64 coefficients each, [16][17]. The number of band-pass filters are equal to the number of bands in the coder, and the same band-pass filters are used at both the encoder and decoder. Consider the example of a 14 band SBC with 64 coefficients filter responses given by \( h_i(k) \), \( i = 1,2,...,14 \) and \( k = 0,1,..,63 \), and sub-band signals represented by \( X_i \). The last two bands are ignored by setting the responses of \( h_{15}(k) \) and \( h_{16}(k) \) equal to zero. The SBC values \( X_i(m) \), \( i = 1,2,...,16 \) are computed in the following way.

\[
X_i(m) = \sum_{k=0}^{63} h_i(k) X_{in}(16n + (i-1) - k) \quad (3.10)
\]

The final output signal \( X_f(n) \) is the result of interleaving the sub-band values, \( X_i(m) \), through the use of a clockwise commutator to produce the desired signal which is the filtered and decimated sub-band signals. See Figure 3.7 and 3.8 for analysis and synthesis implementation of 16 band SBC and Appendix A for the coefficients of 16 parallel filters.

At the decoder, through the use of an anticlockwise commutator the sub-band signals, \( X_{ri} \), \( i = 1,2,...,16 \) are distributed to their corresponding band-pass filters. The output signal \( S_{out} \) is then computed as follows.

\[
X_{out}(n) = 16 \sum_{i=1}^{16} \sum_{k=0}^{63} h_i(k) X_{ri}(m-k)(-1)^{i-1} \quad (3.11)
\]

The second approach uses PFB with a two point FFT, where the number of filters equal to half the bands and has about 80 coefficients, [16]. Consider an example of a 16 band SBC using 8 parallel filters of 80 coefficients in each and two point FFT, see Appendix B for the filter coefficients. The sub-band signals \( X_m(n) \) and \( X_{15-m}(n) \) are computed in the following way.

\[
X_m(n) = Y_A(m) + Y_B(15-m) \quad (3.12)
\]

\[
X_{15-m}(n) = Y_A(m) - Y_B(15-m) \quad (3.13)
\]

\[
X_{14}(n) = X_{15} = 0 \text{ for all } n
\]
Figure 3.7: Parallel filter bank implementation of band splitting in a 16 band SBC.
Figure 3.8: Parallel filter bank implementation of reconstruction in a 16 band SBC.
where,

\[ Y_A(m) = \sum_{i=0}^{30} h_m(2i)X(j-2i) \]  
(3.14)

\[ Y_B(15-m) = \sum_{i=0}^{30} h_m(2i+1)X(j-2i-1) \]  
(3.15)

\[ m = 0, 1, \ldots, 7 \]

At the decoder the output signals \( Y(j+2l) \) and \( Y(j+2l+1) \) are computed in the following way,

\[ Y(j+2l) = 16 \sum_{m=0}^{7} (-1)^m \sum_{i=0}^{4} h_m(16i+2l)Y_m(i) \]  
(3.16)

\[ Y(j+2l+1) = 16 \sum_{m=0}^{7} (-1)^m \sum_{i=0}^{4} h_m(16i+2l+1)Y_{15-m}(i) \]  
(3.17)

where,

\[ X_{14}(n-i) = X_{15}(n-i) = 0 \text{ for all } n \]

\[ Y_m(i) = X_m(n-i) - X_{15-m}(n-i) \]  
(3.18)

\[ Y_{15-m}(i) = X_m(n-i) + X_{15-m}(n-i) \]  
(3.19)

\[ l = 0, 1, \ldots, 7 \]

All of the above approaches for band splitting makes use of frequency domain aliasing cancellation. Similar results may be obtained by considering the Time Domain
3.2.2 Encoding The Sub-Band Signals

After dividing the speech spectrum into desired sub-bands, waveform coding techniques can be introduced to encode the sub-band signals. The most commonly used waveform coding technique in sub-band coders is Adaptive Pulse Code Modulation (APCM). If the number of bands are few so that the samples in each sub-band still show some correlation, a differential type waveform coder can also be used. Depending upon the requirements for delay, performance and complexity, the waveform coders within each sub-band may have one of the two adaptation techniques. These are backward adaptation and forward adaptation. In backward adaptation the quantizer step size is updated for every sample with respect to the previous output codeword from the binary encoder.

\[
\text{Step}(n) = \text{Step}(n-1) a M(I(n-1))
\]  

(3.20)

where \(a\) is a parameter which achieves a smooth adaptation and in practice is just under unity (0.98). \(I(n-1)\) is the previous codeword and \(M\) is a multiplier function which itself is a function of the previous codeword. For simple adaptation, typical values for \(M\) may be, if \(I(n-1)\) is the outermost level of the quantizer then \(M = 2\), else \(M = 0.77\) for a quantizer which has more than one bit. The reason for the restriction to more than one bit is that the backward adaptation cannot be performed with two levels (one bit) quantizer, because there is only one decision level, i.e., the signal is positive or negative. N.S. Jayant suggests multiplier functions up to 5 bit quantizers in reference [19].

If fixed bit allocation is used backward adaptive quantizers do not require any side information to be transmitted to the receiver, and in the case of variable bit allocation the side information required is the number of bits used to code each sub-band signal.

In forward adaptation, the word forward is used to imply that the step sizes of the quantizers are evaluated from the input signal, before it is passed forward to the quantizers, [20][21][22][23]. In order to calculate the step sizes of the quantizers, blocks of speech samples are stored in buffers, and after the computation of the step sizes, these sub-band signals are quantized using these step sizes. Steps are also transmitted to the receiver as side information. One important point to decide is the size of the blocks of samples. For differential coders,
Step \( = f \left[ \frac{1}{B} \sum_{r=2}^{B} (x_r - x_{r-1})^2 \right]^{1/4} \) (3.21)

and for APCM,

\[ \text{Step} = f \left[ \frac{1}{B} \sum_{r=1}^{B} (x_r)^2 \right]^{1/4} \] (3.22)

where \( f \) is a parameter which is a function of the number of bits available in the quantizer and the bit error rate, [25]. \( B \) is the block size, \( x_r \) and \( x_{r-1} \) are the \( r^{th} \) and the \((r-1)^{th}\) samples in each block. The above equations show that the step size is dependent on the standard deviation of the samples in the block \( B \). Hence if \( B \) is small, because the step which will be calculated from \( B \) samples will then be used to encode those samples, the average quantization error becomes smaller. However, because the step is transmitted to the receiver, this will increase the side information. If \( B \) is too long then the average quantization error may be larger, and more importantly the delay may not be tolerable. In forward adaptive systems the side information needed is the step sizes (variances) of each sub-band block for both fixed and variable bit allocation.

3.2.2.1 Bit Allocation

One advantage of sub-band coders noted previously, is the exploitation of the non-flat spectral density of speech signals which allows unequal quantization to be applied to the frequency bands. The allocation of bits for coding each sub-band may be fixed or adaptive.

3.2.2.1.1 Fixed Bit Allocation

In early designs, the number of bits assigned for coding each sub-band signal was determined from long-term signal statistics, and was fixed for a given coder. Crochiere, [7], used the backward adaptive Jayant quantizer (AQJ), [19], for his schemes, while Esteban, [8], employed block quantization with forward transmission of step sizes (AQF), [21]. For a fairly large number of bands, the constraint on available quantizer bits does not in general allow the assignment of 2 bits to code the high frequency bands, a condition which is necessary for the backward adaptation of the AQJ.
3.2.2.1.2 Adaptive Bit Allocation

As speech is a non-stationary signal, fixing the number of bits (from long-term consideration) for coding each sub-band will necessarily be sub-optimal in the short term. Better results can be obtained by allowing the number of bits assigned to each frequency band to vary according to local signal statistics. Adaptive or dynamic techniques of bit allocation attempt to distribute available bits more efficiently by assigning bits to the sub-bands according to their energy composition over a short segment of typically 10 to 30 milliseconds of speech. In this way efficient coding is maintained and no bits are wasted. Naturally, adaptive bit allocation requires the transmission of side information periodically so that the receiver is kept informed of the update in the allocation patterns. The optimum assignment of bits is based on a minimum mean squared error criterion and is given by the well known equation. [9],

\[ R_i = d + \frac{1}{2} \log_2 \left( \frac{\sigma_i^2}{D^*} \right) \quad i = 1,2,...,b \]  

(3.23)

where \( \sigma_i^2 \) is the variance, and \( R_i \) is the optimum number of bits for the \( i^{th} \) sub-band. \( b \) is the number of bands in the sub-band coder, or the number of bands considered in the allocation process, since certain frequency bands beyond the signal cut-off frequency may be omitted. \( d \) is a correction term that reflects the performance of practical quantizers, and \( D^* \) denotes the noise power given by,

\[ D^* = \frac{1}{b} \sum_{i=1}^{b} e_i^2 \]  

(3.24)

where \( e_i^2 \) is the noise power incurred in quantizing the \( i^{th} \) sub-band. The bit assignment obtained must satisfy the constraint of available bits, \( R \),

\[ R = \sum_{i=1}^{b} R_i \]  

(3.25)

It is easy to obtain the result that all bands must have the same distortion. The optimum bit assignment is then.
\[ R_i = \bar{R} + \frac{1}{2} \log_2 \left[ \frac{\sigma_i^2}{\prod_{j=1}^{b} \sigma_j^{2^{1/b}}} \right] \]  

(3.26)

where \( \bar{R} \) is the average bit rate given by,

\[ \bar{R} = \frac{1}{b} \sum_{i=1}^{b} R_i \]  

(3.27)

The \( R_i \)'s calculated from equation (3.24) cannot take on negative or fractional values in practice since they represent the number of quantizer bits to be used. Hence, rounding to the nearest positive integer or zero is necessary within the limits of total bit rate.

The bit allocation equation can be modified slightly to provide some control of the output noise shape which might be desirable from a perceptual point of view. However, the relatively small number of frequency bands in sub-band coders does not allow much room for manoeuvre in this respect. Such frequency domain noise shaping is more appropriate in the context of adaptive transform coding (ATC). (see section 3.3).

The second bit allocation technique [24], is simpler than that above. This again compares the energies of the sub-bands and allocates bits accordingly. The principles of this second technique is quite simple and is as follows,

(i) Find the band with the largest energy.
(ii) Divide this energy by a factor and allocate one bit to that band.
(iii) Check if all the bits are allocated, if yes stop, else repeat the process.

The dividing factor is chosen by listening tests to achieve the best subjective quality. This factor is found in practice to be around 2.

3.3 Adaptive Transform Coding (ATC)

The adaptive transform coder (ATC), [9][10], is a more complex frequency analysis technique which involves block transformations of windowed segments of the input speech. Each segment is represented by a set of transform coefficients which are separately quantized and transmitted. At the receiver, the quantized coefficients are inverse transformed to produce a replica of the original segment. Adjacent segments are then joined together to form the synthesized speech.
3.3.1 The Block Transformation

Block transformation techniques have widely been used in image coding systems with much success and have also been applied to speech coding. The class of transforms of interest for speech processing are the orthogonal time to frequency transformations.

It can be shown, [9], that the gain of a transform coding scheme (using an N point transform) over PCM can be given as,

$$G_{tc} = \frac{\sigma^2}{\prod_{j=1}^{N} \sigma_j^2}^{1/N}$$  \hspace{1cm} (3.28)

where $\sigma^2$ represents the variance of the signal and $\sigma_j^2$ are the variances of the $N$ transform coefficients. This gain is in fact the ratio of the arithmetic and geometric means of the variances of the transform coefficients, since the signal variance $\sigma^2$ for unity transform is equal to the average of the variances of the transform coefficients.

$$\sigma^2 = \frac{1}{N} \sum_{j=1}^{N} \sigma_j^2$$  \hspace{1cm} (3.29)

Zelinsky and Noll [9], obtain the value of $G_{tc}$ for various unitary transforms, using a stationary tenth order Markov process whose first ten autocorrelation coefficients were equal to the first ten long-term autocorrelation coefficients of speech. Figure 3.9 shows the results obtained using various block sizes of the Karhunen-Loeve, discrete cosine, discrete Fourier, discrete slant, and the Walsh-Hadamard transforms.

Note that the DCT has a performance very close to the optimum signal dependent Karhunen-Loeve transform (KLT) and significantly superior to the others. Indeed, the DCT has been found to be ideally suited for coding of speech as well as picture signals. Apart from its signal independence, and its approximation to the KLT, its even symmetry helps to minimize end effects encountered in block coding methods.

The DCT of an $N$ point sequence is formally defined as, [28][29],

$$x_c(k) = \sum_{n=0}^{N-1} x(n) c(k) \cos \left( \frac{(2n+1)k \pi}{2N} \right)$$  \hspace{1cm} (3.30)
where,
\[ c(k) = 1 \quad \text{for} \ k = 0 \]
\[ c(k) = 2^{k/2} \quad \text{for} \ k = 1, 2, \ldots, N - 1 \]

Figure 3.9: Performance comparison of various transforms (DST and WHT are discrete slant and Walsh–Hadamard transforms respectively).

The inverse DCT is defined as,

\[ x(n) = \frac{1}{N} \sum_{k=0}^{N-1} X_c(k)c(k) \cos \left[ \frac{(2n+1)k \pi}{2N} \right] \quad (3.31) \]

\[ n = 0, 1, \ldots, N - 1 \]
Fast algorithms have been derived for implementing the DCT with great computational efficiency, comparable to the FFT. [29].

3.3.2 Quantization Of The Transform Coefficients

The quantization of the transform coefficients is usually made by means of uniform quantizers which are characterized by a step size $\Delta(k)$ and by a number of levels $2^{b(k)}$. The choice of the step size and the number of bits $b(k)$ for a given transform coefficient is of fundamental importance in ATC. Bit allocation will be discussed in the following section (3.3.3).

As observed by Zelinsky and Noll, [9], the probability density function of the (gain normalized) transform coefficients are approximately Gaussian distributed. [23]. Therefore, the choice of optimum (uniform) step size $\Delta(k)$, considering the mean squared error criterion, can be determined from the variance estimate $\hat{\sigma}^2(k)$ according to the theory of Max, [25]. For a given number of bits $b(k)$, the optimum step size is therefore,

$$\Delta(k) = \alpha(b(k))\hat{\sigma}(k)$$  \hspace{1cm} (3.32)

where, $\alpha(b(k))$ is a constant of proportionality, which is a function of the number of bits, and can be found in the tables of Max (see factor $f$ for SBC in equation (3.20)).

From the point of view of subjective quality, however, it is not clear that a mean squared error criterion is the most appropriate choice for determining the step size. Therefore, in practice it is desirable to include an additional factor $Q$, denoted as the quantizer loading factor. Thus,

$$\Delta(k) = Q \alpha(b(k))\hat{\sigma}(k)$$  \hspace{1cm} (3.33)

where $Q = 1$ implies a loading that is optimum in the mean squared (uniform step size) sense. By adjusting $Q$, a trade-off can be made between effects of overload and granular types of distortion in the transform coder.
3.3.3 Bit Allocation

For minimum mean squared error distortion, the number of bits assigned for coding the $N$ transform coefficients is determined by the same bit allocation equations used for sub-band coding, ie, equations (3.21) to (3.25), with $b$ (the number of sub-bands) replaced by $N$ (the number of transform coefficients). Unlike the SBC however, fixed bit allocation is not applicable to ATC. This is because the latter operates by adapting to the fine resolution short-term frequency characteristics of speech which may vary drastically from block to block. Consequently, a bit assignment pattern based on long term statistics would be severely sub-optimal, as has been demonstrated by Zelinsky and Noll. [9]. Further, as was observed previously with regard to SBC, fixed bit allocation requires the assignments of at least one bit to each frequency component to prevent loss of bandwidth in the synthesized speech. This would result in substantial wastage of bits for the transform coder which has typically 128-256 transform coefficients.

3.3.4 Noise Shaping

As in time domain waveform coding techniques, the noise spectrum of frequency domain coders may also be shaped appropriately to improve the perceptual quality of the decoded speech, [26]. The bit assignment rule seen above produces an output noise with flat spectral characteristics, which is known to be perceptually sub-optimal. This flat noise spectrum however, could be controlled to some extent by performing the bit assignment based on a different criterion. The modified bit assignment, [26], is given by:

$$R_i = d + \frac{1}{2} \log_2 \left( \frac{\text{R}_i \sigma_i^2}{D^2} \right)$$

$$i = 0, 1, ..., N - 1$$

where $W_i$ represents a positive weighting. By changing the weighting function $W_i$, the shape of the output noise spectrum can be varied, from the flat minimum distortion case to a shape which follows the input signal spectral envelope. For any particular transmission bit rate, the perceptual optimum value of $W_i$ can be determined by means of listening tests.
3.3.5 Adaptation Strategy

The adaptive bit assignment used in ATC schemes seeks to exploit the non-flatness of the speech signal density, by distributing bits unevenly across the spectrum. The actual step sizes to be used in the quantizer however, needs to be estimated, since the expected spectral levels of the transform coefficients are not known a priori. Thus, some side information which reflects the dynamic properties of speech must be transmitted. This adaptation information is used at both transmitter and receiver to determine the bit assignment pattern and the quantizer step sizes for the block and is therefore of critical importance. Two basic adaptation techniques will now be considered.

3.3.5.1 Zelinsky and Noll's scheme

The best known adaptive transform coder for speech applications is probably the proposal of Zelinsky and Noll shown in block diagram form in Figure 3.10. [9].

A block of \( N \) input speech samples is first normalized by its estimated standard deviation and then transformed into a set of frequency domain coefficients via an \( N \) point DCT. A coarse description of the cosine basis spectrum is extracted and transmitted to the receiver as side information. This (quantized) coarse spectral estimate is used at both transmitter and receiver to calculate the optimum assignment of bits and the quantizer step sizes for coding the coefficients. The spectral estimate consists of a small number of samples computed by averaging the DCT spectral magnitudes. These samples are then geometrically interpolated to yield the expected spectral levels at all frequencies used for determining the quantizer parameters. Excellent synthesized speech quality was reported using this method at 16 Kb/s.

As the bit rate is reduced however, it becomes increasingly difficult to accurately encode the fine structure (pitch details) of the DCT spectrum, and this gives rise to a burbly distortion in the recovered speech. At the same time, the shortage of bits results in wide gaps in the spectrum, as a substantial proportion of coefficients are not transmitted. This leads to significant loss of bandwidth and the so called low-pass effect.

A number of remedial measures have been proposed to combat this quality deterioration at low bit rates. These include uneven spacing of the side information spectral estimates (to give more emphasis to perceptually important frequency regions). [9], ensuring that a minimum proportion of transform coefficients are transmitted and substituting non-transmitted coefficients with an amount of noise (to reduce the low-pass...
effects), and more efficient quantization of the side information by exploiting various redundancies present. However, these attempts have not succeeded in adequately correcting for the inaccuracy of preservation of the short time spectrum, which is the predominant cause of the performance degradation.

Figure 3.10: A block diagram of Adaptive Transform Coder (Zelinsky and Noll's).

3.3.5.2 Vocoder Driven Scheme

A later proposal for low bit rate ATC schemes utilises a more complex speech specific adaptation algorithm based on the traditional model of speech production to predict the DCT spectral levels. The prediction involves two components as illustrated in Figure 3.11. The first is associated with the spectral envelope and the second
with the harmonic (fine) structure of the spectrum. This so called *vocoder driven* ATC, (VDATC), [10], is able to provide a more realistic allocation of available bits according to the fine structure of the spectrum and thus, avoid the quality degradation encountered at low bit rates. A block diagram of the system is shown in Figure 3.12.

The estimate of the short-term DCT spectrum is obtained as follows: The original DCT spectrum is first squared and inverse transformed with an inverse DFT to yield a pseudo autocorrelation function (ACF), rather similar to the normal ACF. The first \( p + 1 \) values of this function are used to define a correlation matrix in the usual normal equation formulation sense (see chapter 4). The solution of these equations yields a Linear Prediction (LP) filter of order \( p \), [27], whose inverse spectrum provides the estimate of the formant structure of the DCT spectrum, (Figure 3.11a). The spectral fine structure is obtained from a pitch model, derived from the maximum value of the pseudo-ACF above the range \( p + 1 \). The corresponding pitch gain \( G \) is the ratio of the pseudo-ACF at this maximum value, over its value at the origin. With these two parameters, a pitch pattern can be generated, (Figure 3.11b). The two spectral components are multiplied and normalized to yield the final spectral estimate used in the bit assignment and step size adaptation process, (Figure 3.11c).
Figure 3.11: Spectral prediction used in vocoder driven transform coder. (a) envelope, (b) fine structure, (c) estimated spectrum.
Figure 3.12: A block diagram of vocoder driven transform coder (a) encoder, (b) decoder.
3.4 References


17. "16 Kbit/s sub-band coder by the Norwegian telecommunication administration for the GSM mobile systems", 1986.


4.1 Basic System Concepts.

For low bit rate speech coding (16 Kbps and below) in the time domain, more speech specific algorithms are required than a simple PCM or a DPCM. Frequency domain coders make use of the non-flat spectral characteristics of speech to reduce the bit rate whilst maintaining reasonable complexity and good quality. Time domain coders, on the other hand, take advantage of the sample to sample correlation as well as periodic similarities present in the time domain speech waveform. The use of these two speech characteristics in coding is called Linear Prediction (LP), and Pitch Prediction (PP). Almost all low bit rate time domain speech coders make use of these two prediction analysis techniques to reduce the signal energy before quantization. The only difference between the various low bit rate time domain speech coders is the way in which they treat the remaining signal which is called the residual. In the following two sections Linear Prediction and Pitch Prediction will be discussed.

4.1.1 Linear Predictive Coding (LPC) Of Speech

In LPC analysis, the combined spectral contributions of the glottal flow, the vocal tract and the radiation of the lips are represented by an all-pole time varying linear filter. The transfer function of this synthesis filter has the form. [3][4][5].

\[
H(z) = \frac{S(z)}{U(z)} = \frac{G}{1 - \sum_{k=1}^{p} a_k z^{-k}}
\]  

(4.1)

Although, the above filter does not represent nasal sounds (it has no zeros) if the order of \( p \) is large enough it becomes a good approximation for almost all the speech sounds. The major advantage of this model is that \( G \) and \( a_k \) can be estimated in a very straightforward and computationally efficient manner by solving a set of linear equations, \( G \) is usually assumed to be unity. The prediction coefficients \( a_k \) are calculated to
minimize the mean squared prediction error.

\[ E_n = \sum_{n=1}^{N} e^2(n) = \sum_{n=1}^{N} [S(n) - \sum_{k=1}^{p} a_k S(n-k)]^2 \] (4.2)

\( E_n \) is then minimized by setting \( \frac{\partial E_n}{\partial a_k} = 0 \) for \( k = 1,2,\ldots,p \).

There are three well known ways of minimizing \( E_n \) and hence calculating the \( a_k \) parameters. These are the autocorrelation, [5], the covariance, [5][6], and the lattice methods, [6][7]. Here, the most common of them, the autocorrelation method will be explained. Setting \( \frac{\partial E_n}{\partial a_k} = 0 \) with \( k = 1,2,\ldots,p \) produces \( p \) equations with \( p \) unknowns which can be written in matrix form as.

\[
\begin{bmatrix}
R_n(1) \\
R_n(2) \\
\vdots \\
R_n(p)
\end{bmatrix} =
\begin{bmatrix}
R_n(0) & R_n(1) & R_n(p-1) \\
R_n(1) & R_n(2) & \vdots & R_n(p-2) \\
\vdots & \vdots & \ddots & \vdots \\
R_n(p-1) & R_n(p-2) & \vdots & R_n(0)
\end{bmatrix}
\begin{bmatrix}
a_1 \\
a_2 \\
\vdots \\
a_p
\end{bmatrix}
\] (4.3)

The above correlation matrix is symmetrical and all the elements along a given diagonal are equal. One of the most efficient way of solving equation (4.3) is the well known Durbin 's recursive procedure, [5], which is as follows.

\[ E_n^0 = R_n(0) \] (4.4)

\[ K_i = [R_n^{(i)} - \sum_{j=1}^{i-1} a_j^{(i-1)} R_n(i-j)] / E_n^{(i-1)} \quad 1 \leq i \leq p \] (4.5)

\[ a_1^{(i)} = K_i \] (4.6)

\[ a_j^{(i)} = a_j^{(i-1)} - K_i a_j^{(i-1)} \quad 1 \leq j \leq i-1 \] (4.7)
Equations (4.5) to (4.8) are solved recursively for \(i = 1, 2, \ldots, p\) and then prediction coefficients are obtained as:

\[
a_j = a_j^{(p)} \quad 1 \leq j \leq p
\]  

(4.9)

4.1.2 Pitch Predictive Coding Of Speech

It was noted earlier that a typical voiced speech waveform is characterized by its periodic structure. The period of this structure is called the pitch period. Accurate estimation of pitch period is essential for a good pitch prediction filter. There are several pitch period estimation algorithms discussed in the literature, the most common ones of which are the average magnitude difference function (AMDF) and the autocorrelation function (ACF). [8][9][10][11][12]. AMDF looks for the minimum value shift and the ACF looks for the maximum value shift, which suggest that they are essentially very similar. In practice the pitch period estimation method is chosen in order to meet the requirements of a specific coder such as delay, complexity and most importantly its performance under specific channel conditions. After estimating the pitch period of speech, a similar procedure to LPC analysis is used to determine the pitch filter coefficients. Typical orders of the pitch filters studied in the literature are one and three, [13].

A single tap (coefficient) pitch filter consists of two parameters. One is the pitch period which determines the delay in the filter and the other is the filter coefficient. The transfer function of a single tap pitch filter can be written as:

\[
H(z) = \frac{S(z)}{U(z)} = \frac{1}{1-\beta z^{-p}}
\]  

(4.10)

where \(\beta\) is the filter coefficient and \(p\) is the pitch period. \(\beta\) is determined by minimizing the mean squared prediction error.

\[
E_n = \sum_{n=1}^{N} e^2(n) = \sum_{n=1}^{N} [S(n) - \beta S(n-p)]^2
\]  

(4.11)
Solving $\frac{\partial E_n}{\partial \beta} = 0$ gives.

$$\beta = \frac{S(n-p)S(n)}{S(n-p)S(n-p)} \quad (4.12)$$

This can be written as.

$$\beta = \frac{R_n(p)}{R_n(0)} \quad (4.13)$$

A better prediction can be obtained by increasing the coefficients of the pitch filter. The second most common pitch filter (for low bit rates) has 3 $\beta$'s and has the following transfer function.

$$H(z) = \frac{S(z)}{U(z)} = \frac{1}{1 - [\beta_1 z^{-(p-1)} + \beta_2 z^{-p} + \beta_3 z^{-(p+1)}]} \quad (4.14)$$

Here the mean squared error is written as.

$$E_n = \sum_{n=1}^{N} e^2(n) = \sum_{n=1}^{N} [S(n) - [\beta_1 S(n-p+1) + \beta_2 S(n-p) + \beta_3 S(n-p-1)]]^2 \quad (4.15)$$

Solving $\frac{\partial E_n}{\partial \beta} = 0$ for $\beta = \beta_1, \beta_2, \beta_3$, produces the following autocorrelation matrix equation.

$$\begin{bmatrix} R(p-1) \\ R(p) \\ R(p+1) \end{bmatrix} \begin{bmatrix} V1 & V2 & V3 \\ V2 & V4 & V5 \\ V3 & V5 & V6 \end{bmatrix} \begin{bmatrix} \beta_1 \\ \beta_2 \\ \beta_3 \end{bmatrix} = \begin{bmatrix} \beta_1 \\ \beta_2 \\ \beta_3 \end{bmatrix} \quad (4.16)$$

where,
\[ V_1 = \sum_{n=1}^{N} [S(n-p+1)]^2 \]
\[ V_2 = \sum_{n=1}^{N} [S(n-p+1)S(n-p)] \]
\[ V_3 = \sum_{n=1}^{N} [S(n-p+1)S(n-p-1)] \]
\[ V_4 = \sum_{n=1}^{N} [S(n-p)]^2 \]
\[ V_5 = \sum_{n=1}^{N} [S(n-p)S(n-p-1)] \]
\[ V_6 = \sum_{n=1}^{N} [S(n-p-1)]^2 \]
\[ R(p-1) = \sum_{n=1}^{N} [S(n)S(n-p+1)] \]
\[ R(p) = \sum_{n=1}^{N} [S(n)S(n-p)] \]
\[ R(p+1) = \sum_{n=1}^{N} [S(n)S(n-p-1)] \]

In the remainder of this chapter some low bit rate time domain speech coders will be discussed.

4.2 Adaptive Predictive Coding (APC)

Predictive coding is an efficient method of converting speech into digital form. [13][14]. The basic idea behind predictive coding is illustrated in Figure 4.1. The coding efficiency is achieved by removing the redundant structure from the signal before quantization. The predictor \( P \) forms the estimate for the current sample of the input speech based on the past samples. The difference between the current value of speech and its predictive value is quantized and sent to the receiver. The receiver constructs the next sample of speech by adding the received signal to the predicted estimate of the present value.

The properties of speech signals vary from one sound to another. It is therefore, necessary for efficient coding that both the predictor and the quantizer in Figure 4.1 be adaptive. The predictor \( P \) includes two separate predictors. These are the linear
prediction (which predicts the envelope) and the pitch prediction (which predicts the fine structure) as seen in section 4.1.

The APC coder of Figure 4.1 provides an improvement in the signal to noise ratio (SNR) over a PCM coder using the same quantizer. The improvement is realized because the power of the quantizer input signal is much smaller than that of the original speech signal. The maximum possible gain in the SNR is generally assumed to be equal to the prediction gain defined as the ratio of the power in the original speech to the power in the prediction residual. Figure 4.2 shows the waveforms of a block of speech and its residuals after LPC and Pitch inverse filtering.

Figure 4.1: A block diagram of adaptive predictive coding (APC)
A typical APC system transmits 2.5 to 3 Kbps side information. Therefore, for 16 Kbps overall transmission, less than 8 KHz sampling frequency is used to take care of the side information and to allocate 2 bits/sample for residual quantization. With only 2 bits/sample for 16 Kbps overall transmission or 1 bit/sample for 8 Kbps overall transmission, it is difficult to avoid both peak clipping of the prediction residual and the granular distortion due to a finite number of levels in the quantizer, and hence large rounding-off errors.

![Figure 4.2: Typical waveforms of (a) original, (b) LPC residual and (c) pitch residual of speech.](image)

The solution to the clipping problem in APC is studied in four sections. The first solution is to use a three tap pitch predictor, [14], to make sure that all of the pitch
pulses are removed before quantization. The second solution is to use a quantizer with locally adaptive step size. [15]. Information regarding the local change in step size is transmitted. This solution redistributes clipping noise as granular noise. These two solutions to the clipping problem maintain a constant number of bits per sample. The next two solutions use a variable number of bits per sample. Itakura in 1978, [16], proposed a pitch adaptive quantizer in which the number of bits are increased with large residual pitch pulses. Information regarding the positions of these large amplitude regions must be transmitted. Finally, the last solution to the clipping problem which is called entropy coding uses a uniform quantizer with an indefinite number of levels, [17][18]. However, to maintain a fixed overall transmission rate a variable length coding can be used. In entropy coding frequently occurring values are coded with a large number of bits in such a way that the overall bit rate is fixed, [19].

Assuming that clipping errors have been minimized the granular noise will continue to be perceived as background hissing noise. The noise usually has a flat spectral envelope that can mask the speech spectrum at high frequencies. The perception of granular noise can be minimized by proper shaping of the noise spectrum, [14][17]. High quality speech can be obtained at 16 Kbit/s with noise shaping. However, the speech quality deteriorates rapidly below 16 Kbit/s.

4.3 Base-Band Coding

Although APC has produced very good results at 16 Kbit/s, the speech quality begins to deteriorate rapidly below 16 Kbit/s. The reason for the reduction in quality is the increased amount of residual quantization noise. One solution to this problem is to use a base-band coder (BBC). BBC also known as voice excited or residual excited coders, [20][21], were originally proposed as a compromise between pitch excited coders (such as LPC, channel and homomorphic vocoders) and waveform coders.

The basic principles of a BBC as shown in Figure 4.3 is to transmit a portion of the residual signal (known as the base-band) and to create the rest of the residual signal at the receiver before passing it through the synthesis filter. The trade-off here is between the noise generated by quantization of the base-band and the noise introduced by the high frequency regeneration (HFR). For a given bit rate a larger base-band leads to increased quantization noise, but decreased HFR noise. An optimal trade-off may be found for any given bit rate.
HFR is one of the most important sections of a typical BBC. It is very important to avoid spectral aliasing, [22], during HFR, which can cause roughness in the output speech. There are three major ways of introducing HFR in BBC.

![Block Diagram of a Typical Base-Band Coder](image)

Figure 4.3: A block diagram of a typical base-band coder.

The first one is a non-linear distortion scheme called rectification which has the following form.

\[ Y(t) = \frac{[(1+\gamma)|x(t)|+(1-\gamma)x(t)]}{2} \]  

(4.17)
where $\gamma$ is a constant in the range $0 \leqslant \gamma \leqslant 1$, $x(t)$ is the received base-band signal and $Y(t)$ is the high frequency signal created.

The most recent HFR technique, which is in common use, is called spectral duplication. The idea behind spectral duplication HFR, [23], derives from the pitch excited coder. Spectral duplication can either be in the time domain or in the frequency domain. One important constraint for the time domain implementation is that base-band with arbitrary width makes the implementation very complex. Implementation is greatly simplified if the signal bandwidth $W$ is an integer multiple of the base-band width $B$; i.e. $W/B = L$ is an integer. Figure 4.4 shows the results for $L = 3$. Figure 4.4a, 4.4b and 4.4c show the original, spectral folding and spectral translation spectrums respectively.

![Figure 4.4](image)

**Figure 4.4**: Spectral demonstration of high frequency regeneration in base-band coders. (a) original spectrum, (b) spectral folding and (c) spectral translation.

To perform spectral folding one simply inserts $L-1$ zeros between samples of the received base-band. This process is the same as upsampling which requires no computations.
Spectral duplication in the frequency domain is achieved by frequency transforming the residual using FFT or DCT. The transformed base-band values are then simply duplicated at higher frequencies. Note that, one now has a fair amount of freedom in performing the spectral duplication because of its simplicity. In particular the signal bandwidth need not be an integer multiple of the base-band width.

4.4 Multi-Pulse Excited Linear Predictive Coder (MPLPC)

Multi-pulse excited linear predictive coders have been proposed to operate from 10 Kb/s down to 7 Kb/s or less, [24][25]. MPLPC may be classified as vocoders with the usual pitch excitation replaced by an optimum excitation. Figure 4.5 shows the block diagram of a LPC speech synthesizer with its multi-pulse excitation signal.

![Diagram of a typical LPC speech synthesizer with traditional pulse-noise and multi-pulse excitations.](Image)
It is quite similar to the traditional LPC synthesizer except for the absence of the pulse and white noise generator and the voiced/unvoiced switch. The excitation for the all-pole filter is generated by an excitation generator that produces a sequence of pulses located at times $t_1, t_2, ..., t_n$, with amplitudes $a_1, a_2, ..., a_n$, respectively.

The locations and amplitudes of the excitation pulses are determined using an analysis by synthesis procedure as shown in Figure 4.6.

Figure 4.6: A block diagram of the analysis by synthesis loop in a multi-pulse coder. (a) encoder, (b) decoder.
The LPC synthesizer produces samples \( \hat{s}_n \) of synthetic speech in response to the excitation \( v_n \). The synthetic speech samples are compared with the corresponding samples of the original speech signal to produce an error signal. This error signal is then modified to take account of how the human perception treats the error. [14]. This noise shaping is similar to that used in APC. The weighted error is then squared and averaged over a short time interval of 5 to 10 msec in duration to produce the mean squared weighted error \( \epsilon \). The locations and amplitudes of the pulses are chosen to minimize this error \( \epsilon \).

The amplitude of each pulse is calculated as follows, [25].

\[
\alpha_k = \frac{\sum_{n=1}^{N} x_n f_n}{\sum_{n=1}^{N} f_n^2} \tag{4.18}
\]

where \( x_n f_n \) is the cross-correlation of the combined filter impulse response \( f_n \) in the analysis by synthesis loop excited by a unit pulse, with \( x_n \), which is the residual signal obtained by subtracting the filter memory carried over from the previous block, from the original input speech. \( f_n^2 \) is the energy in the synthesized speech produced by a single unit pulse.

Total mean squared error, which is minimized by an optimum \( \alpha_k \), is written as.

\[
E_k = \sum_{n=1}^{N} x_n^2 - \frac{[\sum_{n=1}^{N} x_n f_n]^2}{\sum_{n=1}^{N} f_n^2} \tag{4.19}
\]

The optimum pulse location is determined by computing the error \( E_k \) for different values of the index \( k \) from 1 to \( N \) and by finding the minimum of \( E_k \). Alternatively, the best pulse location can be determined by finding the maximum of the second term on the right hand side in equation (4.19). The locations and amplitudes of the pulses are obtained sequentially one pulse at a time. After each pulse has been determined, a new error is computed by subtracting the contribution of the previous pulses and comparing the remaining signal with the contribution of the current pulse. The process of locating new pulses is continued until the number of pulses reaches the maximum value that can
be coded at the specific bit rate. The pulse amplitudes are jointly optimized at each stage by solving a set of linear equations as described in [24]. At each stage only the pulse locations are assumed to be optimal and the amplitudes are updated enabling the pulse amplitudes to be as accurate as possible.

The complexity of MPLPC is very high and increases with the increase in the number of pulses. However, some compact real-time implementations have recently been reported in the literature, [33].

4.5 Code Excited Linear Prediction (CELP)

It was mentioned earlier that the performance of adaptive predictive coders for speech signals using instantaneous quantizers deteriorates rapidly at bit rates below 16 Kb/s and gets worse below 10 Kb/s. The speech quality of the predictive coders was improved at low bit rates by using non-instantaneous stochastic quantizers which minimize a subjective error criterion based on properties of human auditory perception, [26][27][28][29]. Tree search procedures perform very well at 1 bit/sample and the speech quality is maintained even at 0.5 bit/sample. A 0.5 bit/sample tree coder has 4 branches at every node and 4 white Gaussian random numbers on each branch, [26]. The tree search procedures are sub-optimal and the performance of tree coders deteriorates significantly when the signal is coded at only 0.25 bits/sample. Such low bit rates for the residual signal is necessary to bring the total bit rate for coding the speech signal down to 4.8 Kb/s; a rate that offers the possibility of economic digital speech transmission for many radio systems. Fehn and Noll [30] discussed merits of tree coding, trellis coding and code-book coding. Code-book coding is of particular interest at very low bit rates.

In code-book coding, the set of possible sequences for a block of innovation signal is stored in a code-book. For a given speech segment the optimum innovation sequence is selected to optimize a fidelity criterion by exhaustive search of the code-book and an index specifying the optimum sequence is transmitted to the receiver. Exhaustive search for an index is the same as the search for optimizing pulse amplitudes in MPLPC. Although CELP is the most promising low bit rate speech coding technique, it is extremely complex.

Consider the coding of a short block of speech signal 5 msec in duration with 0.25 bit/sample. Each such blocks consists of 40 speech samples at a sampling frequency of 8 KHz. A bit rate of 0.25 bit/sample corresponds to 10 bits for every 40 samples which means 1024 possible sequences. The procedure for selecting the optimum sequence is
shown in Figure 4.7.

Each member of the code-book provides 40 samples of innovation signal. Each sample of the innovation signal is scaled by an amplitude factor that is constant for the 5 msec block and is reset to a new value once every 5 msec. The scaled samples are then filtered sequentially through two recursive filters, one for introducing the voice periodicity and the other for the spectral envelope. The regenerated speech samples at the output of the second filter are compared with the corresponding samples of the original speech signal to produce an error signal. Before comparison, as in MPLPC memory of the filters carried over from previous blocks, is taken away from the original speech to produce the reference signal for comparison. The error signal representing the objective error is further processed through a weighting filter to attenuate those frequencies of which the error is perceptually less important and to amplify those frequencies where the error is perceptually more important. The weighting filter is the same as those used in APC and MPLPC and can be written in z transform notation as follows.
\[ W(z) = \frac{[1 - \sum_{k=1}^{p} a_k z^{-k}]}{[1 - \sum_{k=1}^{p} a_k \alpha^k z^{-k}]} \] (4.20)

where \( a_k \)'s are the LPC coefficients and \( \alpha \) is the factor which controls the spectrum. The weighted mean squared error is determined by squaring and averaging the error samples at the output of the weighting filter for 5 msec block. The optimum innovation sequence is selected as the one with minimum mean squared error. As mentioned earlier, prior to filtering each sample of the innovation sequence is scaled by an amplitude that is constant for the 5 msec block. This scale factor is calculated using equation (4.18), and the error is minimized using equation (4.19). In equation (4.18) \( \alpha_k \) has one value for all pulses and in equation (4.18) and (4.19) \( f_n \) is the output response due to the unit variance innovation sequence. \( f_n^2 \) is the output energy due to unit variance innovation sequence.

Code-books for CELP are usually constructed from white Gaussian numbers. The code-books can also be constructed from the residual signal of the speech by normalizing it to unit variance. Each sample \( v_n \) of the innovation sequence in a Gaussian code-book can be expressed as a Fourier series of \( N \) cosine functions. [27],

\[ v_n = \sum_{k=0}^{N-1} C_k \cos(\pi k_n / N + \Phi_k) \] (4.21)

where \( v_n = 0.1, ..., N-1 \).

where \( C_k \) and \( \Phi_k \) are independent random variables, \( \Phi_k \) is uniformly distributed between 0 and \( 2\pi \), and \( C_k \) is Rayleigh distributed with probability density function,

\[ P(C_k) = C_k \exp(-C_k^2/2) \quad C_k > 0 \] (4.22)

4.6 Harmonic Scaling Of Speech

Harmonic scaling is not a speech coding algorithm in its own right because it involves no quantization. However, it is a useful technique which reduces the sampling rate of the input speech and hence the total bit rate if it is combined with another coder.
such as SBC or ATC.

Sampling rate reduction is based on reducing (for compression) or increasing (for expansion) the interharmonic spectral gaps of the pitch by a factor of up to three using frequency shifting of the pitch harmonics. However, the actual scaling operations are performed in the time domain by means of time domain harmonics scaling (TDHS), [31][32].

Figure 4.8 shows the compression and expansion process of the TDHS. The combination of a waveform coder with TDHS can be viewed as an approach for exploiting the pitch of voiced speech signals in a different way than seen in previous time domain coding algorithms. At the encoder a block of speech is compressed with respect to its pitch period.

\[
S_c(n) = [W(n)S(n)] + [(1-W(n))S(n+p)] \quad n = 1, 2, \ldots, p.
\] (4.23)

where \(S_c(n)\) is the compressed signal, \(p\) is the pitch period and \(S(n)\) is the input speech samples. \(W(n)\) is the weighting function with a window length determined by the compression and expansion factor. For a factor of 2,

\[
W(n) = \frac{(p-n)}{(p-1)}.
\] (4.24)

The meaning of equation (4.23) is that two contiguous blocks of speech are weighted with a triangular window function \(W(n)\) and then summed together to compress two blocks of data into one (block length is \(p\)).

At the receiver the opposite weighting occurs. In order to produce continuous and end-effect free data, at the receiver two blocks of speech are produced for each input block using the window function over three blocks with an overlapping block in the middle.

\[
S(n) = [(1-W(n))S_c(n)] + [W(n)S_c(n+p)]
\] (4.25)

\[
W(n) = \frac{(2p-n)}{(2p-1)} \quad n = 1, 2, \ldots, 2p.
\] (4.26)
In TDHS any integer compression factor may be used. This will depend mainly on delay and quality requirement, because higher compression and expansion factors will require larger window sizes, this means larger delay. Also higher compression/expansion factors introduce larger errors into the recovered speech. The usual compression/expansion factor is 2. Using this compression/expansion factor TDHS-SBC and TDHS-ATC speech at 9.6 Kb/s has been reported with good quality [32].
Figure 4.8: Illustration of (a) TDHS compression and (b) TDHS expansion.
4.7 References


CHAPTER 5

VECTOR QUANTIZATION OF SPEECH SIGNALS

5.1 Basic System Concepts.

The conversion of an analog (continuous-time, continuous-amplitude) source into a
digital (discrete-time, discrete-amplitude) source consists of sampling and quantization
process. Sampling converts a continuous-time signal into a discrete-time signal by
measuring the signal value at regular intervals of time. Quantization converts a
continuous-amplitude signal into a set of discrete-amplitude signal that is different from
the continuous-amplitude signal by the quantization error or noise. When each of a set of
discrete values is quantized separately the process is known as scalar quantization. When
the set of discrete values is quantized jointly as a single vector, the process is known as
vector quantization (VQ), also known as block quantization or pattern matching quanti-
za
tion.

Assume $x = [x_1, x_2, ..., x_N]^T$ is an $N$ dimensional vector whose components
$x_k, 1 \leq k \leq N$ are real valued, continuous-amplitude random variables (the superscript
$T$ denotes transpose). In vector quantization, the vector $x$ is mapped onto another real-
valued, discrete-amplitude, $N$ dimensional vector $y$. $x$ is quantized as $y$ and $y$ is the
quantized value of $x$.

$$y = q(x)$$ (5.1)

where $q(\cdot)$ is the quantization operator. Typically, $y$ takes on one of finite set of values
$y = y_i, 1 \leq i \leq L$, where $y_i = [y_{i1}, y_{i2}, ..., y_{iN}]^T$. The set $y$ is referred to as the recon-
struction code-book, or simply the code-book. $L$ is the size of the code-book, and $y_i$ are
the set of code vectors. The vectors $y_i$ are also known in the pattern recognition litera-
ture as the reference patterns or templates. The size of the code-book is also called the
number of levels. In order to design such a code-book, $N$ dimensional space of the ran-
donm vector is partitioned into $L$ regions or cells $C_i, 1 \leq i \leq L$ and a vector $y_i$ is associ-
ated with each cell $C_i$. The quantizer then assigns the code vector $y_i$ if $x$ is in $C_i$. 

The code-book design process is also known as training or populating the code-book. Figure 5.1 shows an example of the partitioning of two dimensional space ($N = 2$) for the purpose of vector quantization. The region enclosed by the bold lines is the cell $C_i$. Any input vector $x$ that lies in the cell $C_i$ is quantized as $y_i$. The position of the code vectors corresponding to the other cells are shown by dots. The total number of code vectors in the example of Figure 5.1 is $L = 18$.

Figure 5.1: The partitioning of two dimensional space into 18 regions for vector quantization.

For $N = 1$, vector quantization reduces to scalar quantization. Scalar quantization has the special property that whilst cells may have different sizes (step sizes) they all
have the same shape. In the vector quantization cells in two dimensions actually have different shapes. This freedom of having various cell shapes in multi-dimensional space gives vector quantization an advantage over scalar quantization.

When $x$ is quantized as $y$ a quantization error results and a distortion measure $d(x,y)$ can be defined between $x$ and $y$. $d(x,y)$ is also known as a dissimilarity measure or distance measure. As the vectors $y(i)$ are transmitted at different times $i$ one can define an overall average distortion,

$$D = \frac{1}{M} \sum_{i=0}^{M} d[x(i),y(i)]$$  \hspace{1cm} (5.3)

where $M$ is the number of vectors in the data base. For transmission purposes, each vector $y_i$ is encoded into a codeword of binary digits $C_1$ of length $B_1$ bits. The transmission rate $T$ is given by,

$$T = B \cdot F_c \quad \text{bits/second}$$  \hspace{1cm} (5.4)

where,

$$B = \frac{1}{M} \sum_{n=1}^{M} B(n) \quad \text{bits/vector}$$  \hspace{1cm} (5.5)

is the average codeword length, $B(n)$ is the number of bits used to code the vector $x(n)$ at time $n$ and $F_c$ is the number of codewords transmitted per second. The average number of bits per dimension (sample) is,

$$R = \frac{B}{N} \quad \text{bits/sample}$$  \hspace{1cm} (5.6)

When designing a data compression system, one tries to design a quantizer in which the distortion between the original and the quantized vectors is minimized for a given transmission rate. Therefore, when designing a quantizer it is important to decide which
type of distortion measure is likely to minimize the subjective distortion.

5.1.1 Distortion Measures

A distortion measure should be subjectively relevant, so that differences in distortion values can be used as indicating similar differences in speech quality. However, a few decibels of decrease in the distortion may be quite perceptible by the ear in one case but not in another. Whilst objective distortion measures are necessary and useful tools in the design of speech coding systems, decision on direction for improving coder performance should always be made using subjective quality testing.

a) Mean Squared Error.

The most common distortion measure is the mean square error (MSE) defined as,

\[ d(x, y) = \frac{1}{N} (x - y)^T (x - y) = \frac{1}{N} \sum_{k=1}^{N} (x_k - y_k)^2 \]  \hspace{1cm} (5.7)

The popularity of the MSE is due to its simplicity and computability.

b) Weighted Mean Squared Error.

In the mean squared error method, it is assumed that the distortion contributed by quantizing the different parameters \( x_k \) is weighted equally. In general unequal weights can be introduced to render certain contributions to the distortion more important than others. A general weighted mean squared error is then defined by,

\[ d_w(x, y) = (x - y)^T W (x - y) \]  \hspace{1cm} (5.8)

where \( W \) is a positive weighting matrix.

c) The Itakura-Saito Distortion (LPC distortion measure).

In LPC (see chapter 4 section 4.1), the predictor coefficients \( a_k \) are obtained as a result of minimizing the energy of the prediction residual [1].

\[ \sum_{k=1}^{N} a_k R(i-k) = R(k) \quad 1 \leq i \leq N \]  \hspace{1cm} (5.9)
where \( R(i), 0 \leq i \leq N \) are the short term autocorrelation coefficients of the speech signal over a single frame. Direct quantization of these parameters \( a_k \) of an all-pole filter can lead to instability of the filter. In order to reduce the possibility of instability occurring, these coefficients are transformed to another set of parameters known as the reflection coefficient \( K_k, 1 \leq k \leq N \) [2] or the partial correlation (PARCOR) coefficients. For a stable filter the reflection coefficients have the property \( |K_k| < 1, 1 \leq k \leq N \). Therefore, for quantization purposes the reflection coefficients are usually transformed to another set of coefficients that exhibits lower spectral sensitivity as \( K \) approaches 1, which are called log area ratios (LAR) [3][4][5].

\[
LAR_k = \log_{10} \frac{1 + K_k}{1 - K_k}
\]  

(5.10)

The quantization properties of \( K_k \) and \( LAR_k \) have been studied using mean squared error distortion [6][7][8].

An alternative distortion measure used in quantizing predictor coefficients was proposed by Itakura and Saito [9][10], which derives from maximum likelihood principles.

\[
d(x, y) = (x - y) R_x (x - y)
\]  

(5.11)

\( x \) and \( y \) are the current predictor coefficients and code-book entries respectively, and \( R_x \) are the autocorrelation values from which \( x \) is calculated.

d) Perceptually Determined Distortion Measures.

For high bit rates and hence small distortions, reasonable distortion measures including those mentioned above perform well with similar performances. Furthermore, they correlate well with subjective judgements of speech quality. However, as the bit rate decreases and distortion increases simple distortion measures may not be related to the subjective quality of speech. Since vector quantization is expected to be used at low bit rates, it becomes more important to develop and use distortion measures that are better correlated with human auditory behaviour. A number of perceptually based distortion measures have been developed and used. [12][13][14][15][16][17][18]. If high
speech quality at a given bit rate is the most important consideration in a coder design, then a distortion measure that correlates well with human perception should be used.

5.1.2 Code-Book Design

When designing an \( L \) level code-book, \( N \) dimensional space is partitioned into \( L \) cells \( C_i \), \( 1 \leq i \leq L \) and each cell \( C_i \) is assigned a vector \( y_i \). The quantizer chooses the code vector \( y_i \) if \( x \) is in \( C_i \). A quantizer is said to be an optimal quantizer if the distortion in (5.3) is minimized over all \( L \) levels. There are two necessary conditions for optimality. The first condition is that the optimal quantizer is realized by using a minimum distortion or nearest neighbour selection rule. That is, the quantizer chooses the code vector that results in the minimum distortion with respect to \( x \). The second necessary condition for optimality is that each code vector \( y_i \) is chosen to minimize the average distortion in cell \( C_i \). This vector \( y_i \) is called the centroid of the cell \( C_i \). Computation of the centroid of a particular cell depends on the definition of the distortion measure. For either the mean squared error or the weighted mean squared error, distortion in each cell is minimized by,

\[
y_i = \frac{1}{M_i} \sum_{x \in C_i} x(n) \quad (5.12)
\]

That is, \( y_i \) is simply the sample mean of all the training vectors \( M_i \) contained in cell \( C_i \). For Itakura-Saito distortion \( y_i \) is computed by first averaging the normalized autocorrelation corresponding to the sample vectors,

\[
R_{y_i}(k) = \frac{1}{M_i} \sum_{x \in C_i} R_x(k) \quad 0 \leq k \leq N \quad (5.13)
\]

where \( R_x(k) \) are normalized autocorrelation values, such that \( R_x(0) = 1 \). The vector \( y_i \) is then computed by solving (5.9) with \( R_{y_i}(k) \) as the autocorrelation coefficients.

One of the most popular methods for code-book design is an iterative clustering algorithm known as the K-means algorithm [19][20][21][22][23][24]. The algorithm divides the set of training vectors \( x(n) \) into \( L \) clusters \( C_i \) in such a way that the two necessary conditions for optimality are satisfied.
K-means Algorithm

Below, $m$ is the iteration index and $C_i(m)$ is the $i^{th}$ cluster at iteration $m$ with $y_i(m)$ its centroid.

(i) Initialization: Set $m = 0$, choose by an adequate method a set of initial code vectors $y_i(0), 1 \leq i \leq L$.

(ii) Classification: Classify the set of training vectors $x(n), 1 \leq n \leq M$ into the clusters $C_i$ by the nearest neighbour rule,

$$x \in C_i(m) \quad \text{if} \quad d[x, y_i(m)] \leq d[x, y_j(m)] \quad \text{for all} \quad j \neq i.$$

(iii) Code vector updating: $m \rightarrow m + 1$. Update the code vector of every cluster by computing the centroid of training vectors in each cell.

(iv) Termination test: If the decrease in the overall distortion at iteration $m$ relative to $m - 1$ is below a certain threshold, stop; otherwise goto step (ii).

Any other reasonable termination test may be used for step (iv).

The above algorithm converges to a local optimum [20][24]. Furthermore any such solution is in general not unique [25][26]. Global optimality may be achieved approximately by initializing the code vectors to different values and repeating the above algorithm for several sets of initializations and then choosing the code-book that results in the minimum overall distortion.

5.1.3 Computational And Storage Costs

Vector quantization is performed by computing the distortion between $x(n)$ and each code vector in the code-book and choosing the code vector with the minimum distortion as the quantized value of $x(n)$. This type of quantization is called full search. For an $L$ level quantizer, the number of distortion computations needed to quantize a single input vector is $L$. For the mean squared error computation $N$ multiply-adds for each level of the quantizer gives a total computation of $NL$.

If each code vector is encoded into $B = RN$ bits for transmission then,

$$C = N2^{RN} \quad (5.14)$$
where $N$ is the number of elements in each vector and $R$ is the number of bits per element. Thus computation costs grows exponentially with the number of elements in each vector and number of bits per element.

Assuming that one storage location per vector dimension is needed, storage cost $M$ is given by,

$$M = NL = N2^{RN} \quad (5.15)$$

Like computational cost, storage cost is exponential in the number of dimensions and the number of bits per dimension. It is also important to consider the cost associated with the design of the code-book in the first place. In the K-means algorithm, most of the computations result from the classification step. For an $L$ level quantizer $M$ training vectors, and $I$ iterations, the computation cost for training is,

$$C_T = NLMI = N2^{NR} MI \quad (5.16)$$

The storage cost, including the storage cost needed to store all the training vectors is,

$$M_T = N (L+M) \quad (5.17)$$

5.2 Code-Book Design And Search

Vector quantization can offer substantial performance over scalar quantization at very low bit rates. However, these advantages are obtained at considerable computational and storage costs. A number of fast search algorithms have been proposed in the pattern recognition literature [27][28][29], and more recently in vector quantization [30][31], which are designed to reduce the computations in a full search.

5.2.1 Binary Search

With the K-means algorithm, a full search of the $L$ code vectors is required to quantize each input vector. Binary search [32][33], known in the pattern recognition
literature as hierarchical clustering [20][22] is a method for partitioning space in such a way that the search for the minimum distortion code vector is proportional to $\log_2 L$ rather than $L$.

\[ \text{Figure 5.2: Tree splitting of space into 8 cells.} \]

$N$ dimensional space is first divided into two regions (using K-means algorithm with $k = 2$), then each of the two regions is divided further into two sub-regions, and so on, until the space is divided into $L$ regions or cells. Here $L$ is restricted to be a power of 2, $L = 2^B$, where $B$ is an integer number of bits. Each region is associated with a centroid. Figure 5.2 shows the division of space into $L = 8$ cells. At the first binary division $v_1$ and $v_2$ are calculated as the two region centroids. At the second binary division four centroids are calculated as $v_3$ to $v_6$. The centroids of the regions after the third binary division are the actual code vectors $y_1$. An input vector $x$ is quantized, searching the tree along a path that gives the minimum distortion at each node in the path. Again assuming
\( N \) multiply-adds for each distortion computation, computation cost will be,

\[
C = 2N \log_2 L = 2NB
\]  
(5.18)

which is linear with the number of bits.

The total storage cost on the other hand however, is approximately doubled,

\[
M = 2N(L-2)
\]  
(5.19)

Figure 5.3: A block diagram of a two stage cascaded vector quantization.
5.2.2 Cascaded Quantization

A two stage cascaded quantization is shown in Figure 5.3. The major advantage of binary search is the substantial decrease in computational cost relative to full search, with a relatively small decrease in performance. However, the storage of binary search relative to the full search is nearly doubled. Cascaded vector quantization is a method intended to reduce storage as well as computational costs [33][34][35]. Cascaded vector quantization consists of a sequence of vector quantization stages, each operating on the error signal of the previous stage. The input vector \( x \) is first quantized using a \( B_1 \) bit \( L_1 \) level vector quantizer and the resulting error signal is then used in the input to a \( B_2 \) bit \( L_2 \) level second vector quantizer. The sum of the two quantized vectors is the quantized value of the input vector \( x \).

The computation and storage costs for two stage cascaded vector quantization (assuming K-means for each stage) are respectively.

\[
C = N(L_1 + L_2) \quad (5.20)
\]

\[
M = N(L_1 + L_2) \quad (5.21)
\]

5.2.3 Random Code-Books

Whilst it is important to reduce the costs associated with the vector quantization process, there are times that reducing the costs in the training process is of interest. One simple method to design a code-book with essentially no computational training cost is to choose the code vectors at random from a given set of training data. The resulting code-book is called a random code-book. It may appear that a random code-book would not perform well for quantization purposes. However, as the length of the code-book gets longer and the vector dimensions get bigger the performance of a random code-book tends to the performance of the optimal code-book. Stochastic coders use random Gaussian code-books. The difference between a random code-book and random Gaussian code-book is that a random Gaussian code-book contains randomly chosen vectors which themselves contain random numbers. This makes random Gaussian code-books random code-books. However, random code-books are not always Gaussian, and do not always contain random numbers.
5.2.4 Training Testing And Code-Book Robustness

An important aspect of the design of any code-book is the training procedure used to populate the code-book. The training process is simply optimizing a code-book for a given training data by calculating the centroids of the cells. Because the K-means is not guaranteed to result in a code-book that is globally optimum, it is often suggested that one repeats the algorithm with a number of different initial sets of code vectors [21][22][23][36][37].

After a code-book is designed with a given set of training data, it is important to test the performance of the code-book on independent data that was not used in the training. Testing only on the training data always presents an overly optimistic view of how the code-book will perform on operational data.

Code-book robustness refers to the resistance of a code-book to degraded performance when tested on data whose distribution is different from that of the training data. Under operational conditions, one cannot usually predict all of the situations under which a quantizer will be used, and so the distribution of the operational data will in general be different from that of the training data [38]. There are two major types of variations that effect the design and operational performance of a code-book. These are input signal variability and digital transmission channel errors.

For speech, signal variability can be classified further into speaker variability and environmental variability. Speaker variability is that due to changes in each speaker's voice, and may for example be changes due to health conditions. Environmental variabilities refer to the level and type of background noise that surrounds the speaker. For a given bit rate, a speaker independent code-book cannot possibly perform as well as a speaker dependent code-book. One possibility for maximizing performance of a vector quantizer system is to design a speaker independent code-book initially and then as the system is used to have it adapt to the speech of new speakers [16]. Such a system would also have the extra advantage of automatically adapting to the acoustic environment of the speaker.

Transmission channel errors introduce a different type of problem to system robustness. Channel errors translate directly into distortion in the output. Higher error rate means greater distortion. In general vector quantization systems tend to be less robust to random channel errors than scalar quantizers. Consider the example of a 10-bit vector quantizer and 10 one-bit scalar quantizers, and assume a channel error rate of 1%. In the scalar quantizer 1 bit in error causes one value in one dimension to be wrong,
whilst in the vector quantizer the same 1 bit in error causes a whole 10 bit vector to be wrong, which would on average result in larger distortions.

5.3 References


6.1 Introduction

There are three well known candidates for 16 Kb/s speech coding. These are Sub-Band Coding (SBC) [10], Adaptive Transform Coding (ATC) [11] and Adaptive Predictive Coding (APC) [12]. In the following sections, design procedures and simulation results for SBC and ATC is discussed.

6.2 16 Kb/s Sub-Band Coder

A sub-band coder can be divided into three equally important components. These are band splitting, quantization, bit allocation and noise shaping. During the design procedure these three components were simulated separately.

6.2.1 Band Splitting

In chapter 3 four major band splitting techniques were listed. Before simulating them, they were compared in terms of complexity, delay and flexibility. Results of this comparison are shown in Table 6.1. In the Table top to bottom listing is done to represent best to worse in each column. Abbreviations IBS, TQM, PFB and TA represent integer band sampling, tree structure QMF, parallel filter bank and transform approach respectively. During the comparison an 8 equal band SBC application was considered. For other band combinations, the listing order in Table 6.1 may be slightly different. This is because the filter lengths required in PFB, TQM, and IBS will be different. However, the results shown in Table 6.1 will be valid for most sub-band coder applications. From Table 6.1 it is clearly seen that integer band sampling scores the lowest. This is because of its requirement for long filter responses, hence complexity, and its integer band restriction. From Table 6.1 it is also clear that the transform approach was the most complex and the tree structure QMF has large delay. Therefore, it appears that the parallel filter bank solution offers the best compromise. PFB, TA and TQM methods were further compared simply by using them to build an 8 equal bands sub-band coder with
no quantization. Figure 6.1 shows the waveforms of the original input block of speech and the end-to-end error signal waveforms due to splitting alone by PFB, TQM and TA respectively. TA had a 128 point DCT. TQM had 32 tap filters at every stage of splitting and PFB used 48 tap filters.

<table>
<thead>
<tr>
<th>Delay</th>
<th>Complexity</th>
<th>Flexibility</th>
</tr>
</thead>
<tbody>
<tr>
<td>PFB</td>
<td>TQM</td>
<td>PFB</td>
</tr>
<tr>
<td>TA</td>
<td>PFB</td>
<td>TA</td>
</tr>
<tr>
<td>TQM</td>
<td>IBS</td>
<td>TQM</td>
</tr>
<tr>
<td>IBS</td>
<td>TA</td>
<td>IBS</td>
</tr>
</tbody>
</table>

Table 6.1 Comparison of band splitting methods in SBC.

Performance of the two filter bank implementations were similar whilst the performance of the TA was very much higher. Typical signal to noise ratios were 31 dB, 35 dB and more than 100 dB for the TQM, PFB and TA respectively. Although, the filter bank implementations and the transform approach have a huge objective performance difference, listening tests had shown them to be subjectively equal. This is because the human auditory system cannot detect distortion levels lower than SNR's of about 27 dB. If an ideal band splitting procedure was used the expected signal to noise ratio of a sub-band coder due to quantization would be less than 25 dB. Therefore, any of the three band splitting techniques tested can be used in a sub-band coder without limiting its final performance. As a result the parallel filter bank implementation is the best way of splitting the speech into bands. However, as for the simulation purposes because of its easy implementation tree structure QMF will be used in all of the following simulation tests. The same number of taps (32) will be used for the low and high-pass filters at every stage of the tree for the 2, 4 and 8 band SBC's. For the 16 band case, the first three stages will have 32 taps and the last stage will have 16 taps. The filter coefficients are obtained from Johnston's 32 tap (E) and 16 tap (C) designs [1]. These are shown in Table 6.2a and 6.2b respectively.
Figure 6.1: Typical waveforms of (a) original speech and the error caused by (b) TQM approach, (c) PFB approach. (Error of TA is extremely small).
<table>
<thead>
<tr>
<th>Lower-Image</th>
<th>Filter-Coeff</th>
<th>Higher-Image</th>
</tr>
</thead>
<tbody>
<tr>
<td>hl(0)</td>
<td>0.005123</td>
<td>hl(31)</td>
</tr>
<tr>
<td>hl(1)</td>
<td>-0.011276</td>
<td>hl(30)</td>
</tr>
<tr>
<td>hl(2)</td>
<td>-0.000962</td>
<td>hl(29)</td>
</tr>
<tr>
<td>hl(3)</td>
<td>0.015681</td>
<td>hl(28)</td>
</tr>
<tr>
<td>hl(4)</td>
<td>-0.002612</td>
<td>hl(27)</td>
</tr>
<tr>
<td>hl(5)</td>
<td>-0.021038</td>
<td>hl(26)</td>
</tr>
<tr>
<td>hl(6)</td>
<td>0.007380</td>
<td>hl(25)</td>
</tr>
<tr>
<td>hl(7)</td>
<td>0.028123</td>
<td>hl(24)</td>
</tr>
<tr>
<td>hl(8)</td>
<td>-0.014569</td>
<td>hl(23)</td>
</tr>
<tr>
<td>hl(9)</td>
<td>-0.038306</td>
<td>hl(22)</td>
</tr>
<tr>
<td>hl(10)</td>
<td>0.026624</td>
<td>hl(21)</td>
</tr>
<tr>
<td>hl(11)</td>
<td>0.055707</td>
<td>hl(20)</td>
</tr>
<tr>
<td>hl(12)</td>
<td>-0.051383</td>
<td>hl(19)</td>
</tr>
<tr>
<td>hl(13)</td>
<td>-0.097684</td>
<td>hl(18)</td>
</tr>
<tr>
<td>hl(14)</td>
<td>0.138764</td>
<td>hl(17)</td>
</tr>
<tr>
<td>hl(15)</td>
<td>0.459646</td>
<td>hl(16)</td>
</tr>
<tr>
<td>Lower-Image</td>
<td>Filter-Coeff</td>
<td>Higher-Image</td>
</tr>
<tr>
<td>------------</td>
<td>--------------</td>
<td>--------------</td>
</tr>
<tr>
<td>hl(0)</td>
<td>0.006525</td>
<td>hl(15)</td>
</tr>
<tr>
<td>hl(1)</td>
<td>-0.020488</td>
<td>hl(14)</td>
</tr>
<tr>
<td>hl(2)</td>
<td>0.001991</td>
<td>hl(13)</td>
</tr>
<tr>
<td>hl(3)</td>
<td>0.046477</td>
<td>hl(12)</td>
</tr>
<tr>
<td>hl(4)</td>
<td>-0.026276</td>
<td>hl(11)</td>
</tr>
<tr>
<td>hl(5)</td>
<td>-0.099296</td>
<td>hl(10)</td>
</tr>
<tr>
<td>hl(6)</td>
<td>0.117867</td>
<td>hl(9)</td>
</tr>
<tr>
<td>hl(7)</td>
<td>0.472112</td>
<td>hl(8)</td>
</tr>
</tbody>
</table>

(b)

Table 6.2: Coefficients for (a) 32 and (b) 16 tap FIR Quadrature Mirror Filters.

6.2.2 Encoding The Sub-bands

When the number of sub-bands is sufficiently large, certain bands in the high frequency end of the spectrum may not need to be transmitted at all, since they correspond to information beyond the bandwidth of the input signal. The input data used in the simulation is band limited to 3300 Hz and sampled at 8000 Hz, so the frequency band between 3300 Hz and 4000 Hz theoretically does not contain any speech information. Hence, for the so called 8 and 16 band sub-band coders effectively only 7 and 13/14 bands, respectively, are actually transmitted. This is useful in conserving quantizer bits. Figure 6.2 shows the decimated sub-band signals of the 8 band SBC obtained from a typical segment of voiced speech. Notice the characteristic concentration of signal energy in the lower frequency bands and also, the lack of correlation in the signals after decimation. The signal correlation in the sub-bands decreases as the number of bands is increased, since the corresponding spectra becomes progressively flatter as the width of the frequency bands gradually narrows. Table 6.3 shows the average first shift autocorrelation coefficients obtained from the sub-band signals for the 2, 4 and 8 band coders.
Figure 6.2: Typical decimated sub-band signal waveforms.
It can be seen that, apart from the first band of the two-band SBC, little correlation can be expected in the sub-band signals. Correlation values for the same frequency bands also vary widely among different input data. Therefore, the use of differential techniques to encode the sub-band waveforms do not offer any advantage, and consequently, in the simulations all encoding is performed using APCM.

<table>
<thead>
<tr>
<th></th>
<th>a(1)</th>
<th>a(2)</th>
<th>a(3)</th>
<th>a(4)</th>
<th>a(5)</th>
<th>a(6)</th>
<th>a(7)</th>
<th>a(8)</th>
</tr>
</thead>
<tbody>
<tr>
<td>2-Band</td>
<td>0.802</td>
<td>-0.078</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>4-Band</td>
<td>0.620</td>
<td>-0.428</td>
<td>0.121</td>
<td>0.291</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>8-Band</td>
<td>0.203</td>
<td>-0.311</td>
<td>0.410</td>
<td>-0.168</td>
<td>-0.284</td>
<td>-0.21</td>
<td>0.051</td>
<td>-0.016</td>
</tr>
</tbody>
</table>

MALE

<table>
<thead>
<tr>
<th></th>
<th>a(1)</th>
<th>a(2)</th>
<th>a(3)</th>
<th>a(4)</th>
<th>a(5)</th>
<th>a(6)</th>
<th>a(7)</th>
<th>a(8)</th>
</tr>
</thead>
<tbody>
<tr>
<td>2-Band</td>
<td>0.731</td>
<td>-0.381</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>4-Band</td>
<td>0.382</td>
<td>-0.413</td>
<td>0.412</td>
<td>-0.081</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>8-Band</td>
<td>-0.291</td>
<td>-0.262</td>
<td>0.327</td>
<td>-0.262</td>
<td>-0.036</td>
<td>0.259</td>
<td>-0.302</td>
<td>0.076</td>
</tr>
</tbody>
</table>

FEMALE

Table 6.3: One shift correlation coefficients for Sub-Band signals of 2, 4, and 8 band SBC's.

6.2.2.1 Quantization

The sub-band signals are normally coded using APCM-AQF (forward adaptive APCM), particularly when the number of bands is large. The step sizes employed in the quantization are determined from the signal variance of each band, which are transmitted as side information. The proportion of available bits assigned for the side information depends on the frequency of update of the quantizer step sizes. Table 6.4 shows the segmental signal to noise ratio (SegSNR) results obtained for the 2, 4, 8 and 16 band sub-band coders simulated, where the quantizer step sizes (using Max's quantization figures
in reference [21] and bit allocation patterns are updated after 256, 128, 64 and 32 input samples. Allowance has been made for the side information required for transmission of sub-band variances (5 bits each per block), so the results apply for a total transmission rate of 16 Kb/s.

<table>
<thead>
<tr>
<th>Block-Size</th>
<th>256</th>
<th>128</th>
<th>64</th>
<th>32</th>
</tr>
</thead>
<tbody>
<tr>
<td>-</td>
<td>Male</td>
<td>Female</td>
<td>Male</td>
<td>Female</td>
</tr>
<tr>
<td>2-Band</td>
<td>15.7</td>
<td>15.4</td>
<td>16.1</td>
<td>15.7</td>
</tr>
<tr>
<td>4-Band</td>
<td>19.0</td>
<td>18.9</td>
<td>18.1</td>
<td>17.7</td>
</tr>
<tr>
<td>8-Band</td>
<td>20.6</td>
<td>19.7</td>
<td>20.7</td>
<td>19.2</td>
</tr>
<tr>
<td>16-Band</td>
<td>20.9</td>
<td>19.7</td>
<td>19.9</td>
<td>18.3</td>
</tr>
</tbody>
</table>

Table 6.4: Segmental SNR (dB) performance for sub-band coder employing adaptive bit allocation and APCM-AQF for 16 Kb/s.

It can be seen that the SNR generally increases with the number of sub-bands and reaches its peak when the number of bands is 16. SNR also falls as the blocksize for updating the quantizer is reduced, since proportionately less bits are available for signal coding, due to the resulting increase in the side information. A quantizer update block size of 128 samples (16 msec) appears to be a good compromise in terms of performance and delay.

6.2.3 Bit Allocation and Noise Shaping

Both fixed and adaptive methods of assigning bits to code the sub-band signals were investigated. Adaptive bit allocation is performed using the two methods discussed in chapter 3. First adaptive bit allocation is performed using the formula,

\[
R_i = \bar{R} + \frac{1}{2} \log_2 \sigma_i^2 - \frac{1}{2b} \sum_{j=1}^{b} \log_2 \sigma_j^2
\]  
(6.1)
As $R_i$ can only take on integer values, each value as derived from (6.1) must be rounded to the nearest positive whole number or zero. Following this, further adjustments must be made to ensure that the integer bit assignment satisfies the constraint on available bits. The full bit allocation procedure as implemented in the simulation involves the following steps:

(i) The variances $\sigma_i^2$ of each sub-band signal over an appropriate time segment (typically 8 to 32 msec) are first calculated.

(ii) Sub-bands which are beyond the input signal's frequency range (such as 8 for the 8-bands and 14,15,16 for the 16-bands) are effectively prevented from being assigned bits by excluding them from the bit assignment procedure.

(iii) These values of $\sigma_i^2$ are then used in the bit assignment equation of (6.1) to obtain the $R_i$'s. The average bit rate $\bar{R}$ used in the equation must be modified to account for channel capacity occupied by the side information.

(iv) The $R_i$'s are then rounded up or down to the nearest integer value to give the bit assignment map.

(v) Further adjustments are necessary to ensure that the constraint on available bits is satisfied and that no band receives more than the maximum allowable number of bits. If more bits than available have been allocated then the excess bits are taken away from bands which least deserve them, i.e. for which the integer rounding process adds the greatest amount. For example, a band with an initial $R_i$ of 3.6, rounded to 4 is deemed to be less deserving than one with an initial $R_i$ of 4.8 rounded up to 5. Similarly, when the number of bits allocated is fewer than available, the extra bits are given to bands which most deserve them, i.e. the bands from which the integer rounding process takes away the greatest amount.

The second bit allocation procedure is far simpler than that discussed above. The reason for this is that it does not require the $\log_2$ calculations and by its nature does not need to be checked against the integer rounding errors. It involves the following steps.

(i) The energies $\sigma_i^2$ of each sub-band signal over an appropriate time segment (typically 8 to 32 msec) are first calculated.

(ii) The band with the largest $\sigma_i^2$ is allocated 1 bit and its $\sigma_i^2$ is divided by 2.

(iii) Check if all bits are allocated, stop else go to (i).

Adaptive bit allocation is generally used with forward adaptive quantization of the sub-bands, where the sub-band signal variances are transmitted to the receiver. The
quantized versions of these variances are used at both transmitter and receiver to compute the bit allocation pattern and the quantizer step sizes. This ensures that the parameters used at both ends are identical. Consequently, the bit allocation algorithm uses \( \hat{\sigma}_i \), instead of \( \sigma_i \), in practice. The fixed bit allocation map may be obtained by using the same procedure and averaging the bits assigned to each frequency band over the long term. However, to prevent loss of bandwidth in the synthesized speech, at least one bit must be assigned to each frequency band, even though some of the high frequency bands contain insignificant information most of the time. Because of this inefficient utilisation of available bits, and the inability to properly track the short term signal spectral variations, the performance of sub-band coders employing fixed bit allocation is necessarily inferior to the much reduced complexity. A typical bit pattern for an 8 band SBC operating at 16 Kb/s is 33331111 for an input signal band limited from 0 to 4000 Hz. Variable bit allocation was found to be far superior to fixed bit allocation, and hence, in the following simulation results variable bit allocation was used.

6.2.4 Simulations

During the optimization process of a 16 Kb/s sub-band coder, three major areas were considered. These are adaptation of the quantizers for quantization of the sub-band signals, determination of the optimum number of sub-bands, and quantization of the side information with minimum number of bits.

6.2.4.1 Block Forward Adaptive (AQF) and Backward Adaptive Quantization (AQB)

In order to compare the performance of AQF and AQB two similar 8 band sub-band coders were simulated. One had a AQF and the other AQB quantizers. These two coders were then tested and compared using the same conditions and parameters.

6.2.4.1.1 Sub-Band Coder With AQF

Coder specifications are as follows:

* Number of bands : 8 (7 actually used).
* Quantizer update rate : 256 samples (every 32 msec).
* Maximum bits per band : 6.
* Side information : 5 bits per band (1093.76 b/s).
* Sub-band signal rate : 15000 bits/sec.
Total bit rate : 16093.75 bits/sec.

In order to avoid 93.75 bits/sec excess capacity 3 samples per block (every 32 msec) from one of the low energy band (1 bit allocated) were not transmitted.

<table>
<thead>
<tr>
<th>Quantizer Bits</th>
<th>6</th>
<th>5</th>
<th>4</th>
<th>3</th>
<th>2</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Quantizer Step</td>
<td>0.100</td>
<td>0.172</td>
<td>0.350</td>
<td>0.660</td>
<td>1.000</td>
<td>1.050</td>
</tr>
</tbody>
</table>

Table 6.5: APCM-AQF quantizer step sizes for various bits.

<table>
<thead>
<tr>
<th>Band</th>
<th>Bandwidth (KHz)</th>
<th>SNR (dB)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0-0.5</td>
<td>23.23</td>
</tr>
<tr>
<td>2</td>
<td>0.5-1.0</td>
<td>24.93</td>
</tr>
<tr>
<td>3</td>
<td>1.0-1.5</td>
<td>9.64</td>
</tr>
<tr>
<td>4</td>
<td>1.5-2.0</td>
<td>10.43</td>
</tr>
<tr>
<td>5</td>
<td>2.0-2.5</td>
<td>4.87</td>
</tr>
<tr>
<td>6</td>
<td>2.5-3.0</td>
<td>3.34</td>
</tr>
<tr>
<td>7</td>
<td>3.0-3.5</td>
<td>0.25</td>
</tr>
</tbody>
</table>

Total Segmental SNR = 20.7 dB.

Table 6.6: SNR performance of a 7 band SBC-AQF.

During the simulations band energies were not quantized. Band energies were used to normalize the sub-band signals before quantization. Only one quantizer was used to quantize the sub-band signals of each band (since they have the same distribution). The step sizes of this quantizer with respect to the number of bits allocated to each band were simulated. 6.5, and 4 bit quantizer step sizes were simulated using the samples in the first and the second sub-bands, 3.2, and 1 bit quantizer step sizes were simulated using the data in the last five bands. Results are shown in Table 6.5. By employing the coder with the logarithmic variable bit allocation technique a complete coder was then tested. Each band’s signal to quantization noise ratio, as well as the total (reconstructed)
signal to noise ratio, was calculated as shown in Table 6.6.

Informal listening tests were also conducted. These tests showed that the processed speech was almost exactly the same as the original quality.

6.2.4.1.2 Sub-Band Coder With AQB

The same data with the same specifications was processed using SBC-AQB at the total bit rate of 16093.75 bits/sec. The coder was also employed with a variable bit allocation algorithm. Therefore, to adapt the one bit AQB, results in [15] were used based on the following assumption. Firstly, it was assumed that the speech waveform in each band had approximately the same shape with different amounts of energy. When variable bit allocation is used, because the bit allocation process is based on the energies of each band, the second assumption was to assume that the number of bits in each band represents the energy content of each band. It works as follows.

(i) Choose a band which has large energy (low frequency band, eg. first and the second bands in 8 and 16 band SBC respectively).

(ii) When only one bit per sample in a block is assigned to a certain band, use it to transmit the sign of each sample in that block.

(iii) Find a number of scaling factors which will be dependent on the number of bits assigned to the reference band and get the amplitude of the one bit assigned band samples by scaling down the reference band samples and use their transmitted sign bits.

Optimum scale factors given in [15] are tabulated in Table 6.7.

Figure 6.3 illustrates how the method of scaling the one-bit AQB output to a reference band compares with the original unquantized signal. The example is for the sixth band of an 8 band SBC when the reference (first) band is assigned 6 bits. Notice that the signal envelope for the one bit band has been reasonably well preserved. N.S. Jayant gives AQB multiplier functions up to 5 bit quantizers in reference [3]. However, as it was decided to use variable bit allocation and allow maximum bits per band of up to 6, multiplier values for the 6 bit quantizer from [15] were used. These multiplier values are listed in Table 6.8. In the Table going from left to right and top to bottom represents multiplier values for quantizer levels going from 1st to 32nd.
Figure 6.3: One bit adaptive AQB quantized 6th band signal of an 8 band SBC when reference band has 6 bits. (a) original, (b) quantized.
The complete SBC-AQF coder employed with the scale factors listed in Table 6.7 and multiplier values given in reference [3] and Table 6.8 was then tested both objectively and subjectively. Individual sub-bands together with the overall signal to quantization noise ratios are tabulated in Table 6.9.

The test data used was 2 seconds long low pitch male sentence, "an apple a day keeps the doctor away". The reason for choosing a mainly voiced low pitch male test data was to ensure the short time first order sample to sample correlation was maximized. This would make it possible to measure the maximum performance of a SBC-AQB. Although, SBC-AQB coder produced highly intelligible speech, its quality compared to SBC-AQF was inferior. Quantization noise was clearly audible. Processed speech also had large amount of high frequency aliasing noise. Objective results tabulated in Table 6.6 and 6.9 show that, there is at least 3 dB difference between SBC-AQF and SBC-AQB. Informal subjective tests however, even showed a larger difference between the two adaptation techniques. This is because AQB systems are based on the correlation in the signal and as it was shown earlier the correlation in sub-band signals is almost non-existent. On the other hand the AQF system does not relay on any correlation at all, but on the distribution of the signal (Gaussian approximated).
<table>
<thead>
<tr>
<th>Band</th>
<th>Bandwidth (KHz)</th>
<th>SNR (dB)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0-0.5</td>
<td>17.95</td>
</tr>
<tr>
<td>2</td>
<td>0.5-1.0</td>
<td>21.95</td>
</tr>
<tr>
<td>3</td>
<td>1.0-1.5</td>
<td>7.73</td>
</tr>
<tr>
<td>4</td>
<td>1.5-2.0</td>
<td>8.09</td>
</tr>
<tr>
<td>5</td>
<td>2.0-2.5</td>
<td>3.43</td>
</tr>
<tr>
<td>6</td>
<td>2.5-3.0</td>
<td>1.20</td>
</tr>
<tr>
<td>7</td>
<td>3.0-3.5</td>
<td>0.31</td>
</tr>
</tbody>
</table>

Total Segmental SNR = 17.123 dB

Table 6.9: SNR performance of a 7 band SBC-AQB.

6.2.4.2 Optimum Number Of Bands

A sub-band coder can have a wide range of numbers of bands such as 2, 4, 8, 16, and even 32. As it was decided earlier to use AQF quantization and hence variable bit allocation, it is desirable to have as many bands as possible, in order to distribute the available bits in the best possible way. However, as it was also discussed earlier, in an AQF-variable bit allocated coder band energies are sent as side information, more bands will mean more side information. In order to find the optimum number of bands a sub-band coder should have, 2, 4, 8, and 16 bands sub-band coders were simulated using the same data and specifications listed:

Test Data: "Hello operator operator" and "Yes what can I do for you". Male and Female speech respectively with about 1.5 seconds idle segment in between.

Quantizer update: AQF at every 256 samples (every 32 msecs).

Side information: Allowance made for 5 bits per band per 32 msecs.

All four coders were compared as shown in Table 6.10.
As can be seen from Table 6.10, the best performance sub-band coder has 16 bands (only 13 used). This was expected because the 16 band coder had the largest average bits/sample ratio, and since it had more bands than the 2, 4, and 8 band coders, it made use of the variable bit allocation much more efficiently than the others. However, if the number of bands is doubled to 32, the side information will occupy 25% of the overall capacity leaving only 1.74 bits/sample on average, to quantize the actual speech signals. This will produce degradation in the processed speech.

The other important result observed was that both 8 and 16 band coders had reduced SNR values (5 to 6 dB) to the 8 band coder tested in section 6.2.1.1. The first reason for this is the characteristics of the two test sentences. "An apple a day keeps the doctor away", contains higher low frequency formants and small high frequency formants. "Hello operator operator yes what can I do for you", on the other hand, has a much more even spectrum. See Figure 6.4 for spectral comparison of the two test sentences. When variable bit allocation was applied to both of these sentences it was observed that the bit pattern in the bands for every block showed different variations. As expected, due to its spectral shape, in "Apple a day ....", low frequency bands were allocated bits which varied from 3 to 6 and high frequency bands were allocated with only 1 or at maximum 2 bits. In "Hello operator....", variations of bits between low and high frequency bands was not so large and this was due to its flatter spectrum. The sum

<table>
<thead>
<tr>
<th>Bands</th>
<th>2</th>
<th>4</th>
<th>8</th>
<th>16</th>
<th>32</th>
</tr>
</thead>
<tbody>
<tr>
<td>B.width (KHz)</td>
<td>0-4</td>
<td>0-4</td>
<td>0-3.5</td>
<td>0-3.25</td>
<td>0-3.375</td>
</tr>
<tr>
<td>Bits-per-sample</td>
<td>2</td>
<td>1.75</td>
<td>2.142</td>
<td>2.153</td>
<td>1.74</td>
</tr>
<tr>
<td>Side-Infor. (bps)</td>
<td>312.5</td>
<td>625</td>
<td>1093.75</td>
<td>2031.25</td>
<td>4218.75</td>
</tr>
<tr>
<td>Bit-Rate (bps)</td>
<td>16321.5</td>
<td>14625</td>
<td>16093.75</td>
<td>16031.25</td>
<td>15968.75</td>
</tr>
<tr>
<td>Max bit-per-band</td>
<td>3</td>
<td>4</td>
<td>6</td>
<td>6</td>
<td>-</td>
</tr>
<tr>
<td>SNR (dB)</td>
<td>12.74</td>
<td>13.82</td>
<td>15.45</td>
<td>16.27</td>
<td>-</td>
</tr>
</tbody>
</table>

Table 6.10: Comparison of Sub-Band coders with respect to the number of bands in the coder.
Figure 6.4: Spectral comparison of (a) "an apple .", (b) "hello ." test sentences.
of the quantization levels in sub-bands per block for data which has flatter spectra will be less than of data containing higher formant variations.

Consider the example of a 16 band coder (13 bands used) which has 2 bits per sample per allocated band. The total quantization levels per block will be 

\[(2.2)^{13} = 52\]

However, if it was such that 3 bands with 5 bits, 1 band with 2 bits, and 9 bands with 1 bit, the total quantization levels per block would then be

\[(2.2.2.2.2.2.2.2.2.2)3 + (2.2)1 + (2)9 = 118\]

The second reason is that "Hello operator ...." contained 40% non-speech signal which again for the reason discussed above, decreased the signal to quantization noise ratio.

Subjective comparison of the coders by informal listening tests showed that as the number of bands in the coder was increased from 2 to 16, quality of the processed speech was also increased. Although, the 8 band coder had about 5.5 dB less SNR that the one tested in section 6.2.1.1, subjectively, the qualities of both were very similar. The 16 band coder had even better quality than either of the two 8 band coders. In 4 and 2 band coders quantization noise was very much noticed.

6.2.4.3 Quantization Of Side Information

In a sub-band coder, if AQF is used the only side information required is the band energies. However, if AQB is used no side information is needed. In our designs our objective was to maximize the quality and keep the complexity and delay within reasonable limits. Therefore, it was decided to use a SBC-AQF system which requires the transmission of band energies as side information. In all of the previous tests 5 bits per band side information was allocated but the actual band energies were not quantized. During the observation of the dynamic range of the band energies it was noticed that some bands had larger dynamic range variations than others. Therefore, 13 separate uniform 5 bit PCM quantizers were designed to quantize 13 band energies (in the previous section it was decided that the 13 band SBC was the best SBC). During the design of these quantizers, data from each band was used for each corresponding quantizer. An initial step size was chosen for each quantizer and this step size was either increased or decreased as long as the overall SNR was increased. Finally, when SNR started to decrease the previous, SNR maximized step size was chosen to be the optimum for each quantizer.

Using these quantizers a 13 band SBC was then tested. Subjectively, while the test data was active, the quality of the processed speech was very much like the original.
However, during silence periods the processed speech contained very annoying background noise. Although, the SNR performance of the coder was very similar to those tested without quantizing the band energies, this idle noise was at an unacceptable level. The reason for the idle background noise was that, when the uniform PCM (5 bit) quantizers were designed, they had step sizes determined by maximizing the average SNR in each band. In order to cover the instants which most contributed to overall SNR (active sections) step sizes were optimized to be very much larger than was needed for the idle sections. During quantization of the idle band energies, very large positive error (sometimes larger than the band energies themselves) was introduced. As the quantized band energies determine the APCM quantizer step sizes to quantize the sub-band signals, these large errors were then introduced into the sub-band signals. This was heard as idle background noise at the decoder.

In order to avoid noise due to inefficient side information quantization, non-uniform PCM quantizers were tested and used. These quantizers had initial step sizes (first levels of the quantizers) which were chosen to be approximately equal or twice the minimum energy level in each band. Remaining quantizer levels were calculated by the expression:

\[
\text{Level}(x) - \text{Level}(x-1) = M[\text{Level}(x-1) - \text{Level}(x-2)]
\] (6.2)

For each band, initial step size together with the suitable multiplier factor \( M \) was calculated as shown in Table 6.11.
<table>
<thead>
<tr>
<th>Band</th>
<th>Initial-Step</th>
<th>M</th>
<th>Quantizer-Bits</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>16.8</td>
<td>1.069</td>
<td>5</td>
</tr>
<tr>
<td>2</td>
<td>22.4</td>
<td>1.047</td>
<td>5</td>
</tr>
<tr>
<td>3</td>
<td>5.6</td>
<td>1.035</td>
<td>5</td>
</tr>
<tr>
<td>4</td>
<td>16.8</td>
<td>1.043</td>
<td>5</td>
</tr>
<tr>
<td>5</td>
<td>8.4</td>
<td>1.105</td>
<td>5</td>
</tr>
<tr>
<td>6</td>
<td>8.4</td>
<td>1.035</td>
<td>5</td>
</tr>
<tr>
<td>7</td>
<td>8.4</td>
<td>1.038</td>
<td>5</td>
</tr>
<tr>
<td>8</td>
<td>8.4</td>
<td>1.092</td>
<td>5</td>
</tr>
<tr>
<td>9</td>
<td>8.4</td>
<td>1.302</td>
<td>5</td>
</tr>
<tr>
<td>10</td>
<td>16.8</td>
<td>1.296</td>
<td>4</td>
</tr>
<tr>
<td>11</td>
<td>5.6</td>
<td>1.310</td>
<td>4</td>
</tr>
<tr>
<td>12</td>
<td>16.8</td>
<td>1.318</td>
<td>4</td>
</tr>
<tr>
<td>13</td>
<td>22.4</td>
<td>1.362</td>
<td>4</td>
</tr>
</tbody>
</table>

Table 6.11: Non-uniform PCM characteristics for 13 band energies.

<table>
<thead>
<tr>
<th>Bits</th>
<th>7</th>
<th>6</th>
<th>5</th>
<th>4</th>
<th>3</th>
<th>2</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Step</td>
<td>0.035</td>
<td>0.074</td>
<td>0.145</td>
<td>0.275</td>
<td>0.540</td>
<td>0.960</td>
<td>1.246</td>
</tr>
</tbody>
</table>

Table 6.12: APCM step sizes for a 13 band SBC for various bits.

Before testing the coder APCM step sizes for the 13 band SBC with respect to the number of bits were simulated and found to be as shown in Table 6.12.
<table>
<thead>
<tr>
<th>Band</th>
<th>Bandwidth (Hz)</th>
<th>Step</th>
<th>Bits</th>
<th>Performance (dB)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0-250</td>
<td>0.315</td>
<td>5</td>
<td>37.5</td>
</tr>
<tr>
<td>2</td>
<td>250-500</td>
<td>0.220</td>
<td>5</td>
<td>30.0</td>
</tr>
<tr>
<td>3</td>
<td>500-750</td>
<td>0.175</td>
<td>5</td>
<td>28.2</td>
</tr>
<tr>
<td>4</td>
<td>750-1000</td>
<td>0.120</td>
<td>5</td>
<td>27.7</td>
</tr>
<tr>
<td>5</td>
<td>1000-1250</td>
<td>0.295</td>
<td>4</td>
<td>19.0</td>
</tr>
<tr>
<td>6</td>
<td>1250-1500</td>
<td>0.260</td>
<td>4</td>
<td>20.3</td>
</tr>
<tr>
<td>7</td>
<td>1500-1750</td>
<td>0.165</td>
<td>4</td>
<td>21.5</td>
</tr>
<tr>
<td>8</td>
<td>1750-2000</td>
<td>0.210</td>
<td>4</td>
<td>19.0</td>
</tr>
<tr>
<td>9</td>
<td>2000-2250</td>
<td>0.245</td>
<td>4</td>
<td>20.3</td>
</tr>
<tr>
<td>10</td>
<td>2250-2500</td>
<td>0.165</td>
<td>4</td>
<td>22.1</td>
</tr>
<tr>
<td>11</td>
<td>2500-2750</td>
<td>0.450</td>
<td>3</td>
<td>17.0</td>
</tr>
<tr>
<td>12</td>
<td>2750-3000</td>
<td>0.360</td>
<td>3</td>
<td>18.5</td>
</tr>
<tr>
<td>13</td>
<td>3000-3250</td>
<td>0.475</td>
<td>3</td>
<td>15.2</td>
</tr>
</tbody>
</table>

Table 6.13: Uniform PCM quantizer step sizes for block normalized 13 band SBC.

The subjective quality of the coder was much more noise free. However, it was noticed that the quality was talker and sentence dependent. Initial step sizes and \( M \) values were very sensitive to talkers energy levels variations in each band. When the training data was large enough to get optimum initial step sizes and \( M \) values, it was noticed that 5 bits per band was not adequate for some low frequency bands. We needed to have some kind of control over the energy variations from talker to talker. This was achieved by block normalizing the data before splitting. This of course needed extra side information. However, it was observed as shown in Table 6.13 that some bands (high frequency) did not need the total 5 bits. Therefore, the savings made were used to quantize the block energy. A seven bit non-uniform quantizer with an initial step size of 22
and \( M \) value of 1.009 produced 44.1 dB overall SNR. Uniform PCM quantizers for band energies were then designed knowing that the sum of the band energies per block would be unity. Step sizes of these quantizers together with the number of bits and their average SNR's are listed in Table 6.13.

<table>
<thead>
<tr>
<th>Band</th>
<th>SNR (dB)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>14.36</td>
</tr>
<tr>
<td>2</td>
<td>21.00</td>
</tr>
<tr>
<td>3</td>
<td>20.17</td>
</tr>
<tr>
<td>4</td>
<td>15.71</td>
</tr>
<tr>
<td>5</td>
<td>13.97</td>
</tr>
<tr>
<td>6</td>
<td>13.71</td>
</tr>
<tr>
<td>7</td>
<td>13.22</td>
</tr>
<tr>
<td>8</td>
<td>12.41</td>
</tr>
<tr>
<td>9</td>
<td>14.79</td>
</tr>
<tr>
<td>10</td>
<td>12.89</td>
</tr>
<tr>
<td>11</td>
<td>13.77</td>
</tr>
<tr>
<td>12</td>
<td>11.59</td>
</tr>
<tr>
<td>13</td>
<td>13.12</td>
</tr>
</tbody>
</table>

Total Segmental SNR = 20.26 dB.

Table 6.14: SNR performance of a fully quantized 13 band 16 Kb/s SBC.

The objective performance of the coder was as good as when unquantized band energies were used. Band SNR's and the overall segmental SNR are tabulated in Table 6.14. Overall quality of the processed speech was very similar to the original even when
compared using highly sensitive ear-phone.

The final designed 13 band SBC had the listed specifications.

- AQF update: Every 192 samples (every 24 msecs).
- Side Information: 60 bits/block (2500 bits/sec).
- Max bit per band: 5.
- Bit allocation: Divide $c$ by 2 and allocate one bit.
- Allocated bits/block: 27.
- Sub-Band bit rate: 13500 bits/sec.
- Total bit rate: 16000 bits/sec.

### 6.2.5 Further Considerations On Bit Allocation And Quantization.

In this section we discuss some issues related to the bit allocation and quantization procedures for sub-band signals.

#### 6.2.5.1 Forward And Backward Adaptation Variations

Forward adaptive quantization of the sub-band signals, although undoubtedly efficient, becomes progressively less attractive as the number of bands employed increases. This is because the side information requirements also become increasingly non-trivial and coding accuracy can be seriously affected. A further disadvantage associated with all forward adaptive schemes is of course delay.

Fixed bit allocation, if used together with backward adaptive quantization offers a distinct advantage in terms of available bits for coding the sub-band signals (as no side information is required), and a reduction in coder delay. Unfortunately however, as discussed previously, the inability to track the short-term frequency variations in the input signal imposes a severe limit to performance, especially with a large number of bands. Also in such cases, a significant proportion of available bits are tied up by the high frequency bands (to prevent loss of bandwidth) leading to a reduction in overall coding efficiency.

Backward adaptive bit allocation with backward quantization, which offers the promise of dynamic assignment of bits without the need for side information is an attractive proposition. The bit allocation can be made to vary according to the relative energy composition of previously decoded sub-band samples. Unfortunately, although
theoretically possible, most conceivable forms of backward bit allocation adaptation would be extremely sensitive to transmission errors. Once the bit allocation pattern in the receiver is not matched to that at the transmitter, the system collapses unless some form of recovery is incorporated (which inevitably means more complexity and loss of performance).

Another possible combination is to employ forward adaptive bit allocation with backward quantization. In this case, the adaptive bit allocation process is performed at the transmitter and the bit allocation map is communicated to the receiver. This method would retain the advantage of optimum bit allocation, with reduced side information and lower receiver complexity (as the bit allocation procedure need not be repeated at receiver). The reduction in side information arises because, unlike signal variances which must be fairly accurately quantized, the information concerning the bit allocation pattern can only take on a very limited range of integer values, and thus can be transmitted with a smaller number of bits.

Figure 6.5 shows an example, the histogram for the number of bits assigned to each sub-band for an 8 band SBC, with the bit allocation updated every 256 samples (32 msec). It can be seen that generally, the bit information for the lower sub-bands of the signal can be coded with 2 bits (4 possible values) whilst the same information related to higher part of the spectrum requires no more than 1 bit. This provides a saving of 3 to 4 bits for each band, compared to the case where the average energy of each band is coded with 5 bit accuracy. The saving is substantial when the spectral resolution is high, as in the 32 band case, where the increased side information can seriously impair coding efficiency. This method of transmitting the bit allocation map may be considered as a simple form of vector quantization (see chapter 5), where the code-book contains a set of all bit allocation patterns of practical interest, and a codeword is transmitted once per block of samples to indicate which pattern is to be used.

A potential problem exists with the use of instantaneous backward adaptive quantizers (AQB) with adaptive bit allocation. The adaptation algorithm of AQB requires a minimum of 2 bits to allow the step size to adapt to the magnitude variations of the quantizer input but the high frequency bands are often only assigned one bit. One method to overcome this difficulty uses the \( \frac{1}{k} \) bit quantizer, where the sign information is transmitted with one bit every sampling instant, while the magnitude is encoded with an additional bit every \( k \) samples. Another method of adapting 1 bit AQB was discussed in section 6.2.4.1.2 where an approximation for the magnitude of the 1 bit AQB output is
Figure 6.5: Probability histogram of bit allocation to various bands in an 8 band SBC.
obtained from a suitably scaled versions of the output of one of the lower bands. The actual ratios for scaling depend on the energy in the reference band (which would be indicated by the number of bits assigned to it) and can be optimized from long-term statistics. Using this technique, the important zero crossing (sign) information is preserved for these high frequency bands and the magnitude follow a scaled down version of the signal envelope of the lower bands.

Hybrid methods of quantization may also be used, where some bands (especially the 1-bit high frequency bands) are coded with AQF, while the lower bands use AQB, the particular design chosen would obviously depend on the enviornment and application.

6.2.5.2 Parallel Bit Allocation

Typically about half the delay incurred by the SBC is due to the use of forward adaptive bit allocation and quantization, since the adaptation is based on the outputs of the filter bank. While this delay may be avoided by employing fixed bit allocation with AQB, the resultant degradation in performance is unfortunately far from acceptable.

This delay may be reduced by attempting the bit allocation for the sub-band signals during the necessary time delay incurred in the filter bank. For an 8 or 16 band SBC, the delay due to the tree structure QMF analysis bank is typically about 15 to 30 msec (depends on the filter length), which is a suitably long time for bit allocation and quantizer adaptation.

This parallel bit allocation can be carried out by performing a spectral analysis on the input signal segment, while it is propagating through the analysis filter bank. One way to do this is by using the discrete Fourier transform (DFT). The short-time Fourier spectrum of the input speech segment provides an estimate of the energy distribution in the various frequency bands. The accuracy of estimation might not be sufficiently high to permit the use of AQF (with transmission of step sizes) although, the relative energy composition of the various bands should be adequate to provide a bit allocation pattern (for use with AQB) which reflects the dynamic spectral variations of the input signal.

6.3 16 Kbps Transform Coder

Like the sub-band coder seen in section 6.2 a transform coder design can be divided into three sections. These are again the band splitting, quantization, bit allocation and
noise shaping. Band splitting is usually performed by discrete cosine transform (DCT) which makes it possible to have much more finer frequency bands than a typical 16 band SBC. Bit allocation and noise shaping used for the SBC in section 6.2 is directly applicable to a transform coder. The only difference is that in an SBC band energies are used to allocate bits and in a transform coder the average of certain groups of frequency coefficients are used. Ideally, frequency coefficients should be used separately in the bit allocation process. However, as seen in SBC, increase in side information does not permit this. The vocoder driven transform coder [13], however, makes it more possible by finding an estimate of each coefficient and using it in the bit allocation process. The efficiency of this scheme, of course, depends on the accuracy of the estimation of the coefficients. Quantization of frequency coefficients is usually done by APCM-AQF, which is again the same as for SBC.

6.3.1 Simulations

As we are concerned with evaluating the performance of speech coders operating at 16 K/s, both Zelinsky and Noll's [11], and the complicated vocoder driven adaptive strategy [13] were simulated on the computer. We then designed and simulated a new transform coder which we called pitch driven transform coder.

6.3.1.1 Zelinsky And Noll’s Approach

A 128-point DCT was used to perform the block transformation. The basis spectrum was estimated using 16 uniformly spaced support values, each obtained by averaging over 8 neighbouring transform coefficients. For example, the first support value obtained from the average variance of the first 8 coefficients was positioned at location 4, the next at location 12, then 20 and so on until location 124. Average energy of these 16 support values was quantized using the 7 bit non-uniform quantizer of SBC with initial step size of 22 and the $M$ value of 1.009. Quantized average energy was then used to divide each support value. These support values were then quantized using 3 bits for each. Base 2 logarithmic values were taken of these support values before quantization to ensure a more uniform amplitude distribution. The bit allocation procedure using this spectral estimate was performed in the same way as for the sub-band coder. The number of bits assigned to each frequency component were rounded to the nearest integer. Excess bits were taken from the least deserving coefficients and extra bits were given to the most deserving cases in the same manner as before. With 7 bits used for coding the block
standard deviation and 42 bits for the 14 support values (14 because the last two represent the theoretically non-existent coefficients, i.e., beyond the cut-off frequency of 3.3 KHz), a total of 207 bits per block of 128 samples were available for distributing among the transform coefficients.

Using the SBC's AQF step sizes given in Table 6.12, and limiting the maximum bit per coefficient to 5, the coder was simulated and its objective and subjective performance was evaluated. In the simulation mixture of male and female speech with some silences were used. Signal to noise ratio of the 14 spectral regions as well as the overall SNR can be seen in Table 6.15. Regional SNRs and the total segmental SNR were about 2.5 dB less than the 13 band SBC. This was not surprising, since the side information update rate of ATC was twice that of SBC. Also ATC transmits 7000 coefficients whereas SBC on the other hand uses 13 bands and transmits 6500 samples per second. This relatively lower SNR performance of the ATC however, was not noticeable in the informal subjective listening comparison.

Subjective quality of the recovered speech was extremely good for the male speech, where distortion was barely perceptible. For the female speech however, a slight buzz could be heard in the background, due possibly to edge effects related to the use of block transforms.

Signal to noise ratio of ATC can be improved by reducing the side information. This can be done simply by updating the side information every 256 rather than 128 samples. The edge effects seen in female speech can be reduced by the use of larger size transform, e.g., 256-point. However during the voiced segments this causes large quantization errors. A 256-point DCT produces clear pitch harmonics and makes it difficult to quantize without considering the pitch measurement. The other reason for the background noise during the female speech when using 128-point DCT is that female speech usually contains more pitch pulses in a given block of 128 samples than the male speech, making it more possible to have clearly defined pitch harmonics in the spectrum. This of course reduces the quantization efficiency (quantizers were designed for a random Gaussian approximated signal). Figure 6.6 shows the 128-point DCT transformed male and female speech, where female pitch harmonics are clearly seen.
Figure 6.6: Spectral comparison of typical (a) male, (b) female speech when transformed using 128-point DCT.
<table>
<thead>
<tr>
<th>Region</th>
<th>Bandwidth (Hz)</th>
<th>SNR (dB)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0-250</td>
<td>13.60</td>
</tr>
<tr>
<td>2</td>
<td>250-500</td>
<td>20.40</td>
</tr>
<tr>
<td>3</td>
<td>500-750</td>
<td>20.51</td>
</tr>
<tr>
<td>4</td>
<td>750-1000</td>
<td>16.40</td>
</tr>
<tr>
<td>5</td>
<td>1000-1250</td>
<td>16.10</td>
</tr>
<tr>
<td>6</td>
<td>1250-1500</td>
<td>14.91</td>
</tr>
<tr>
<td>7</td>
<td>1500-1750</td>
<td>14.80</td>
</tr>
<tr>
<td>8</td>
<td>1750-2000</td>
<td>13.34</td>
</tr>
<tr>
<td>9</td>
<td>2000-2250</td>
<td>15.72</td>
</tr>
<tr>
<td>10</td>
<td>2250-2500</td>
<td>15.54</td>
</tr>
<tr>
<td>11</td>
<td>2500-2750</td>
<td>14.36</td>
</tr>
<tr>
<td>12</td>
<td>2750-3000</td>
<td>13.83</td>
</tr>
<tr>
<td>13</td>
<td>3000-3250</td>
<td>14.44</td>
</tr>
<tr>
<td>14</td>
<td>3250-3500</td>
<td>10.50</td>
</tr>
</tbody>
</table>

Total segmental SNR = 18.68 dB.

Table 6.15: SNR performance of ATC with Zelinsky and Noll's adaptation.

6.3.1.2 Vocoder Driven Approach

The vocoder driven transform coder differs from Zelinsky and Noll's approach in two ways. Firstly, bit allocation is performed by the use of an estimated spectrum, making it possible to allocate bits per coefficient bases rather than per group of coefficients bases as in Zelinsky and Noll's. Secondly, the complete spectrum is normalized using this estimated spectrum, rather than the average support values. Therefore, the estimated spectrum should closely resemble the original spectrum, if any improvement is to be
achieved over Zelinsky and Noll's approach.

A 256-point DCT was used to perform the block transformation. In the time domain, 12 LPC coefficients together with the pitch period and single tap pitch gain was calculated. Using 256-point DFT, impulse response of the LPC filter was transformed to obtain an estimate of the spectral envelope. Pitch period and gain were used to create the estimate pitch harmonics which were then multiplied with the estimated envelope to obtain the final spectrum. Estimated and original spectra were normalized to unit variance before dividing for quantization. A total of 50 bits were used to code the 12 LPC parameters which were 6,6,5,5,4,4,4,4,3,3,3,3 for coefficient 1 to 12 respectively. The single tap pitch gain and the pitch period were coded with 3 and 7 bits respectively. Also the same 7 bit non-uniform quantizer used for coding the block energy in SBC and Zelinsky and Noll's was used to code the block energy which was used to normalize the spectrum before dividing it by the unit variance estimated spectrum. The bit allocation for the speech parameters produced 2094 bits/sec side information leaving 445 bits to be distributed for coding the coefficients per every 256 samples.

Segmental SNR was observed to be about 1 dB higher at 19.55 dB than for the Zelinsky and Noll approach. Subjective quality of both male and female sentences were close to the original quality. Background noise seen in the Zelinsky and Noll system during female speech, was reduced to a level which was hardly noticeable. During male speech no difference between the Zelinsky and Noll's and vocoder driven approach was noticed.

Although, the vocoder driven transform coder may in general be preferred to the Zelinsky and Noll's approach, its complexity is more than twice that of the Zelinsky and Noll's. It requires, the LPC, pitch and pitch gain calculations as extra information. In order to create the estimated spectrum it uses a 256-point DFT and hence is forced to double the size of the block transform to 256-point DCT. Its final disadvantage is that if the modeling of the estimated spectrum fails, the coder fails to produce good results. For example, if the pitch period is incorrectly measured, then the bit allocation process fails to allocate bits to pitch harmonics in the spectrum. Also when estimated and original spectrums are divided the resulting residual is no longer unit variance which causes large quantization errors. In order to minimize these unnecessary quantization errors the resulting residual signal should be normalized to unit variance before quantization. This of course means more complexity and extra side information (2 to 3 bits per 256 samples).
6.3.1.3 Pitch Driven Transform Coder (PDTC)

The presence of leakage between frequency bands can affect the performance of a frequency domain coder in two ways. Firstly if a particular frequency band is low in energy compared to other bands, the energy leaked from the other bands can represent a significant portion of the energy in that band. This leakage can interfere with the ability of the coder to take full advantage of the true spectrum of the signal in that band. Secondly, after encoding of the bands, the leakage, or aliasing, from one band to another is not entirely cancelled in the synthesis. Therefore, interband leakage in the analysis stage of a frequency domain coder can lead to undesirable effects of frequency domain aliasing in the synthesis. The effects of frequency domain aliasing generally becomes more pronounced as the dynamic range of the spectrum of the signal being analysed becomes larger. One way of controlling this aliasing is by increasing the size of the analysis window (Transform size in ATC and filter impulse response for SBC). Another method of controlling frequency domain aliasing is by reducing the dynamic range of the spectrum by pre-emphasis or spectral flattening prior to the analysis/synthesis. In this way the leakage from large energy bands to low energy bands is reduced.

It has been reported in the literature [4] that pre-emphasizing the speech before block transformation helps to reduce the block edge effects and hence the slight back ground noise seen in female speech. However, these coders still require 256-point DFT for the calculation of the estimated envelope and hence require 256-point DCT for block transformation of speech. Therefore, although the speech quality may be improved with the use of pre-emphasis, coder complexity is still too high to be acceptable.

Here, we describe a new transform coder called pitch drive transform coder (PDTC) which is a simplified version of the vocoder driven approach. A block diagram of PDTC is shown in Figure 6.7. As in vocoder driven approach 12 LPC parameters are calculated every 256 samples and quantized using 50 bits. However, rather than using them in an DFT transformation to get the formant structure, here LPC parameters are used to inverse filter (pre-emphasize) the speech, to remove formants and flatten the spectrum. The remaining LPC residual is then block transformed using a 256-point DCT. This creates a flat spectrum with well defined pitch harmonics if the segment is voiced speech. It is important to preserve pitch harmonics (excitation pulses) in speech if high quality is desired. Therefore, using the pitch period and single tap pitch gain an estimate of the residual spectrum is obtained. Using this spectral estimate pitch harmonics are removed by dividing into the residual spectrum. In order to maximize the signal to quantization
Figure 6.7: A block diagram of PDTC (a) encoder, (b) decoder.
noise ratio of the residual spectrum estimated coefficients are used to allocate bits, allocating more bits to the pitch harmonics than the coefficients in between the harmonics. During the bit allocation the maximum and the minimum bits per coefficient are limited to 4 and 1 respectively. Bits are distributed using the divide by 2 and allocate one bit procedure.

A 5 bit non-uniform quantizer with the initial step size of 19 and $M$ value of 1.071 was used to quantize the energy in the LPC residual. In addition to the 5 bits, an extra 2 bits were allocated to a uniform quantizer to quantize the divided residual spectrum block energy. This was done to make sure that the divided spectrum always had unit energy before quantization. This of course also reduces the errors introduced by division when there were pitch errors (if there is a pitch harmonic mismatch between the original and the estimated spectrums, then when dividing into one another the resulting residual may not be unit variance, causing clipping). Total side information per 256 samples was 67 bits leaving 445 bits to quantize the 256 residual coefficients. Overall segmental SNR obtained was lower than both vocoder driven and Zelinsky and Noll's approaches at 17.85 dB. However, subjectively it was at least as good as the vocoder driven approach. Back ground noise and the block edge effects were completely eliminated. Memory of the LPC synthesis filter being carried over from one block to the next helped to interpolate the blocks and smooth out the block edge effects. Also simple pitch driven bit allocation and quantization made it possible to quantize the LPC residual as well as the vocoder driven approach. However, the pitch driven coder did not require DFT computation, thus, reducing the computation by a large amount (approximately by a factor of 2). Also the simple divide by 2 and allocate 1 bit, bit allocation procedure replaced the complex logarithmic bit allocation procedure, resulting in a moderate complexity high quality 16 Kb/s transform coder.

6.4 Discussions

The efficiency of frequency domain speech coding has been amply demonstrated by the sub-band and transform coders described above. Much of the superiority of such frequency domain coders over their time domain counterparts, lies in the effective exploitation of the non-flat spectral density of speech and the use of different encoding accuracy for different frequency regions. This flexibility ensures that the usefulness of every available bit is maximized.
Variations to the basic structure of the coders described, have been proposed by several researchers, but most of these involve very minor modifications. In sub-band coding, much of the more research effort have concentrated on simplifying the bit allocation process [5] and reducing side information by exploiting spatial redundancies in the signal energy [6]. Pitch prediction has also been incorporated in some systems [7][8], although the justification for this substantial additional complexity is dubious.

More recently, an attempt to bridge the gap between wide-band and narrow-band frequency domain coders came in the form of a 32 band sub-band coder [9], which uses vector quantization techniques for adapting the bit allocation and quantizer step sizes in order to minimize side information requirement. This highly complex scheme was reported to provide comparable quality with ATC schemes at the same bit rate. Obviously, the advantages of these powerful techniques over time domain methods have not been achieved without cost. Until the recent use of parallel filter banks in sub-band coders [14], the use of FIR tree structure QMF filter banks with their inherent delay had been a limiting factor in sub-band coders. This delay and the computational complexity of the analysis/synthesis filter bank processes increased proportionately with the number of bands.

The delay in the transform coder depends on the size of transform used which is usually sufficiently large to provide adequate frequency resolution. While this delay may generally be less than that of sub-band coder with AQF, the complexity of transform coders is much higher, since the encoding and bit allocation processes are effectively performed for a considerably larger number of frequency bands. Also the computation of block transformations usually requires more computation than a parallel filter bank. This complexity issue renders the otherwise powerful transform coder unattractive for many applications. A reduction in block transformation size may be a possible means of coder simplification. Unfortunately, the advantages of coding in the frequency domain also tends to be eroded when the transform size is decreased, and the resultant performance degradation far outweighs the reduction in complexity.

6.5 References
2. J.Max, "Quantization for minimum distortion", IRE Trans, on Information Theory, pp 7-12, March 1960.
CHAPTER 7

12 KB/S TO 9.6 KB/S CODERS

7.1 Introduction

Digital coding of speech at around 16 Kb/s is now well developed. The next bit rate of interest is at around 9.6 Kb/s. In chapter 6 we have shown that in order to reduce the bit rate from 32 Kb/s down to around 16 Kb/s something more than a simple waveform coder (APCM, ADPCM) is needed. Coders seen in chapter 6 combined the speech modelling process with a suitable waveform coder to enhance digital speech quality at 16 Kb/s. Similar procedures will be followed in developing algorithms to operate at even lower bit rates.

In order to reduce the bit rate down to about 9.6 Kb/s, reduction in both side information capacity and the actual speech signal (residual) capacity will have to be made. Although, reducing the side information will help in reducing the overall bit rate, typical reductions in side information are of the order of 1.5 Kb/s. Thus, in going down from 16 Kb/s to 9.6 Kb/s the capacity for coding the actual speech signals need to be reduced by about 5 Kb/s. This of course cannot be accomplished without reducing the quality of digital speech unless speech quality is improved by additional means to those used in 16 Kb/s coders. There are two major techniques that can reduce the bit rate from 16 Kb/s down to 9.6 Kb/s whilst maintaining high quality. These are base-band coding (BBC) and vector quantization (VQ), see chapters 4 and 5 respectively. In this chapter we will first explain how a typical SBC and ATC can be improved to operate at 9.6 Kb/s. We will then discuss various schemes of producing improved quality coders in the region of 12 to 9.6 Kb/s. In the design of 9.6 Kb/s coders, as well as producing high quality it is important to keep the delay and the complexity as low as possible. Therefore, whilst comparing the coder qualities additional comments will be made concerning their expected delay and complexity factors.

7.2 Sub-Band Coder

In order to demonstrate the effect of reducing the bit rate on the speech quality, an original sub-band coder which was designed to operate at 16 Kb/s was simulated at 9.6
Kb/s. Two simple capacity adjustments were made to fit the overall transmission into 9.6 Kb/s. These are shown in Table 7.1.

<table>
<thead>
<tr>
<th>Coder</th>
<th>16 Kbps SBC</th>
<th>9.6 Kbps SBC</th>
</tr>
</thead>
<tbody>
<tr>
<td>AQF Update (msec)</td>
<td>24</td>
<td>32</td>
</tr>
<tr>
<td>Side info (bps)</td>
<td>2500</td>
<td>1844</td>
</tr>
<tr>
<td>Max bit.per.band</td>
<td>5</td>
<td>4</td>
</tr>
<tr>
<td>Sub-band info (bps)</td>
<td>13500</td>
<td>8000</td>
</tr>
<tr>
<td>Total rate (bps)</td>
<td>16000</td>
<td>9844</td>
</tr>
</tbody>
</table>

Table 7.1: 16 and 9.8 Kb/s SBC specifications.

It can be seen from Table 7.1 that most of the reduction in bringing down the total bit rate from 16 to 9.6 Kb/s was made by reducing the sub-band information, ie. 13500 b/s to 8000 b/s, a reduction of 5.5 Kb/s. Reduction in the side information was only 656 b/s.

The SBC as outlined above was simulated. Due to shortage of bits to be allocated to the sub-bands, during the bit allocation procedure it was noticed that for most of the time (during active data) at least 4 of the 13 bands had zero bits allocated to them. The effect of this was clearly seen when the coder was tested by informal listening tests. Processed speech was significantly band limited at about 2.5 KHz (during the bit allocation process high frequency, low energy bands were usually allocated zero bits). This band limitation effect also introduced aliasing in the high frequency bands which was very annoying. In order to reduce the effect of band limitation and hence mask the aliasing noise, adaptive Gaussian noise generator was included in the decoder. The Gaussian noise generator was used in order to fill up the empty segments in each band according to their original energy contents. Although, some bands may be allocated zero bits, their energy contents are known at the decoder for the bit allocation process. The amount of noise power was determined by subjective listening tests using the expression.
where $N_i$ is the noise power (standard deviation) of the $i^{th}$ segment of a zero bit allocated band. $f$ is the scale factor which was determined by listening tests and $\sigma_i$ is the standard deviation of the $i^{th}$ segment.

$$N_i = f \sigma_i$$

<table>
<thead>
<tr>
<th>Band</th>
<th>SNR (dB)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>8.1</td>
</tr>
<tr>
<td>2</td>
<td>13.4</td>
</tr>
<tr>
<td>3</td>
<td>12.3</td>
</tr>
<tr>
<td>4</td>
<td>12.8</td>
</tr>
<tr>
<td>5</td>
<td>9.9</td>
</tr>
<tr>
<td>6</td>
<td>10.0</td>
</tr>
<tr>
<td>7</td>
<td>8.3</td>
</tr>
<tr>
<td>8</td>
<td>8.1</td>
</tr>
<tr>
<td>9</td>
<td>9.1</td>
</tr>
<tr>
<td>10</td>
<td>9.5</td>
</tr>
<tr>
<td>11</td>
<td>9.1</td>
</tr>
<tr>
<td>12</td>
<td>6.9</td>
</tr>
<tr>
<td>13</td>
<td>8.9</td>
</tr>
</tbody>
</table>

Total segmental SNR = 13.74 dB.

Table 7.2: SNR performance of a 9.8 Kb/s SBC.
Although adaptive noise substitution into the empty segments helped in reducing the band limitation effects and masked the aliasing noise, the quality of the speech was not as good as that of 16 Kb/s SBC. Quantization noise and some background noise was clearly audible. The performance of the coder was noticeably worse for female speech.

As there is no additional processing required to that for the 16 Kb/s coder the complexity is comparable to the 16 Kb/s case. In fact, because the bit allocation is performed less frequently (every 32 msec), 9.8 Kb/s SBC is less complex. However, the delay of the coder is increased from 24 msec to 32 msec.

7.2.1 SBC With Vector Quantized Side Information

It has been shown that 5.5 Kb/s reduction in the transmission of the sub-band signals is the major cause of speech quality degradation when going down from 16 Kb/s to around 9.6 Kb/s. The cut in the side information can only be accomplished in two ways: either by reducing the AQF quantizers update rate or by reducing the number of bands in the coder. As we discussed in chapter 3 and 6 the lesser number of bands in the coder the less the accuracy of spectral modelling and hence the reduction in the quality. Also in chapter 3 it was shown that the more often one updated the AQF quantizer the better it performed. These two factors demonstrate that the side information cannot be reduced by more than about 600 to 800 bits/sec, i.e., a minimum update rate of every 32 msec and minimum number of bands of at least 8. In order to reduce the side information and to allow more bits for sub-band signal coding whilst maintaining adequate number of bands in the coder and optimally updating the AQF quantizer, vector quantization (VQ) may be used to code the band energy vectors, see chapter 5.

7.2.1.1 Simulation Of The VQ Code-Books

As discussed in chapter 5 VQ is a form of block quantization where the single elements are grouped together and coded jointly as a single vector. The total bit rate needed for transmission is dependent on the rate of vectors, i.e., quantizer update rate, and the number of bits per vector, i.e., the accuracy of quantization.

In order to simulate the performance of VQ on the side information, i.e., band energies, band energies of an 8 band and 16 band coder (actually 7 and 14 bands used respectively) were calculated every 256 samples and stored as a large block of training data. Using this data base 3 VQ's were designed for the 8 and 16 band SBC band energies.
These were the binary tree search, optimized full search, and randomly chosen full search.

**Binary Tree Search VQ**

As discussed in chapter 5, binary tree search code-book design requires very heavy computation. In order to reduce the load on the computer at any one time, the complete code-book was designed in four separate blocks. Rather than starting with 2 initial vectors and splitting them iteratively until a desirable distortion limit was met, we started with four vectors, each of which was treated as an initial vector for the four segments of the training data. Here four initial vectors were chosen in such a way that each represented almost equal number of training vectors. During the optimization of vectors at each node of the tree, a mean squared error distortion measure was assumed. Therefore, the calculation of the centroids of the clusters were the averaged sum of each vector in the clusters.

\[ C(k) = \frac{1}{N} \sum_{\ell=1}^{N} V_\ell(k) \quad \text{for} \quad k = 1, 2, \ldots, b. \]  

(7.2)

In equation (7.2), \( k \) represents the vector elements which are the band energies and \( b \) represents the number of vectors in each cluster.

An optimization procedure which involves calculating the centroids and then choosing new clusters can continue as long as specified. During tests we found that 4 or 5 iterations were adequate. Iterations beyond 5 did not contribute much to the overall distortion minimization. After optimization of each node vector, the vectors were then split into two, by using the following expressions,

\[ V_{e_1}(k) = V_p(k) + \epsilon_k \]

\[ V_{e_2}(k) = V_p(k) - \epsilon_k \]

(7.3)

Typical \( \epsilon_k \) values are given in Table 7.3. Whilst splitting it is important to check that each branch has a direction towards the densely populated space. In order to achieve this
condition, after splitting the vectors, those that have only one or two vectors in their clusters were further split by the \( \epsilon_k \) factors given in Table 7.4, until their direction was found to be towards the densely populated space.

<table>
<thead>
<tr>
<th>( k )</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \epsilon_k )</td>
<td>0.04</td>
<td>0.03</td>
<td>0.02</td>
<td>0.03</td>
<td>0.01</td>
<td>0.005</td>
<td>0.005</td>
</tr>
</tbody>
</table>

Table 7.3: Vector splitting factors for 7 band SBC band energies.

<table>
<thead>
<tr>
<th>( k )</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \epsilon_k )</td>
<td>0.004</td>
<td>0.003</td>
<td>0.002</td>
<td>0.003</td>
<td>0.001</td>
<td>0.0005</td>
<td>0.0005</td>
</tr>
</tbody>
</table>

Table 7.4: Secondary vector splitting factors for 7 band SBC band energies (used when Table 7.3 is unsuccessful).

NB: \( \epsilon_k \) values in Table 7.3 and 7.4 are the same as for the 14 band case. Each value is used for the corresponding two bands energies in the 14 element vector. The results are shown in Figure 7.1.

During various simulation tests it was noticed that the presentation of results in Figure 7.1 should really have another dimension. This was the rate that the vectors were calculated. Results including the vector update variation are thus shown in Figure 7.2. Results in Figure 7.1 and 7.2 were calculated using the training data. However, as the number of training and independent test data vectors became large the performance of the code-books for both the training and independent data approached each other as shown in Figure 7.3.

**Optimized Full Search VQ**

In a binary tree search VQ, at every node, the input vector was compared with two
Figure 7.1: MSE comparison of various size tree search code-books for (a) 14 band and (b) 7 band SBC energies (vector update rate every 256 samples).
Figure 7.2: MSE comparison of various size tree search code-books with respect to vector update rate, (a) 256, (b) 128, (c) 64, (d) 32, (e) 16 samples.
code-book vectors until the final stage was reached. In a full search VQ we started at the final stage and searched through all vectors to select the one with minimum mean squared error. Initial vectors of a full search code-book are chosen from the training data at random intervals. During the optimization process it was noticed that different initial code-books produced different overall distortions. It was also noticed that some vectors in the code-book did not contribute to the overall distortion minimization. This was due to the fact that when vectors were selected randomly from the training data some were chosen from the spaces which had only few vectors. Thus, it was necessary to ensure that the code-book chosen initially was as good as possible, and also to ensure that all of the code-book entries were useful. We found it very important to start with a larger code-book than the optimum size. The optimization process is as follows.

(i) Start with a larger code-book, about 1.5 times the optimum (maximum allowed by the overall bit rate constraint).

(ii) Quantize the training data.

(iii) Optimize the code-book by calculating the centroids of the clusters.

(iv) Is the size of the code-book optimum? If yes stop, else continue.

(v) Ignore some of the excess vectors by discarding the least used vectors and goto (ii).

At every stage the amount of vectors discarded can be varied and for the termination of the process any other condition may be used, i.e., overall distortion rather than the size of the code-book.

As was expected, full search VQ produced better results than the tree search VQ. It was also found that the decrease in distortion for each element of the vectors was different. Distortion in the large energy points of the full searched and the tree searched vectors was very similar. However, full search performed better over the vector elements with small energy.

**Random (non-optimized) Full Search VQ**

In this case various randomly chosen code-books were compared and the best performer was chosen as the non-optimized full search code-book. When the results were compared with optimized full search and the tree search cases it was concluded that when the size of the code-book was large, then randomly chosen code-books perform as good as any other code-book, see Figure 7.4. Similar results were obtained for the 14 band SBC.
Figure 7.3: Performance difference of a code-book when tested using data from inside and outside training data.

Figure 7.4: Performance comparison of (a) full search, (b) tree search and (c) randomly selected full search code-books.
7.2.1.2 Simulation Of 8 And 16 Band SBCs With Vector Quantized Side Information

Using 8 bit full search (optimized) code-books for both 8 and 16 band (only 7 and 14 used respectively) SBC coders were simulated.

8 Band SBC

The specific parameters of the 8 band SBC are as follows.

- Quantizer Update: every 256 samples (every 32 msec).
- Max bit per band: 4.
- Bits per analysis block: 9.
- Sub-band info bits/sec: \((9/7) \times 7000 = 9000\).
- Total bit rate bits/sec: 9000 + 437.5 = 9438.

As can be seen from the above figures, reduction in the side information of about 65% has been achieved by the use of VQ to quantize band energies. This yields an extra bit per block to be allocated to the sub-bands. Reduction in the side information can be more significant if the quantizer update rate is increased.

An 8 band SBC with vector quantized side information with only 7 bands transmitted was simulated. Speech quality at 9.438 Kb/s with 7 bands was much better than that seen earlier in section 7.3 with a bit rate of 9.84 Kb/s and 13 bands. The reason is that the 9.84 Kb/s, 13 band coder had 1.193 bits/sample for its sub-band signal whereas the 7 band coder had 1.286 bits/sample allocated for this purpose. Although, the test data was filtered at 3.3 KHz, allowing the transmission of a 7 band, 3.5 KHz signal as opposed to 3.25 KHz in the 13 band case, produced a wider band signal which sounded more pleasant. The overall segmental signal to noise ratio was also higher by 1.26 dB. Higher segmental signal to noise ratio can be obtained by transmitting only 6 bands. However, this does not improve the subjective quality. SBC with 6 bands sounds band limited and looses its sharpness.

16 Band SBC

Using the same specifications as for the 8 band coder but transmitting 14 bands, another coder was simulated. Vector quantization of the band energies was performed by an 8 bit, 14 element code-book. Although, it was expected that the 14 band coder would produce better quality coded speech than the 7 band coder, results actually contradicted
Coded speech produced large quantization noise in the high frequency bands which was very annoying. When analysed, it was seen that high frequency band signals had large clipping errors. In some blocks, the clipping effect was so large that the block signal to noise ratio was negative. In other words clipping errors were larger than the signal itself. This was due to poor vector quantization of the 14 sub-band energies in one vector. The 8 bit code-book (full search) had a -11/-12 dB distortion performance which seemed to be adequate. However, when the code-book was searched, low frequency, high energy bands dominated the distortion minimization. Therefore, high frequency bands, or the bands which had small energy contents, had large errors which in turn caused errors in APCM step sizes, thus producing large clipping errors. In order to prevent this, 14 element vectors were split into two 7 element vectors. In order to produce unit variance vectors each of two 7 element vectors were normalized by their energy sums. These energy sums were coded each with 5 bits. However, as the two vector elements were normalized with different energy contents, bit allocation to the sub-bands had to be modified to take these energy variations into account. In every block, two stages of bit allocation were performed. Firstly, bits were allocated or shared between the two vectors by comparing their total energy; \( \hat{\sigma}_1 \) and \( \hat{\sigma}_2 \) (\( \hat{\sigma}_1 \) and \( \hat{\sigma}_2 \) are quantized values of \( \sigma_1 \) and \( \sigma_2 \)). Secondly, shared bits were then allocated amongst the vector elements. The 14 band coder was then tested again with these modifications. During coding, sub-band adaptive noise substitution to the empty segments (zero bit allocated) was employed, with an \( f \) factor of 0.6.

Overall segmental SNR was about 0.59 dB higher than that for the 7 band coder. This was due to better spectral modelling via the 14 bands. Larger SNR could be achieved if the side information was kept as low as that for the 7 band coder. The 7 band coder had 437.5 bits/sec side information, but 14 band coder had 750 bits/sec side information ((8000/256)*(7+7+10)). However, as was discussed for the 7 band coder, higher SNR does not necessarily mean higher subjective quality. This was the case with these coders. Although, the SNR difference between the 7 band and the 14 band coders was only 0.59 dB, subjective quality was considerably more improved than was reflected by 0.59 dB. The 14 band coded speech very clean and pleasant. Aliasing or high frequency distortion was hardly noticeable. Slight quantization noise was heard when using high quality listening equipment. Table 7.5 shows band signal to noise ratios as well as the overall SNR.
Theoretical delays for both the 7 band and the 14 band coders were 32 msec. However, as the results show in Figure 7.2, the quantizer may be updated every 16 msec with no significant degradation in the performance of VQ on the side information. This will double the side information (which is fairly small), causing minor degradations in the sub-bands. The quantizer update rate also determines the number of times the codebooks are searched, requiring 7 multiply-add operations per level of the code-book. For
the 14 band coder, two 7 bit code-books were searched requiring 14 multiply-add operations per level of the code-book. Assuming 7 bit full search code-books for 7 and 14 band coders 896 and 1792 multiply-add operations are required respectively per quantizer update. The rest of the coders complexity is equivalent to 7 and 14 band coders with no VQ.

Better performance in quantizing the side information of the 14 band coder may be achieved by using two 7 bit cascaded 14 element code-books. However, this will require twice the computation of two 7 element, 7 bit code-books.

**7.2.2 Fully Vector Quantized SBC**

The idea of fully vector quantization of a sub-band coder was first proposed by A.Gersho and others in 1984 [1]. The reported coder operated at 16 Kb/s with 8 bands. Although, promising to report on a similar 16 band coder, no paper has currently been published to date. The 8 band coder employed similar vector quantization on the side information as was discussed in section 7.2.1. Scalar quantizers (APCM) were replaced by various size code-books in the sub-bands. Bit allocation procedure determined the size of the code-books to be used in each sub-band. Therefore, each band produced one or more vectors per block depending upon the block length, independent of other bands. This we call serial vector quantization, meaning that each band is quantized separately in series. Here, in the following section we discuss the prospects of the serial VQ for lower bit rates than 16 Kb/s and report on a different VQ for SBC which we name 'parallel VQ'.

**7.2.2.1 Serial Vector Quantization**

In serial VQ each band is normalized with its band energy. Normalized sub-band signals are then split into a number of vectors.

\[ n_v = \frac{N}{b \cdot n_l} \]  

(7.4)

where \( n_v \) and \( n_l \) are the number of vectors in each band per block, and the number of elements in each vector respectively. \( N \) is the analysis block length and \( b \) is the number of bands. (\( n_l \) can be different in each band). For simplicity we will assume that \( n_l \) is the same for all bands. Most of the SBC coder output is usually occupied by the transmission of the sub-band signals, therefore, when deciding on \( n_l \), hence \( n_v \) and the length of the
code-books overall bit rate should be considered. Also important is the ratio of $\frac{N}{b \ n_l}$. If $b \ n_l = N$, then each band has only one unit variance vectors per analysis block. However, if $N > b \ n_l$ then each band will have more than one non-unit variance vector, which means further bits are necessary to normalize each vector to unit variance.

An 8 band (only 7 used) SBC with vector quantized side information and serial vector quantized sub-band signals was simulated with the following parameters.

Number of bands : 7.
Analysis length : 32 msec (256 samples).
$n_l$ : 8.
$n_v$ : 4.
Length of code-books : 64, 128, 256, 512, 1024, 2048.
Vector energy : 3 bits.
Side information : $2625 + 438 = 3063$ bits/sec.
Bits allocated per block : 52.

(for 9.6 Kb/s overall transmission, there are $9600 - 3063 = 6537$ bits/sec to code the sub-band signals, ie. $0.9338$ bits/sample. Because each vector has 8 samples and there are 7 bands total bits per block is $8 \times 7 \times 0.9338 = 52$).

During the bit allocation, because minimum code-book size was 6 bits, each band was allocated with 6 bits and the remaining 10 bits were distributed via the usual method of bit allocation. The subjective quality of coded speech was poor compared to that with no VQ, or VQ on the side information only. High quality speech could only be produced at around 12 Kb/s, i.e. when 1.276 bits per sample was used, 71 as opposed to 52 bits per block. This demonstrates that if high quality is to be achieved, high energy bands should be quantized with 11 bit code-books and the low energy bands should be quantized with 9 or 10 bit code-books.

Code-books were searched to find the vector which minimized the mean squared error.
\[ E_r = \sum_{i=1}^{nl} [X(i) - V(i)]^2 \]  

where \( X(i) \) is the input vector and \( V(i) \) is the code-book sequence.

Since \( nl \) is 8 then the sign combination needed is \( 2^8 = 256 \). Therefore, during quantization 256 levels are needed to provide the right sign combination for each vector. A 10 bit code-book provides 4 amplitude variations with the correct sign bits. This was one of the reasons that large code-books are needed to produce high quality speech. The second reason for the poor quality of SBC with serial VQ is that the number of bands was 8 and hence each band had a bandwidth larger than the pitch frequency. When the pitch frequency is smaller than the sub-band bandwidth each band shows pitch structure as shown in Figure 7.5. This means that code-books for sub-bands should be designed carefully to take this pitch structure into account. (In the simulation random Gaussian code-books were used). However, as the range of pitch frequency varies from about 50 to 400 Hz it is extremely difficult to design small code-books to produce high quality, unless the coder is designed for a restricted range of talkers.

Assuming that the code-books are searched without introducing any delay, SBC with serial VQ has 32 msec delay. As in SBC with VQ on the side information, delay can be halved by halving the analysis block length from 256 to 128 samples. However, as was discussed earlier, this will double the side information VQ complexity. It will also double the side information and may cause minor degradations in speech quality.

Major complexity of the coder is introduced by vector quantizers for the sub-band signals. Code-books with 10 and 11 bits (full search) require 8192 and 16384 multiply-add operations respectively for each input vector. Since the size of input vector is 8, then the above figures are needed per millisecond. The complexity of these vector quantizers may be reduced by using cascaded quantizers or by splitting the 8 element vector into smaller vectors and allocating smaller code-books. However, all of these require either more side information, or introduce degradations into the overall subjective quality of the coder. Each band should ideally use different code-books which are optimized using that bands signal. Also during coding, various bands will be coded with different code-book sizes. Therefore, under channel errors the decoder may lose synchronization with the encoder (if bit allocation is performed wrongly).
Figure 7.5: Waveform of decimated sub-band signal which still have the pitch structure.
7.2.2.2 Parallel Vector Quantization

Here, we explain a new vector quantization scheme for sub-band coders which we call parallel vector quantization. Parallel vector quantization eliminates the disadvantages of serial vector quantization. It also allows automatic bit allocation without the danger of lost synchronisation between the encoder and the decoder. Parallel VQ does not require specially designed (trained) code-books for each band.

As the name indicates in parallel VQ all bands are simultaneously vector quantized in parallel using a single fixed code-book.

\[
X_k(i) = b_k(k) \quad (7.6)
\]

\[i = 1, 2, ..., b, \quad k = 1, 2, ..., \frac{N}{b}.
\]

where \(X_k(i)\) is the \(i^{th}\) element of the \(k^{th}\) vector, and \(b_k(k)\) represents samples taken from each band in sequence. As with serial VQ, in parallel VQ each band signal is normalized by its band energy. However when the vector is made up of samples across the bands, it may not be unit variance and hence it should be normalized to enable unit variance code-book comparison. During the search of the code-book, mean squared error between the sub-band vectors and the code-book entries is minimized using equation (7.5). Using equation (7.5) as stated produces quantized outputs with equal bits in each band. In order to provide bit allocation and hence enable noise shaping we have modified equation (7.5) as follows.

\[
E_r = \sum_{i=1}^{b} \left[ \sigma^p(i)(X(i) - V(i)) \right]^2 \quad (7.7)
\]

In equation (7.7), \(b\) is the number of bands, which is also the number of elements in each vector. \(\sigma(i)\) is the energy in the \(i^{th}\) band and \(p\) is the noise shaping factor chosen by subjective listening tests, which lies within \(0 \leq p \leq 1\). Figure 7.6 shows the effect of various \(p\) values on the quantization noise spectrum. Parallel vector quantized SBC was simulated with the parameters given in Table 7.6.
Figure 7.6: Noise shaping in speech. (a) maximum noise shaping \((p=0)\), (b) 50% noise shaping \((p=0.5)\) and (c) no noise shaping \((p=1)\).
Table 7.6: Parallel VQ-SBC parameters.

<table>
<thead>
<tr>
<th>Bands</th>
<th>8</th>
<th>16</th>
</tr>
</thead>
<tbody>
<tr>
<td>Analysis block (msec)</td>
<td>32</td>
<td>32</td>
</tr>
<tr>
<td>Bands Used</td>
<td>7</td>
<td>14</td>
</tr>
<tr>
<td>Vector gain (bits)</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>Side information (bps)</td>
<td>3000+438</td>
<td>1500+750</td>
</tr>
<tr>
<td>Code-book size (bits)</td>
<td>8,10</td>
<td>8,10</td>
</tr>
<tr>
<td>Bit rate (bps)</td>
<td>11438,13438</td>
<td>6250,7250</td>
</tr>
</tbody>
</table>

Results of signal to noise ratios are tabulated in Table 7.7.

<table>
<thead>
<tr>
<th>Bands</th>
<th>7</th>
<th>14</th>
</tr>
</thead>
<tbody>
<tr>
<td>8-bit SNR (dB)</td>
<td>10.07</td>
<td>5.048</td>
</tr>
<tr>
<td>10-bit SNR (dB)</td>
<td>12.261</td>
<td>6.782</td>
</tr>
</tbody>
</table>

Table 7.7: SNR performance of 8 and 16 band parallel vector quantized SBCs.

Subjective quality of a 13.438 Kb/s, 8 band coder was very similar to the original, however when the bit rate was reduced to below 11.438 Kb/s quantization noise could be heard. The 16 band coder's quality using 8 or 10 bit code-books was worse than the 8 band using 6 bit code-books. This was confirmed by the SNR results in Table 7.7. The 16 band coder had SNRs of approximately half that of the 8 band coder.

Parallel vector quantized SBC can operate in the region of 10 to 11 Kb/s and produce good quality, which is a little less than the serial vector quantized SBC. Delay of the parallel VQ-SBC is similar to the serial VQ-SBC. However, the complexity can be reduced. The code-book search can be limited to certain elements in order to chose a
number of possible candidate vectors and then apply full search to only those candidate
vectors. For an 8 band and hence 8 element vector, the search may be limited to the first
three elements which reduces the candidate vectors to about 32 or 64 and then these can
be searched for overall minimized distortion. The search can also be made adaptive. For
an 8 element vector, rather than assuming that the first 3 elements are the most impor-
tant, band energy comparison can be applied to pick the three most important elements.
In parallel VQ-SBC bit allocation is also eliminated making the coder simpler and more
robust under channel errors. The proposed coder also eliminates the need for trained
code-books, because during tests it was found that random Gaussian code-books per-
formed just as well as code-books populated by training vectors. This of course means
that the coder performance will also be robust for all speakers. One of the most impor-
tant advantages of the parallel over serial vector quantized SBC which has not been men-
tioned so far is the capability of the parallel VQ-SBC to operate in a variable transmis-
sion rate environment. Decreasing and increasing the areas searched in the sub-band
code-book will adjust the overall transmission rate with no other modifications. This is
possible because there is only one code-book for all bands and this code-book does not
require training.

7.2.2.3 Parallel VQ-SBC With Cross Correlation Error Minimization

In both serial and parallel VQ as seen so far, there is an implicit assumption, that
the size of the code-book is assumed to be large enough to cover all possible signal
sequences. Therefore, the expected noise power has always been assumed to be less than
the signal power. Before searching the code-book, every input sequence is normalized and
at the decoder the chosen unit variance sequence is multiplied by the original energy in
order to scale it.

\[ Y(i) = \sigma(S(i))V(i) \]  

\( Y(i) \) is the decoded vector \( \sigma(S(i)) \) is the energy in the input vector \( S(i) \) and \( V(i) \) is
the best match of \( X(i) \) in the code-book, chosen by minimizing \( E_s \) in equation (7.5), and
equation (7.7) for serial and parallel VQ respectively. Even if the assumption made
above is not valid according to equation (7.8) we still transmit. That is to say, if there is
not a good match for \( X(i) \) in the code-book then,
\[ d[X(i), V(i)] > d[X(i), 0] \] (7.9)

or,

\[ d[\sigma(\mathbf{S}(i)) X(i), \sigma(\mathbf{S}(i)) V(i)] > d[\sigma(\mathbf{S}(i)) X(i), \sigma_{opt} V(i)] \] (7.10)

where \( d(X(i), 0) \) represents the distance between \( X(i) \) and the zero element vector and \( \sigma_{opt} \) is the optimum scale factor.

Equations (7.9) and (7.10) show that if the assumption made above is not truly valid then we may transmit an excess amount of noise. In order to find \( \sigma_{opt} \) in which a minimum amount of noise power can be transmitted together with the signal, the expected noise power should be calculated [2][3].

\[ \sigma_{opt}^2 = MAX [E_s - E_n, 0] \] (7.11)

Equation (7.11) guarantees positive or zero signal to quantization noise ratio at all times, where \( E_s \) and \( E_n \) are signal and noise power respectively. Equation (7.11) also shows that provided that there is no quantization noise then \( \sigma_{opt} < \sigma(\mathbf{S}(i)) \). The best way of calculating \( \sigma_{opt} \) is to find the cross correlation of the input vector and the code-book entries.

\[ \sigma_{opt} = \sum_{i=1}^{N} X(i)V(i) \] (7.12)

If normalization is not used, equation (7.12) can be written as,

\[ \sigma_{opt} = \frac{\sum_{i=1}^{N} X(i)V(i)}{\sum_{i=1}^{N} V^2(i)} \] (7.13)

Equations (7.12) and (7.13) can be applied to serial vector quantized SBC, but for parallel VQ-SBC they need further modifications. In serial VQ each vector is made up of samples in each band which are then divided by the band energy and then the vector energy.
However in parallel VQ each element of the vector is divided by its corresponding band energy, and then the vector energy is used for normalization. Therefore, equation (7.13) takes the form,

\[ \sigma_{opt} = \frac{\sum_{i=1}^{b} \sigma(i)X(i)\sigma(i)V(i)}{\sum_{i=1}^{b} [\sigma(i)V(i)]^2} \]  

(7.14)

where \( \sigma(i) \) is the band energy of the \( i^{th} \) band or the \( i^{th} \) element of the vector. In equation (7.14) \( \sigma(i)X(i) \) is the original band signal \( S(i) \), which means no normalization is required, and is written as,

\[ \sigma_{opt} = \frac{\sum_{i=1}^{b} S(i)\sigma(i)V(i)}{\sum_{i=1}^{b} [\sigma(i)V(i)]^2} \]  

(7.15)

Equation (7.5) should also be modified to take account of \( \sigma_{opt} \).

\[ E_r = \sum_{i=1}^{b} [S(i) - \sigma_{opt}\sigma(i)V(i)]^2 \]  

(7.16)

As \( \sigma_{opt} \) will be different for each vector in the code-book, the search can be simplified if \( \sigma_{opt} \) for the vector which \( E_r \) is minimized can alone be calculated. In order to do this \( E_r \) should be calculated for each vector, but \( E_r \) cannot be calculated without \( \sigma_{opt} \). Therefore, in order to calculate minimum \( E_r \) with respect to \( \sigma_{opt} \) we set,

\[ \frac{\partial E_r}{\partial \sigma_{opt}} = 0 \]  

(7.17)

and obtain,
The sequence which maximizes the second term in equation (7.18) is chosen because it maximizes the correlation between the original and the quantized sequences. \( \sigma_{opt} \) for that sequence is then calculated using equation (7.15).

\[
E_r = \frac{\sum_{i=1}^{b} \sigma(i)V(i)S(i)^2}{\sum_{i=1}^{b} [\sigma(i)V(i)]^2}
\]

(7.18)

<table>
<thead>
<tr>
<th>Distortion</th>
<th>Simple-Matching (dB)</th>
<th>Cross-Correlation (dB)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Band</td>
<td>8-bit</td>
<td>10-bit</td>
</tr>
<tr>
<td>2</td>
<td>11.39</td>
<td>13.42</td>
</tr>
<tr>
<td>3</td>
<td>7.88</td>
<td>9.91</td>
</tr>
<tr>
<td>4</td>
<td>6.69</td>
<td>9.19</td>
</tr>
<tr>
<td>5</td>
<td>4.70</td>
<td>6.88</td>
</tr>
<tr>
<td>6</td>
<td>2.32</td>
<td>5.08</td>
</tr>
<tr>
<td>7</td>
<td>2.09</td>
<td>3.85</td>
</tr>
<tr>
<td>Overall SegSNR</td>
<td>10.07</td>
<td>12.26</td>
</tr>
</tbody>
</table>

Table 7.8: SNR performance of 7 band SBC using simple matching and cross correlated error minimization.

By replacing the simple error minimization with the cross correlation error minimization discussed above 7 and 14 band SBCs were simulated (parallel VQ). SNR performances of 7 and 14 band SBC using cross correlation error minimization and simple matching (using equations (7.18) and (7.15) respectively) are tabulated in Table 7.8 and Table 7.9 respectively.
<table>
<thead>
<tr>
<th>Band</th>
<th>Simple-Matching (dB)</th>
<th>Cross-Correlation (dB)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>5.98</td>
<td>9.13</td>
</tr>
<tr>
<td>2</td>
<td>7.84</td>
<td>15.96</td>
</tr>
<tr>
<td>3</td>
<td>7.96</td>
<td>18.32</td>
</tr>
<tr>
<td>4</td>
<td>7.09</td>
<td>9.58</td>
</tr>
<tr>
<td>5</td>
<td>5.11</td>
<td>5.89</td>
</tr>
<tr>
<td>6</td>
<td>5.08</td>
<td>7.23</td>
</tr>
<tr>
<td>7</td>
<td>5.24</td>
<td>7.27</td>
</tr>
<tr>
<td>8</td>
<td>2.83</td>
<td>3.15</td>
</tr>
<tr>
<td>9</td>
<td>1.61</td>
<td>1.26</td>
</tr>
<tr>
<td>10</td>
<td>2.60</td>
<td>3.11</td>
</tr>
<tr>
<td>11</td>
<td>1.26</td>
<td>0.58</td>
</tr>
<tr>
<td>12</td>
<td>-1.20</td>
<td>-2.25</td>
</tr>
<tr>
<td>13</td>
<td>-0.03</td>
<td>-1.29</td>
</tr>
<tr>
<td>14</td>
<td>-0.13</td>
<td>-0.13</td>
</tr>
<tr>
<td>Overall SegSNR</td>
<td>6.78</td>
<td>10.79</td>
</tr>
</tbody>
</table>

Table 7.9: SNR performance of 14 band SBC using simple matching and cross correlated error minimization, using a 10 bit code-book.

In the usual simple matching process, vector energies were quantized using 3 bits. However, in the cross correlation process, scale factors, which we denoted here as $\sigma_{opt}$, needs an extra bit to achieve the same accuracy. The reason for this is that as it can be seen from equation (7.15), $\sigma_{opt}$ can be negative as well as positive. Therefore an extra bit is necessary for the sign bit, which means for the 7 band and 14 band SBCS cross correlated error minimization results in 1000 bits/sec and 500 bits/sec more transmission.
capacity respectively.

Results presented in Table 7.8 and 7.9 show that cross correlated error minimization outperforms the simple matching technique. Good quality speech can thus be obtained at bit rates as low as 9.6 Kb/s. Results in Table 7.8 and 7.9 also show that low frequency high energy bands are extremely well quantized but that the high frequency low energy bands have larger quantization noise than the simple matching process. Although, the subjective quality of processed speech could be considered good, when tested using a sensitive ear-piece, high frequency distortions could be heard. In order to reduce these distortions in the high frequency bands noise shaping was used by modifying equation (7.18) (only the second term is considered).

\[
MIN \ E_r = MAX \ \frac{\left[ \sum_{i=1}^{b} \sigma^p(i) V(i) S(i) \sigma^{-p}(i) \right]^2}{\sum_{i=1}^{b} \left[ \sigma^p(i) V(i) \right]^2}
\]

In equation (7.19), \( p \) is the noise shaping factor and found to be best around 0.75. Although the overall SegSNR's of both 7 and 14 band coders were about 0.5 dB less than for the case with \( p = 1 \), (flat noise spectrum), using \( p = 0.75 \) and a 7 band coder high quality speech was produced at around 11 Kb/s. Increasing the number of bands from 7 to 14 also introduced quantization noise but high quality speech can be produced at bit rates around 9.6 Kb/s.

The delay of SBC with cross correlated error minimization is exactly the same as SBC with simple matching and serial vector quantization. Although equation (7.19) looks relatively complex for computation, it can be simplified to have the same complexity as the simple matching. If top and bottom of equation (7.19) is divided by \( \sigma^{2p} \) and \( V(i) \) is normalized to unit variance only \( b \) multiply-adds will be required for each level of the code-book. Also during the searching of the code-book, calculations can be first restricted to a smaller number of elements, e.g. 3, to minimize the candidate vectors for full search.

### 7.3 Transform Coder

In chapter 6 optimization methods for transform coders were discussed in order to produce high quality digital speech at around 16 Kb/s. In this section we explain how
these transform coders can be further refined to produce good quality speech at even lower bit rates. During optimization procedures the results obtained for SBC will be used where applicable.

All three types of transform coders (Zelinsky and Noll's, Vocoder driven and Pitch driven) which were optimized for 16 Kb/s were tested at 9.6 Kb/s by reducing the capacity allocated to code the residual signal. Results obtained were in the region of fair to poor. Band limitation, aliasing effects and DCT block edge effects were the three major reasons for quality degradation. It was seen that DCT block edge effects and aliasing occur when there is quantization noise, which in turn depends on the number of bits allocated to code the residual transform coefficients. Therefore, it is necessary to improve signal to quantization noise ratio, if good quality digital speech is to be produced at around 9.6 Kb/s. This can be done in two ways, as with SBC. First, more bits can be allocated to code the residual coefficients and hence improve SNR of residual coefficients and secondly, improve the quantization by replacing the scalar quantizers with VQ.

7.3.1 Zelinsky and Noll's approach

In order to be able to allocate more bits for coding the residual coefficients side information can be reduced by using VQ to code the regional average energy of transform coefficients. If the number of (equal) bands in an SBC is the same as the number of (equal) regions in the transform coder, and provided that their analysis block lengths are equal, then SBC band energy code-books can be used to vector quantize the regional average energy of the transform coefficients.

Using the 14 band SBC side information code-books Zelinsky and Noll's approach was simulated with the following parameters.

- Analysis window: 256 samples.
- DCT transform size: 128.
- Max bit per coefficient: 4.
- Side information: \[\frac{8000}{256} \times 10 + 14\] = 750 bits/sec.
- Residual information: 9000 bits/sec.
- Total information: 9750 bits/sec.

Each analysis block was split into two groups of 128 samples and frequency transformed using 128 point DCT. Each group of 128 transform coefficients were then split into 16 groups of 8 coefficients. Mean values of 16 groups of 8 coefficients in each
block were then calculated and finally the average values of corresponding 16 means in each block were computed. Using the two 7 bits energy code-books for the 14 band SBC, side information was vector quantized and used to adapt APCM step sizes and allocate bits. As in SBC, when decoding, zero bit allocated segments were filled with Gaussian noise. The amount of noise was adjusted by 0.6σ, where σ was the coefficients average in the zero bit allocated segment. The signal to noise ratios of 14 regions, and the overall segmental SNR are tabulated in Table 7.10.

Results in Table 7.5 and 7.10 show that SBC with VQ on the side information and with the same parameters has larger band SNRs as well as overall SNR than the transform coder. The obvious cause of reduction in SNR is the high pitch test data. The high pitch speech produces clear pitch harmonics even if the DCT size is 128. It was stated in chapter 6 that when the pitch harmonics are well defined, APCM quantizers in each band (region) of the spectrum produce large clipping errors.

Although, the overall performance of ATC with VQ on the side information was much better than ATC with no VQ, band limitation, aliasing and transform block edge effects could still be heard during high pitch talkers (female). Clipping errors can be reduced by using smaller size DCTs, but as the transform block edge effects are proportional with the decrease in the transform size, this will introduce more noise. Quality of the coded speech for low pitch talkers (male) was very similar to the 16 Kb/s quality. Delay in the ATC depends on the DCT size used. It can be 16 or 32 msec, because the most likely DCT sizes that one can use are 128 or 256 point. The complexity of the coder also depends on the size of DCT used. Complexity of ATC with VQ on the side information is very similar to an SBC with VQ on the side information. The only difference between them is the complexity of filter bank and DCT implementations.
<table>
<thead>
<tr>
<th>Frequency Region</th>
<th>SNR (dB)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>7.21</td>
</tr>
<tr>
<td>2</td>
<td>14.89</td>
</tr>
<tr>
<td>3</td>
<td>14.39</td>
</tr>
<tr>
<td>4</td>
<td>11.91</td>
</tr>
<tr>
<td>5</td>
<td>9.34</td>
</tr>
<tr>
<td>6</td>
<td>8.87</td>
</tr>
<tr>
<td>7</td>
<td>8.18</td>
</tr>
<tr>
<td>8</td>
<td>7.02</td>
</tr>
<tr>
<td>9</td>
<td>9.10</td>
</tr>
<tr>
<td>10</td>
<td>9.34</td>
</tr>
<tr>
<td>11</td>
<td>8.51</td>
</tr>
<tr>
<td>12</td>
<td>6.92</td>
</tr>
<tr>
<td>13</td>
<td>8.14</td>
</tr>
<tr>
<td>14</td>
<td>9.15</td>
</tr>
<tr>
<td>Overall SegSNR</td>
<td>13.2</td>
</tr>
</tbody>
</table>

Table 7.10: SNR performance of 9.7 Kb/s Zelinsky and Noll's approach with vector quantized side information.

7.3.1.1 Fully Vector Quantized ATC

We have shown above that ATC with VQ on the side information cannot maintain good quality across a range of talkers at around 9.6 Kb/s. Although, reducing the side information and hence increasing the residual coefficients coding capacity improves the quality of ATC with no VQ, more improvement in quality is required. Following similar procedures as for SBC, residual coefficients were vector quantized. Unlike SBC, in ATC,
coefficients can be split in any order with equal ease. However, for the sake of simplicity all splitting was performed equally. In order to eliminate the need for bit allocation, parallel VQ with cross correlated error minimization as in SBC was used. However, rearrangement of the coefficients was necessary before applying VQ. This rearrangement is shown in Figure 7.7. Each vector was constructed using the expression.

\[ X_k(i) = C((i \cdot b) + k) \quad i = 0, 1, \ldots, b - 1, \quad k = 1, 2, \ldots, \frac{N}{b} \]  \hspace{1cm} (7.20)

Where \( C(i) \) are the transform coefficients, \( b \) is the number of split regions and \( k \) is the number of coefficients in each region and \( N \) is the DCT transform size.

In order to keep the side information to a minimum, band (region) energies were updated every 256 samples. Therefore, each analysis block was split into two groups of 128 and transformed using 128 point DCT. Only the first 108 coefficients of each of the two groups of 128 coefficients were taken and split into 12 bands where each band contained 18 coefficients. Average values of these coefficients were calculated and split into two groups of 6. Each group average was then calculated and coded with 5 bits and then used to normalize the group elements for vector quantization. Using 7 bit code-books, two 6 element energy vectors were vector quantized. Equations (7.19) and (7.15) were then used to vector quantize 12 element residual coefficient vectors using a 10 bit code-book. After rearranging the coded coefficients and IDCT transforming them coded speech was recovered and compared with ATC using VQ on the side information. The overall bit rate of the coder was 8437.5 bits/sec for residual coefficients and 750 bits/sec for side information adding up to 9.186 Kb/s. The segmental signal to noise ratio of the coder was 13.5 dB, which was a little higher than the one with VQ on the side information only. However, subjective quality improvement was much higher. The band limitation effect was completely eliminated and block edge effects were reduced. However, slight aliasing effects of high pitch talkers could still be heard when tested with a highly sensitive ear-phone. Extra delay and complexity of fully vector quantized ATC is exactly the same as that of fully vector quantized SBC.

7.3.2 Vocoder Driven ATC

As was expected, the vocoder driven coder with the following parameters produced good results at around 9.6 Kb/s.
Figure 7.7: Rearrangement of frequency coefficients for parallel vector quantization.
Analysis block: 256 samples (32 msec).

DCT size: 256-point.

LPC order: 10.

Single tap pitch.

Side information: 10 bits for pitch and pitch gain, 44 bits (5,5,5,4,4,4,4,4,4,4) for 10 LPC coefficients, 6 bits for gain, for every 256 samples, thus, 1875 bits/sec.

Residual bits/block: 247. Total bit rate: 7718.75 + 1875 = 9594 bits/sec.

The quality of the coded speech was more or less steady across various high and low pitch talkers. Also there was no band limitation effect. This was due to better modelling of the discrete spectra via the use of the pitch model. However, coded speech showed quantization errors and occasional clicks. Quantization noise was obvious, because of the shortage of bits to code the spectra efficiently. Occasional clicks were due to mismatches between the original and estimated spectrums.

The delay of the coder was 32 msec when sampled at 8 KHz, but the complexity of the coder was very high. For every 256 samples, at the encoder 256 point DCT, 256 point DFT and at the decoder 256 point IDCT and 256 point DFT is required. Unless some simplifications are made this coder is not practical. The increased complexity compared with ATC (Zelinsky and Noll's) is not worth the quality improvement.

7.3.2.1 Fully Vector Quantized Vocoder Driven ATC

In order to reduce side information and hence allocate more bits to code the spectral coefficients and also to replace the scalar quantizers with the vector quantizers, vocoder driven ATC was fully vector quantized. Using the arrangement shown in Figure 7.7 with parallel vector quantization, the long and complex bit allocation process was eliminated.

7.3.2.1.1 Vector Quantization Of LPC Parameters

Similar procedures as for SBC band energies can be followed to design code-books for LPC parameters. It was mentioned in chapter 5 that it is not a good idea to quantize the LPC parameters directly because of their sensitivity to quantization errors. Therefore, they are usually transformed into another domain before quantization. One of the most common transformations is called log area ratios (LAR) and given by equation (5.10) in chapter 5 [4][5].
It is also possible to quantize the autocorrelation coefficients before calculating the LPC parameters. In our simulation we have used the gain normalized version of the Itakura-Saito distortion measure \([5][9]\), which is given by the expression,

\[
d[x, y] = \frac{R_x(0)R_a(0)}{E_{\text{min}}} + 2 \sum_{i=1}^{M} \frac{R_x(i)R_a(i)}{E_{\text{min}}} - 1
\]

(7.21)

where \(R_x(i)\) are the autocorrelation values of the current block of speech from which LPC parameters are calculated, \(E_{\text{min}}\) is the energy in the residual after inverse filtering with the original LPC parameters and \(R_a(i)\) are the autocorrelation sequences of LPC parameters stored in the code-book. \(x\) and \(y\) represent the optimal LPC parameters and code-book entries respectively.

As in SBC band energies, for the full search code-book, process is started with an initial randomly chosen code-book. The size of the initial code-book chosen was 1540 vectors. The size of the code-book was gradually reduced by ignoring the least used vectors in steps of 50's and 10's. At each reduction step the remaining code-book entries were optimized twice by calculating the centroids of the autocorrelation vectors of the clusters, and then computing the LPC parameters for each cluster using the centroid autocorrelation values. When the size of the code-book was reduced to 1024 vectors, at each further reduction step, the code-book was optimized 4 times. The performance of the various length code-books were then checked by measuring the block segmental energy in the residual signal. Results obtained are tabulated in Table 7.11 (for 10 LPC parameters in each vector). Scalar quantization was performed by using LAR's transformation.

Also during tests of the use of vector quantized LPC parameters in coders, similar results to those given in Table 7.11 were obtained by subjective listening tests. It was concluded that a 10 bit full search code-book was the best compromise in terms of performance and complexity. Therefore, using a 10 bit code-book to vector quantize the 10 LPC parameters of the vocoder driven transform coder the side information was reduced by \((44-10) \times 8000/256 = 1062.5\) bits/sec. This allowed 34 more bits per 256 samples to be used to code the spectral coefficients, which improved the quality of the coded speech and the SNR was increased by about 1.2 dB.
Table 7.11: Comparison of scalar and vector quantization of LPC parameters in terms of their equivalent number of bits for coding.

<table>
<thead>
<tr>
<th>VQ Bits</th>
<th>Scalar Bits</th>
</tr>
</thead>
<tbody>
<tr>
<td>11</td>
<td>30</td>
</tr>
<tr>
<td>10</td>
<td>26</td>
</tr>
<tr>
<td>9</td>
<td>23</td>
</tr>
<tr>
<td>8</td>
<td>16</td>
</tr>
</tbody>
</table>

7.3.2.1.2 Vector Quantization of Spectral Coefficients

Vector quantization of the spectral coefficients was performed in the same way as in the ATC of Zelinsky and Noll's. The estimated spectrum was normalized and then rearranged together with the original spectrum using equation (7.20), for parallel vector quantization. During rearrangement the first 216 coefficients were considered and were split into 18 groups of 12. Each group of 12 was then quantized with a 10 bit Gaussian code-book. Equation (7.18) was used to search the code-book and 7.15 was used to calculate each vector scale value. In equations (7.15) and (7.18) energy vector (σ(i)) was replaced by the rearranged estimated spectral coefficients vector. In order to reduce the code-book searching complexity, in the first stage of search only a few elements of each vector which coincide with the maximum elements of the estimated vectors were considered. Therefore, the procedure and the complexity of vector quantization of spectral coefficients of vocoder driven ATC was exactly the same as that of the ATC of Zelinsky and Noll's. In order to reduce the dynamic range of 18 scale factors they were normalized by their average energy which was coded with 6 bits. Each scale factor was then coded with 5 bits, one for the sign and 4 bits for the magnitude. Overall parameters of vector quantized vocoder driven ATC was as follows.

Analysis length: 256 samples.
Order of LPC: 10.
Side information: 7 bits for pitch and 4 bits for pitch gain, 10 bits for LPC parameters, 6 bits for average energy, and 5 bits for each vector scale value. (11+10+6+(18x5)) 8000/256 = 3657 bits/sec.
Residual information: \(18 \times 10 \times 8000/256 = 5625\) bits/sec.

Overall bit rate: \(3657 + 5625 = 9282\) bits/sec.

Vector quantizer parameters can be varied to operate at various rates from 8 Kb/s to 12 Kb/s. Speech produced at around 9.6 Kb/s was considered to be very good quality. No aliasing or band limitation could be heard even with highly sensitive ear-phones. Provided that the first round search of the code-book did not reduce the size of the code-book below 128 vectors, no quantization noise could be heard. However, occasional block edge effects of the DCT could be heard.

Delay of the coder was 32 msec when sampled at 8 KHz, assuming no computation time. However this delay will increase because of extra delay for computation. Fully vector quantized vocoder driven ATC is more complex than the already very complex vocoder driven ATC with no VQ. Unless some form of simplifications are made, real time implementation will be very costly. Some possible simplifications may include tree search code-books for coding the LPC parameters and possible elimination of DFT's. We have shown in chapter 6 how vocoder driven ATC can be simplified and yet still maintain high quality. This we called pitch driven transform coder. In the following section we will show how the pitch driven transform coder can further be improved to produce high quality speech at around 9.6 Kb/s as well as keeping the complexity relatively low.

### 7.3.3 Hybrid Transform Coder

In recent years frequency domain speech coding has been shown to be efficient at bit rates of as low as 16 Kb/s. Vocoder driven ATC as seen earlier has been proposed for lower bit rates than 16 Kb/s. Although, vocoder driven ATC can produce high quality speech at around 9.6 Kb/s, its very high complexity is a big disadvantage. The two main types of distortion seen in low bit rate transform coders are band limitation and occasional clicks with some aliasing. These are simply the result of poor representation of the residual signal. These distortions are eliminated in the Hybrid Transform Coder (HTC) [6] using three improvements, e.g. reduced side information, improved quantization and use of high frequency regeneration (HFR).

A block diagram of the HTC is shown in Figure 7.8. First, speech is analysed to measure the pitch period and to calculate 10 linear prediction coefficients. The LPC parameters are vector quantized using the same 10 bit full search code-book as was used in the VDATC. Speech is then filtered through the LPC inverse filter. The remaining
Figure 7.8: A block diagram of HTC (a) encoder, (b) decoder.
residual signal is then frequency transformed by a 256 point DCT. In order to remove
the pitch harmonics of the residual spectrum, the original coefficients are divided by their
estimated values. Coefficients in the lower frequency part of the spectrum (0-1.5 KHz)
are then quantized and transmitted. Quantized coefficients are also cross correlated with
the high frequency coefficients to find the best matching points for high frequency regen-
eration. With this information at the decoder a full band residual spectrum is obtained
by shifting the received coefficients to the calculated frequency points. Full band
coefficients are then inverse discrete cosine transformed and filtered through the LPC
synthesis filter to recover the output speech.

In the proposed coder most of the side information is due to the transmission of the LPC
parameters. To reduce this by around a factor of 4, a 10 bit full search code-book is used
to vector quantize the LPC parameters. The distortion measure and also the procedure for
designing the code-book is exactly the same as that given in VDATC.

7.3.3.1 Quantization Of Residual Coefficients

In order to eliminate band limitation effects due to the shortage of bits to code the
complete 4 KHz band, it was decided to transmit a portion of the spectrum and to use
high frequency regeneration to obtain the remainder. Although, this does eliminate the
band limitation effects, it introduces high frequency distortion which increases with
decrease in the transmitted bandwidth. Therefore, it is necessary to transmit the largest
possible bandwidth in order to keep high frequency distortion low. In order to transmit
the largest possible bandwidth without noticeable clicks or quantization noise, two
quantization algorithms were investigated. These were scalar block quantization with
dynamic bit allocation and vector quantization.

Scalar Block Quantization

Prior to quantization an estimate of the spectrum is used to both remove the pitch
harmonics and to allocate bits. To create a frequency domain estimate (pitch pattern) of
the residual coefficients, the model given below was used:

\[ C_p(\omega) = \left| \frac{1}{1 - Ge^{-j\omega P}} \right| \]  \hspace{1cm} (7.22)
where $C_p(\omega)$ represents the magnitude of the estimated frequency coefficients, $p$ is the pitch period and $G$ is the pitch gain. During simulations it was found that computing $G$ for every block was not necessary. In fact it caused problems for extreme values of $G$. This was due to the estimated levels between the pitch harmonics being very small or very large, thus, causing the quantized signal to have large clipping errors. After several experiments it was found that a constant pitch gain of 0.65 produced the best results, see Figure 7.9 for typical waveforms of original and estimated residual coefficients. In order to maximize the signal to quantization noise ratio and hence, transmit the largest possible bandwidth for a given bit rate, bits were allocated according to the variations in the coefficient amplitudes. Using the estimated coefficients, bits were allocated in two steps.

(i) Find the largest amplitude, allocate one bit to it and divide by 2.

(ii) Check if all bits are allocated, stop else go back to (i).

In order to avoid cases in which all bits are allocated to the pitch harmonics and no bits to the coefficients in between the harmonics, it was found necessary to limit the maximum and minimum bits per coefficient to 4 and 2, respectively. This produced 13.87 dB signal to quantization noise ratio as against 12.30 dB for the fixed bit allocated case (measured using voiced and unvoiced data), thus enabling transmission of the 1.56 KHz base-band. Here, 281 bits were used to code the first 100 of 256 coefficients (1.56 KHz). Although, the SNR difference between the fixed and variable bit allocated cases was only 1.57 dB, the subjective difference was much more pronounced.

Vector Quantization

In addition to vector quantizing the LPC parameters, vector quantization was also used to quantize the residual coefficients. It was assumed that the residual coefficients after removal of the pitch harmonics, had Gaussian distribution. Therefore, random Gaussian code-books were used for vector quantization. During the quantization process 7 and 8 bit code-books, with vector dimensions of 4 and 5 were tried. Using the mean squared error minimization two types of search procedures, and optimum scale calculations were tested.

(i) Direct Error Minimization (DEM)

The block diagram of the DEM is shown in Figure 7.10. The first 100 coefficients of the residual were normalized and divided by the created pitch pattern. They were then
Figure 7.9: Spectral comparison of LPC residual coefficients (a) original, (b) estimated.
grouped into 4's and 5's and again normalized. Using random Gaussian code-books mean squared error between the input and the code-book vectors were calculated as in equation (7.5). In the decoder vectors which minimized the error were then scaled up by the energy in each corresponding input vector to recover the quantized residual coefficients, which were then multiplied by the created pitch pattern and finally scaled up by the block energy.

(ii) Synthesized Error Minimization (SEM)

Block diagrams of synthesized error minimization and code-book search are shown in Figure 7.11a and 7.11b. In SEM, vectors in a unit variance Gaussian code-book were multiplied by the created pitch pattern and then scaled up by a scale factor. Original input vectors including their pitch harmonics were then compared with the synthesized vectors. In the decoder, code-book vectors which minimized the error were multiplied by the pitch patterns and then scaled up by the optimum scale factor to recover the quantized residual coefficients. Here the error was minimized using equation (7.18) and the optimum scale factors were calculated using equation (7.15) where $\sigma(i)$'s were replaced by the created pitch patterns.

<table>
<thead>
<tr>
<th>Bits</th>
<th>Vector dimension</th>
<th>DEM (dB)</th>
<th>SEM (dB)</th>
</tr>
</thead>
<tbody>
<tr>
<td>8</td>
<td>5</td>
<td>9.09</td>
<td>12.23</td>
</tr>
<tr>
<td>8</td>
<td>4</td>
<td>11.99</td>
<td>15.54</td>
</tr>
<tr>
<td>7</td>
<td>4</td>
<td>10.23</td>
<td>13.73</td>
</tr>
</tbody>
</table>

Table 7.12: VQ performance of residual quantization in HTC.

The performance of the two types of error minimization with respect to the size of the code-books and vector dimensions are given in Table 7.12.

Results tabulated in Table 7.11 show that SEM which uses cross correlation error minimization has about 3.5 dB better performance than DEM, which was confirmed by subjective quality comparison. This was also shown to be the case for the sub-band coder in section 7.2.2. There are three major reasons for this.
Figure 7.10: A block diagram of direct error minimization.

Figure 7.11: Block diagrams of synthesized (a) error minimization (b) code-book search.
The first reason concerns the difference between the optimum scale (gain) calculation for each vector. For DEM, decoded vectors are scaled by the energy in the original vector, which means even if the noise power is larger than the signal power, transmission still takes place. However, in the SEM case, scale calculation takes into account the expected noise power.

The second reason is that in DEM the error is amplified by the pitch filter (in SBC, ATC and VDATC errors are amplified by band energies, coefficient support values and the estimated spectral coefficients respectively). However, for SEM this is not the case since the error is minimized after the pitch filter response.

Finally, in SEM the scale factor can take either negative or positive sign, which means that the size of the code-book is virtually doubled with no extra storage and computation for the search. In DEM the scale is always positive.

7.3.3.2 High Frequency Regeneration

The principles of the HFR schemes used are explained in section 4.3 [7]. At the encoder, quantized coefficients were cross correlated with the unquantized coefficients in the 1.5 to 3.0 KHz region. In the cross correlation process three shifts on either side of the 1.5 KHz point were calculated. The location of the maximum in magnitude and its sign bit were transmitted. At the decoder this information was used to shift the received coefficients to optimum locations. The region from 3.0 to 4.0 KHz was simply filled by translating the coefficients from 0.5 to 1.5 KHz.

7.3.3.3 Discussion

The goal of producing a good quality transform coder at around 9.6 Kb/s has been achieved by reducing side information, improving quantization and using high frequency regeneration. Side information was reduced by vector quantizing the LPC parameters and also by eliminating the need for variable pitch gain. However, extra side information was needed for high frequency regeneration. Therefore, the total side information is as follows,

- Analysis block: 256 samples.
- 10 LPC parameters: 10 bits.
- Pitch: 7 bits.
HFR shift and sign: 4 bits.

Block energy: 5 bits (5 bits were used to code the average of the vector scale factors).

Total side information: \[(10+7+4+5) \times 8000/256 = 812.5 \text{ bits/sec}\]

In order to improve quantization of the base-band coefficients, scalar block quantization with dynamic bit allocation and vector quantization were used. For scalar block quantization, 100 coefficients were normalized by their block energy which was coded with 5 bits and then quantized with 281 bits producing 8782 bits/sec and thus an overall bit rate of \(8782 + 812.5 = 9594 \text{ bits/sec}\). For the vector quantization each block of 100 coefficients was split into 20 groups of 5. Then each vector was coded with 8 bits and its scale factor with 4 bits. Before coding the scale factors, they were normalized by their magnitude average which was coded with 5 bits. This produced a base-band residual of 7500 bits/sec and an overall bit rate of \(7500 + 812.5 = 8313 \text{ bits/sec}\). Of course other vector quantization combinations with different sizes of code-books and different vector dimensions are available for various bit rates. The efficient quantization was also due to transmission of the base-band only.

The use of high frequency regeneration usually introduces tonal distortions in base-band type coders. This is because of broken pitch harmonics in folded sections of the spectrum. However, in the proposed HTC, high frequency regeneration is performed in the frequency domain which makes it possible to locate the pitch harmonics at the correct frequency points. This eliminates the tonal distortion.

Using either scalar block quantization with dynamic bit allocation or vector quantization, good quality speech can be produced at around 9.6 Kb/s, with relatively low complexity. The major complexity of the HTC lies in the DCT and inverse DCT. The LPC code-book search is also complex but this can be simplified. When using scalar block quantization with dynamic bit allocation, pitch adaptive quantizers may be used to replace the bit allocation process. For vector quantization, the search may be simplified in two stages as explained in SBC and ATC. For high frequency regeneration, points of shift may be calculated directly from the measured pitch period. This will eliminate the need for cross correlation calculation.
7.4 Linear Predictive Coding Of Speech With VQ and Frequency Domain Noise Shaping

In chapter 6 section 6.3.1.3 we have shown how a VDATC can be simplified and still maintain high quality speech, which we called the pitch driven transform coder (PDTC). In PDTC LPC inverse and synthesis filters were implemented in the time domain but residual quantization was performed in the frequency domain. Because there were sufficient bits to code the residual signal, there was no need to consider any noise shaping. In order for the PDTC to operate at around 9.6 Kb/s (1 bit per sample coding approximately), it is necessary to vector quantize the residual coefficients. Let us assume that a 10 bit code-book is employed and also 4 bits are used to code the gain of each vector, then the minimum dimension of each vector should be at least 14, which is quite high.

Using the following parameters PDTC was vector quantized; 256 residual coefficients were split into 16 vectors with 16 consecutive coefficients in each. Created pitch pattern was also split in the same way. Using equations (7.18) and (7.15) residual coefficient vectors were vector quantized with a 10 bit random Gaussian code-book. The results were not very good. The recovered speech had low frequency roughness and clearly audible quantization noise. Annoying high frequency distortion was also heard. The results were even worse when the simpler search using equation (7.5) was used.

Consider the model for vector quantization of the LPC residual shown in Figure 7.12. In Figure 7.12 it is assumed that $V(i)$ is optimized to take care of the pitch pulses in $S(i)$ and that vectors $S(i)$ and $V(i)$ are unit variance. The difference between $S(i)$ and $V(i)$ is fed through the synthesis filter. If $V(i)=S(i)$, then $d(i)=0$, which means there is no quantization error. However, when $d(i)\neq 0$ there is always quantization error present.

In PDTC, mean squared error was minimized in such a way that $d(i)$ was minimized. Since $S(i)$ and $V(i)$ have flat spectra, $d(i)$ also has a flat spectrum. $\hat{d}(i)$ on the other hand has a spectrum shaped by the LPC synthesis filter. In practice we are more interested in $\hat{d}(i)$ than $d(i)$. However, when $d(i)$ is very small it may be assumed to be equivalent to $\hat{d}(i)$. This is one of the most important reasons behind PDTC performing well at 16 Kb/s and not so well at 9.6 Kb/s. At 16 Kb/s $d(i)$ is approximately equal to $\hat{d}(i)$, which means

$$\sum_{i=1}^{N} [(S(i) - V(i))^2] = \sum_{i=1}^{N} [S(i) - V(i)]^2$$

(7.23)
Figure 7.12: A block diagram of vector quantization of LPC residual.
where \( h_1(i) \) is the impulse response of the LPC synthesis filter. However, at 9.6 Kb/s because of large quantization noise \( d(i) \neq \hat{d}(i) \), which also makes equation (7.23) false. Therefore, the speech quality of PDTC at 9.6 Kb/s may be improved if an appropriate noise shaping is introduced during the searching of the code-book.

In PDTC residual coefficients were quantized in the frequency domain, one reason being to allocate most of the bits to the high energy pitch harmonics. Pitch harmonics were also removed in the frequency domain before quantization. However in the case of vector quantization of the residual coefficients each vector is quantized using the same code-book with the same number of levels, which makes the frequency domain bit allocation redundant. Hence the pitch harmonics can be removed in the time domain. Therefore, the resulting coder was named linear predictive coding with VQ and frequency domain noise shaping.

7.4.1 Coder Description

The block diagram of Linear predictive coding with VQ and frequency domain noise shaping (LPC-VQ-FNS) is shown in Figure 7.13. A block of speech is inverse filtered using LPC and pitch filters and then frequency transformed using DCT. In parallel with the inverse filtering an estimate of the speech envelope was obtained either by filter bank or FFT computations. Both the residual coefficients and the estimated envelope coefficients were then rearranged as discussed in section 7.3.1.1 and shown in Figure 7.7. Whilst searching for the optimum vector, error vectors produced by each sequence in the code-book were weighted by their corresponding estimated envelope vectors and then the minimum weighted error sequence was selected as the optimum. The resulting decoder is fairly simple. Received vectors were scaled up by their scale factors, inverse DCT transformed and then fed through the pitch and LPC synthesis filters to recover the output speech.

7.4.1.1 Noise Shaping

In order to shape the quantization noise after the LPC synthesis filter (which introduces the envelope to the residual signal), information concerning the spectral shape of the original speech should be known. Noise can be shaped in time domain [8][3] using the LPC coefficients. However, this will involve excess amount of computations because of the convolution processes during the code-book search [9]. In order to eliminate large computations it is sensible to apply noise shaping in the frequency domain where there is
Figure 7.13: A block diagram of LPC-VQ-FNS (a) encoder, (b) decoder.
no convolution. There are two ways of estimating the spectral envelope of the speech in the analysis block. One is the sub-band coder approach, where a filter bank is used to split the speech signal into a number of bands. Energy in each band yields the average point on the spectrum for the frequency range equal to the width of each band. The accuracy of this method depends on the number of bands used in the estimation process.

The other method of estimating spectral envelope is the transform coder approach, where the impulse response of the LPC inverse filter is Fourier transformed to obtain a very close estimate. Here the accuracy depends on the size of the FFT, but sizes as low as 128 point are adequate for the noise shaping purposes.

The accuracy of the envelope estimate required also depends on the dimension of the residual vectors. Large residual vector dimensions means that each element of the vector represents a finer frequency range requiring a finer spectral estimate for efficient noise shaping. For vector dimensions 16 and 32, 8 and 16 band SBC respectively were adequate.

7.4.1.2 Vector Quantization Of Residual Signal

Two of the most important aspects of a vector quantizer are,

(i) the distortion measure to be used and,

(ii) the finding of the optimum code-book to match the data using the distortion measure.

The distortion measure used herein is the weighted mean squared error, which is simple and straightforward. Equations (7.18) and (7.15) are directly applicable. The code-book used is however not as simple as the distortion measure. In HTC we have used Gaussian random code-books because the data contained in each vector was assumed to be Gaussian. The DCT transformed LPC residual could not be coded using Gaussian code-books if the pitch harmonics are not removed, and the vector dimensions are not very small. Larger dimension vectors would have pitch patterns formed within them. This of course would need specifically trained code-books to account for the patterns. The same will apply to the case in question. Therefore, it is important to make the residual as close as possible to a Gaussian random process before quantization, if Gaussian code-books are to be used. The only way one can randomize the residual signal in the time domain is to use the LPC and pitch inverse filters in sequence. Correlation in speech cannot be absolutely removed, otherwise we would not need to transmit any residual signal. However, if the
parameters of LPC and pitch filters are updated often enough the remaining residual tends to be very close to a Gaussian random process [2]. The transmission capacity required to transmit the LPC and pitch filter parameters sets a limit to the update rate of these parameters.

### 7.4.2 Simulations

In the first part of the simulations various LPC and pitch update rates were tested to compare the performance of random Gaussian (RG), trained Gaussian (TG) and trained (T) full search code-books. The random Gaussian code-book was constructed from random Gaussian numbers regardless of the numbers. The trained Gaussian code-book was also constructed from random Gaussian numbers but it was trained to make sure that all chosen vectors were useful. Trained code-book was constructed and trained using the residual signal itself. Three update rates of the LPC and pitch parameters were used, (10 LPC and a single tap pitch), every 256, 128 and 64 samples. The results are shown in Table 7.13.

<table>
<thead>
<tr>
<th>Parameters update (samples)</th>
<th>256 (dB)</th>
<th>128 (dB)</th>
<th>64 (dB)</th>
</tr>
</thead>
<tbody>
<tr>
<td>RG</td>
<td>8.0</td>
<td>8.6</td>
<td>9.0</td>
</tr>
<tr>
<td>TG</td>
<td>8.3</td>
<td>8.7</td>
<td>9.1</td>
</tr>
<tr>
<td>T</td>
<td>9.0</td>
<td>9.1</td>
<td>9.2</td>
</tr>
</tbody>
</table>

Table 7.13: Performance of various code-books using weighted mean squared error measure.

The results in Table 7.13 were obtained using 10 bit code-books, 256 point DCT and vector dimensions of 32. Similar relative performances were obtained for 9, 8, and 7 bit code-books. The results in Table 7.13 confirm that the argument made in the previous section 7.3.4.1.2 that when the speech residual contains redundancies it is better to use trained code-books. However, the performance of the random Gaussian code-book is as good as any other code-books if redundancies in speech as far as possible removed. Redundancies can be removed simply by increasing the update rate of model parameters.
This of course requires increasing side information. In order to find the best update rate for LPC and pitch parameters, in terms of code-book performances, tests were carried out. Firstly, pitch parameters update rate was kept constant at every 256 samples and 10 LPC parameters were updated at various rates. The results are shown in Table 7.14.

<table>
<thead>
<tr>
<th>LPC update (samples)</th>
<th>256 (dB)</th>
<th>128 (dB)</th>
<th>64 (dB)</th>
</tr>
</thead>
<tbody>
<tr>
<td>RG</td>
<td>8.0</td>
<td>8.2</td>
<td>8.2</td>
</tr>
<tr>
<td>TG</td>
<td>8.3</td>
<td>8.3</td>
<td>8.3</td>
</tr>
<tr>
<td>T</td>
<td>9.0</td>
<td>9.1</td>
<td>9.1</td>
</tr>
</tbody>
</table>

Table 7.14: Code-book performances with respect to LPC parameters update rate (pitch update every 256 samples).

Secondly, the pitch update rate was kept constant at every 64 samples and the results are tabulated in Table 7.15.

<table>
<thead>
<tr>
<th>LPC update (samples)</th>
<th>256 (dB)</th>
<th>128 (dB)</th>
<th>64 (dB)</th>
</tr>
</thead>
<tbody>
<tr>
<td>RG</td>
<td>8.8</td>
<td>8.9</td>
<td>9.0</td>
</tr>
<tr>
<td>TG</td>
<td>8.9</td>
<td>9.1</td>
<td>9.1</td>
</tr>
<tr>
<td>T</td>
<td>9.0</td>
<td>9.2</td>
<td>9.2</td>
</tr>
</tbody>
</table>

Table 7.15: Code-book performances with respect to LPC parameters update rate (pitch update every 64 samples).

Finally, to confirm the combined results of Table 7.14 and 7.15, the LPC parameters were kept updated at every 128 samples and the pitch parameters update rate was varied. Results are shown in Table 7.16.
Table 7.16 shows that pitch parameters update rate is more important than the LPC parameters update rate. There is not much to gain by transmitting the LPC parameters more often than about every 128 to 200 samples. However, the performance of the code-books (Gaussian) increases with the increase in the pitch parameters update rate. In fact if the LPC parameters are calculated every 128 to 200 samples and the pitch parameters are updated every 64 samples, the performance of the random Gaussian code-book becomes equivalent to that of the other code-books.

A 16 band SBC filter bank was used to obtain the weighting patterns $W(i)$. During the search, equation (7.13) was used to calculate the optimum scale and equation (7.16) was modified and used to calculate the weighted mean squared error. In equation (7.13) because the code-book sequences used were unit variance, only the top part (cross correlation) was calculated as,

$$\sigma_{opt} = \sum_{i=1}^{N} X(i) V(i)$$

(7.24)

where $X(i)$ are the DCT transformed LPC and pitch residual and $V(i)$ are the code-book sequences. Equation (7.16) was modified to,

$$E_r = \sum_{i=1}^{N} [(X(i) - \sigma_{opt} V(i))]W(i)]^2$$

(7.25)
to include the noise shaping vector \( W(i) \). Each element of 16 element vectors was weighted by the corresponding band energy.

Typical fixed parameters of the complete coder that was tested were as follows:

Analysis block length: 256 or 128 samples.

10 LPC parameters: 17 + 3 bits

Pitch parameters: 7 + 4 bits

Pitch update: every 64 samples

Power (gain): 6 bits

10 LPC parameters were quantized using 2 cascaded 9 and 8 bit code-books. The first code-book employed the mean squared error distortion measure to code the LAR parameters. The difference, error vectors were then normalized and coded again using an 8 bit code-book by employing mean squared error distortion. Here, stability checks were made to ensure the resulting filter was always positive by making sure that its \( K \) parameters were always in the unit circle. Pitch period was coded with 7 bits covering the range from 20 to 147 samples. Pitch gain was coded with 4 bits uniform quantization ranging from 0.03125 to 0.96875 in steps of 0.0625. The top limit of the pitch gain was kept to a value never more than 1.0 to ensure stable pitch synthesis filter. Pitch gains up to 1.8, even 2.0, were observed during simulations which were achieving better pitch prediction. However, due to the quantization errors the pitch synthesis filter was occasionally becoming unstable.

Residual coefficients were rearranged in 16 groups of 16 coefficients each, and coded with a 9 bit random Gaussian code-book. Scale factors were coded with 4 bits, one for the sign bit and 3 for the magnitude. Before coding each scale factor, the average value of all 16 was found and coded with 6 bits, and then used as a normalizing factor. The overall bit rate of the coder can be varied by changing the vector dimensions or model parameters update rate. For bit rates around 9.6 Kbps, the LPC update rate was every 128 samples, power was calculated every 256 samples and pitch parameters transmitted every 64 samples, giving a total side information of,

\[
(6 + 40 + 44) \times \frac{8000}{256} = 2812.5 \text{ bits/sec.}
\]

Using 9 bits for the vectors and 4 bits for their scale factors, for each of the residual vectors, the total residual signal was,
(9 + 4) 8000/16 = 6500 bits/sec.
making a total bit rate of 6500 + 2812.5 = 9312.5 bits/sec.

Subjective performance of the processed speech was very good. There was no low
frequency roughness and the high frequency distortion heard in PDTC was very much
reduced. When a low pass filter with a 3.3 KHz cut off was used at the output of the
coder high frequency distortion was reduced even further. Objective performance of the
coder in segmental SNR was found to be 14.3 dB.

7.4.3 Discussion

Here we have shown how a linear predictive coder can be vector quantized using
frequency domain noise shaping to produce good quality speech at around 9.6 Kb/s. Good
quality has been achieved by optimizing the pitch filter in the time domain more fre-
quently, (every 64 samples) to improve the gain of inverse filtering and by using adap-
tive noise shaping. These are the only two differences between the LPC-VQ-FNS and
PDTC. Here, the bit allocation process was eliminated which made the system more
robust to channel errors.

Complexity of LPC-VQ-FNS is approximately the same as that of the PDTC. Although, bit allocation was eliminated and the code-book search was simplified with
equation (7.24), extra computations are needed to calculate the noise shaping vectors
(spectral envelope). In PDTC, coding of the residual vector was performed by 10 bits,
and in LPC-VQ-FNS the complexity was halved by using 9 bits.

The delay of LPC-VQ-FNS can be reduced to 16 msec by reducing the size of the
analysis frame. This will increase the transmission capacity by 6 bits per 256 samples,
which can easily be accommodated in the 9.6 Kb/s overall transmission capacity. This
means that the size of DCT used should be 128 points or less, halving the DCT complex-
ity as well. In PDTC this cannot be accomplished because well defined pitch harmonics
are needed for pitch inverse filtering and bit allocation in the frequency domain.

Three more improvements can be added to LPC-VQ-FNS to improve the quality or
to reduce the total bit rate below 9.6 Kb/s. These are more efficient implementation of
the pitch filter, consideration of the gain of the LPC synthesis filter and the memory of
the pitch and LPC synthesis filters during the code-book search. These three considera-
tions will be explained in more details in chapter 8, and here we will only explain them
briefly. In Figure 7.13 during inverse filtering, the LPC filter is followed by the pitch
filter and vice versa when reconstructing. This system set up does not take into account the effect of the quantization noise. The predicted signal at the encoder is not the same as the predicted signal at the decoder unless there is no quantization error. Therefore, the gain achieved by inverse filtering does not necessarily correspond to the gain of the synthesizer, because prediction in synthesis is disturbed by the quantization noise. The second consideration is that when coding the residual vectors, only the shape of the LPC filter response is considered in selecting the best sequence. However there are two more important factors which are the memory of the synthesis filters and the gain of the LPC synthesis filter. If the above considerations are implemented then it will be possible to minimize the error between the original input speech and the recovered output speech.

7.5 Linear Predictive Base-Band Coding And High Frequency Regeneration Of Speech

Base-band coding and high frequency regeneration of speech is known as voice excited linear prediction (VELP) or residual excited linear prediction (RELP). Base-band coding is used where there are not enough bits to code the full band residual adequately. In section 7.3.3 we have shown how the idea of base-band coding and high frequency regeneration can be used to improve the quality of a transform coder at around 9.6 Kb/s. RELP is the time domain implementation of HTC with few differences. Therefore a RELP coder was simulated for comparison purposes with the other 9.6 Kb/s algorithms. In simulations the RELP coder given in reference [10] was used.

7.5.1 Coder Description

A block diagram of the coder under investigation is shown in Figure 7.14. Speech is stored in blocks and analysed to obtain the LPC parameters and inverse filtered using these parameters. The LPC residual is then low-pass filtered to obtain the base-band signal which is then decimated and quantized. At the receiver, the decimated base-band signal is upsampled by inserting zeros in between the samples (spectral folding) and then passed through the LPC synthesis filter to recover the output speech.

In a RELP coder the quality of the digital speech produced for a given bit rate depends on two processing stages, quantization of the base-band and high frequency regeneration. As in HTC, the width of the base-band, hence quantization performance of base-band for a given bit rate, and the performance of the high frequency regeneration are all dependent on each other. They all effect the overall speech quality produced by
Figure 7.14: A block diagram of RELP (a) encoder, (b) decoder.
the coder.

7.5.1.1 Quantization Of The Base-Band

After LPC inverse filtering the remaining residual signal is low-pass filtered (usually at 1 KHz) and down sampled to produce the signal input to the quantizer. Here, logarithmic, Max or uniform quantizers can be used to code the base-band signal. In simulations we have used uniform quantizers with block adaptive step sizes. Results of the quantizer performance with respect to the number of bits in the quantizer are tabulated in Table 7.16.

<table>
<thead>
<tr>
<th>Bits</th>
<th>Step-size</th>
<th>SNR (dB)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1.266</td>
<td>3.455</td>
</tr>
<tr>
<td>2</td>
<td>0.974</td>
<td>8.503</td>
</tr>
<tr>
<td>3</td>
<td>0.546</td>
<td>13.298</td>
</tr>
<tr>
<td>4</td>
<td>0.2871</td>
<td>17.294</td>
</tr>
</tbody>
</table>

Table 7.17: Quantization performance of 1 KHz base-band using uniform quantizers.

7.5.1.2 High Frequency Regeneration (HFR)

We have shown how pitch adaptive HFR can be performed in the HTC. The same technique can be implemented in the time domain and used in RELP. However, this will involve too much unnecessary computation. Here the simplest of all HFR techniques was used. Simply by inserting zeros after the received samples and passing them through the LPC synthesis filter produces the high frequencies by folding the base-band spectrum (see section 4.3). One problem with this method of HFR is that the pitch harmonics in the folded regions may be displaced, which causes tonal distortion in the recovered speech. One obvious cure for this problem is to make sure that the pitch harmonics are produced in the correct locations. This of course requires the calculation of the pitch period in the first place.
7.5.1.3 Pitch Filtering

In a RELP coder pitch filtering can be used to both improve the base-band quantization performance by reducing the dynamic range of the base-band signal and reduce the HFR distortion by locating the pitch harmonics in the correct positions. The pitch filter can either be used inside the base-band after the signal is decimated or outside the base-band before low-pass filtering. When used in the base-band, the pitch filter may help to improve base-band quantization but have no effect on the control of the HFR noise. Therefore it is better to use it outside the base-band, as shown in Figure 7.15.

When the pitch filter is used before low-pass filtering, the remaining signal has a bandwidth of 4 KHz (half the sampling frequency). When recovering the signal, the base-band signal is passed through the pitch synthesis filter before going through the LPC synthesis filter. This produces a better excitation residual with its pitch pulses located in the correct locations. In addition to locating the pitch pulses in the correct locations, its memory response replaces the zero valued samples before going through the LPC synthesis filter. This produces much more natural and smooth recovered speech.

During pitch synthesis filtering the full band excitation residual is obtained with only 1/4 of the information entering the filter (assuming a decimation factor of 4). Therefore the pitch synthesis filter is always in danger of becoming unstable. In order to make sure that the pitch synthesis filter is always stable, the pitch filter gain (coefficient) should always be kept at a value less than one.

7.5.2 Discussions

Without extensive simulations it was tried to investigate the performance of a base-band RELP coder. We have shown that the quality of the coder can be improved by a pitch filter. For high frequency regeneration spectral folding was used. This is the simplest HFR technique, but its quality depends on the talker. For a high pitched talker its quality is worse, because of broken pitch harmonics in the folded regions.

The overall quality of RELP using 3 bits/sample quantization of a 1 KHz base-band and pitch filtering before decimation was good. In fact for male talkers the quality is very similar to the original, but for female talkers tonal, as well as aliasing noise, is heard. In the simulations the following parameters were used:

10 LPC updated every 160 samples coded with 40 bits.
Figure 7.15: A block diagram of RELP with pitch prediction (a) encoder, (b) decoder.
Single tap pitch updated every 160 samples coded with 11 bits.

1 KHz base-band quantized using 3 bits/sample.

Block energy of the base-band updated every 160 samples coded with 6 bits

Overall bit rate = \((3 \times 2000) + (40 + 6 + 11)\)\(8000/160 = 8850\) bits/sec.

The RELP coder with above parameters was the simplest discussed in this chapter. Its quality can be improved by increasing its complexity. 10 LPC parameters may be vector quantized to save bits and increase the base-band width. When the saved bits were allocated for base-band (1 KHz) quantization no significant improvement was achieved. This means that 3 bits/sample quantization of the base-band signal is adequate. Improvements in quality may also be achieved by increasing the rate of update of the pitch filter parameters.

Delay in the coder is equal to the analysis block size which is 160 samples (20 msec) in this case. It is flexible but is not expected to be less than 16 msec.

7.6 Discussions

Various ways of reducing the bit rate down to the 12 to 9.6 Kb/s region have been discussed. Simulation results have confirmed that frequency domain speech coding below 16 Kb/s may still be superior to its time domain counterparts. A sub-band coder and transform coder have been simulated in various forms which has demonstrated the potential of these coders at even lower bit rates than 9.6 Kb/s. We have shown how VQ can be applied to code the SBC band energies or the LPC parameters to reduce the side information of a typical medium bit rate coder. Vector quantization was also used to code the residual signal. However, vector quantization of the residual has not been very successful simply because of complexity constraints. Large dimensions are required to reduce the bit rate of residual transmission which in turn requires large code-books and hence high complexity. In the context of VQ of the residual we have shown that the cross correlation error minimization \([9]\), although more complex, outperforms the simple matching process. When applied in the frequency domain no convolution is required which simplifies the search considerably.

Another conclusion drawn is that a coder such as SBC requires the inclusion of pitch modelling if the bit rate is to be further reduced. This was confirmed by transform coders which included pitch modelling. Most promising of the coders discussed in this chapter are in fact the transform coders with pitch modelling. Linear predictive coding
with vector quantization and frequency domain noise shaping has the potential to operate at bit rates around 4.8 Kb/s. Base-band coders also have the potential to operate at lower bit rates than 9.6 Kb/s. In the following chapter these two coders, together with some others will be optimized and modified to operate at lower bit rates than 9.6 Kb/s.

7.7 References


8 KB/S TO 4.8 KB/S CODERS

8.1 Introduction

There are several algorithms which can be used to code speech at rates of 8 to 4.8 Kb/s [1]. Although, they are given different names, the principle on which they are based are very similar. They all estimate and remove the correlation in the speech signal and then quantize and transmit the remaining (residual) signal. At the receiver, the removed correlation is introduced into the received residual with the help of model parameters, which are also transmitted by the encoder.

Estimation of speech parameters both in the time and frequency domain are well developed and currently in use. Therefore, to improve the coded speech quality, most of the recent research work has been concentrated on finding the best possible estimate of the residual signal. It may be possible to divide these coders into two groups in terms of the way in which they operate on the residual signal, i.e. the analysis and synthesis systems and the analysis by synthesis systems.

Analysis and synthesis systems include sub-band coders (SBC), adaptive transform coders (ATC), base-band coders (BBC), etc. These obtain the residual by an analysis procedure and then directly quantize and transmit this residual. During the quantization process the error between the residual and its quantized value is minimized. Hence, the quality of the synthesized speech is very much dependent on the accuracy of quantization of the residual signal. These coders, as seen in chapter 7, are capable of producing good quality speech at bit rates as low as 8 Kb/s with moderate complexity. Their quality deteriorates rapidly at bit rates below 8 Kb/s.

Analysis by synthesis systems on the other hand aim to replace the residual signal by a sequence of pulses which minimize the error between the original and the synthesized speech. Analysis by synthesis systems include multi-pulse excited linear predictive coders (MPLPC) and code excited linear prediction (CELP), both of which are capable of producing good quality digital speech in the region of 8 to 6 Kb/s. These coders aim to overcome the limitations of vocoders by replacing the existing excitation source by a sequence of pulses which are optimized either one by one for MPLPC or as a block for
CELP [2]. Both MPLPC and CELP type coders are very complex, because during the optimization of the excitation sequence long and exhaustive search is required. The most promising analysis by synthesis coder for low bit rates around 4.8 Kb/s is CELP. In this chapter we will discuss various ways of producing high quality speech in the 8 to 4.8 Kb/s region.

8.2 Code Excited Linear Prediction (CELP)

In chapter 4 we have briefly summarized the basic principles of a typical CELP coder. First 40 samples of the memory response of the recursive synthesis filter is subtracted from the original speech to produce a reference signal. Each sequence of 10 bit (1024 sequences) code-book is then scaled up by an optimum scale factor and filtered through the synthesis filter with its memory set to zero. The scale factors are calculated using equation (4.18). First 40 output samples of the synthesis filter are then compared with the reference 40 samples to produce an error signal. The error signal is further processed by an appropriate weighting filter, to produce subjectively meaningful error measurement. The sequence which produces the minimum weighted mean squared error is then selected (using equation (4.19)) and its code-book index is transmitted to the receiver which has an identical code-book. The optimum scale factors together with the parameters of the synthesis filter are also transmitted to reconstruct the same signal at the decoder.

The synthesis filter consists of two separate filters. The first is the pitch filter which introduces the fine structure to the code-book sequences and the second is the LPC filter which produces the spectral envelope. In order to operate at various bit rates between 8 and 4.8 Kb/s the rate at which the model parameters are updated together with the vector dimensions are modified accordingly. In the original proposed CELP [10], three tap pitch filter parameters were updated every 5 msec which means 200 times a second. Under these circumstances it is impossible to fit the total information rate into a 4.8 Kb/s transmission rate, even if a single tap pitch filter is used (which requires less bits to transmit). Assuming 5 bits are required to code the optimum scale factor, which is transmitted 200 times a second, and a 10 bit code-book the information rate, without the short term predictor (LPC) parameters will be \([5+10+11]\times 200 = 5200\) bits/sec, (11 bits being used to code the single tap pitch filter parameters). Here, we will show possible combinations of parameters to maximize the quality for 8 and 4.8 Kb/s.
8.2.1 8000 bits/sec CELP

Prior to quantization of the short and long term predictor (filter) parameters, initial tests were conducted to ascertain number of LPC coefficients which would be a compromise between quality and extra information rate. The results are shown in Table 8.1.

<table>
<thead>
<tr>
<th>LPC order</th>
<th>SNR (dB)</th>
</tr>
</thead>
<tbody>
<tr>
<td>6</td>
<td>0</td>
</tr>
<tr>
<td>8</td>
<td>0.90</td>
</tr>
<tr>
<td>10</td>
<td>1.23</td>
</tr>
<tr>
<td>12</td>
<td>1.36</td>
</tr>
</tbody>
</table>

Table 8.1: Performance of various LPC orders relative to 6 order.

These results in Table 8.1 show that an LPC order of 10 is a good choice.

8.2.1.1 Quantization Of Short And Long Term Predictor Parameters

In section 5.1.1 and 7.3.2.1.1 we have briefly discussed the ways of quantizing the LPC parameters. LPC parameters can be scalar or vector quantized. Although, vector quantization requires less information rate, for a given performance it is more complex. Therefore, it was decided to scalar quantize the LPC parameters in the form of log area ratios (LAR), see equation (5.10). A total of 40 bits were allocated to the 10 LARs as shown in Table 8.2.

<table>
<thead>
<tr>
<th>LAR</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bit</td>
<td>6</td>
<td>5</td>
<td>5</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
</tr>
</tbody>
</table>

Table 8.2: Bit allocation to 10 LARs.
In order to produce high quality digital speech it was decided to use a 3 tap pitch filter as explained in section 4.1.2. Coding of the filter parameters was performed as suggested in [3]. As with the LPC parameters, \( 3\beta \) values were first transformed as follows, to reduce the dynamic range of the coefficients, and hence, quantization noise.

\[
\begin{align*}
\beta_1 &= \beta_1 + \beta_2 + \beta_3 \\
\beta_2 &= \beta_1 - \beta_3 \\
\beta_3 &= \beta_1 + \beta_3
\end{align*}
\]

The bit assignment and the ranges of the transformed parameters \( b_1, b_2, b_3 \) are shown in Table 8.3.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Minimum</th>
<th>Maximum</th>
<th>Bits</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pitch</td>
<td>20</td>
<td>147</td>
<td>7</td>
</tr>
<tr>
<td>( b_1 )</td>
<td>-1.0</td>
<td>1.0</td>
<td>4</td>
</tr>
<tr>
<td>( b_2 )</td>
<td>-1.0</td>
<td>1.0</td>
<td>4</td>
</tr>
<tr>
<td>( b_3 )</td>
<td>-1.0</td>
<td>1.0</td>
<td>4</td>
</tr>
</tbody>
</table>

Table 8.3: Bit allocation to three tap pitch filter parameters.

The optimum scale factor for each block also needs quantization. Tests were carried out to find the number of bits required to quantize this factor without causing degradation in the overall speech quality. It was found that 6 bits were required to code its sign and magnitude without causing any noticeable degradation. The difference in using 6 bits or 5 bits was not very significant, which means that if required 1 bit saving per scale factor may be achieved.

8.2.1.2 Code-Book Generation

In the original design of CELP [2][10], the code-book for excitation sequences was populated with white Gaussian random numbers. Although, other alternative ways are now available [4], in our tests we have used white Gaussian random numbers to populate the excitation code-book. The reason for using white Gaussian random numbers to
represent the excitation sequences is that the residual signal of speech after LPC and pitch inverse filtering is assumed to be white Gaussian. In section 7.4.2 we have investigated the performance of code-books and found that provided the LPC parameters were updated about every 20 msec and pitch filter parameters about every 8 msec, Gaussian code-books performed as well as any other in representing the residual signal. In order to check the performance of Gaussian code-books with respect to their sizes, tests were carried out. In these tests LPC parameters were updated every 20 msec and pitch filter parameters every 5 msec with a 40 sample excitation vector dimension. Relative performances of various code-book sizes are tabulated in Table 8.4.

<table>
<thead>
<tr>
<th>Bit</th>
<th>SNR (dB)</th>
</tr>
</thead>
<tbody>
<tr>
<td>7</td>
<td>0</td>
</tr>
<tr>
<td>8</td>
<td>0.66</td>
</tr>
<tr>
<td>9</td>
<td>1.39</td>
</tr>
<tr>
<td>10</td>
<td>1.94</td>
</tr>
<tr>
<td>11</td>
<td>2.53</td>
</tr>
<tr>
<td>12</td>
<td>3.21</td>
</tr>
</tbody>
</table>

Table 8.4: Performance of Gaussian code-books relative to 7 bits.

Results in Table 8.4 show that a linear increase in the size of the code-book produced a steady increase in the overall SNR. For every bit increase in the size of the code-book, about 0.65 dB increase in the performance of the code-book was observed.

8.2.1.3 Simulations

A standard CELP at 8 Kb/s was simulated with the specific parameters given in Table 8.5.
Processed digital speech quality at 8 Kb/s was very good and could be considered transparent. During simulations male and female speech was mixed together and passed through the coder. Although, there was not any significant quality differences between male and female test sentences, when tested using highly sensitive ear-phone it was noticed that male speech contained just a little more roughness than the female speech. However, the speech quality of both male and female speech was very close to the original. Small differences between male and female speech was also confirmed by the objective segmental SNR calculations as tabulated in Table 8.6. In order to see how the coder would perform in idle sections, i.e. when there was no speech signal, segments of Gaussian signal with very small energy were inserted in between the sentences. It was found that idle sections did not disturb the performance of the speech quality produced by the coder.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Number</th>
<th>Update (msec)</th>
<th>Bits</th>
<th>Bit-Rate</th>
</tr>
</thead>
<tbody>
<tr>
<td>LPC</td>
<td>10</td>
<td>22.5</td>
<td>40</td>
<td>1777.8</td>
</tr>
<tr>
<td>Pitch</td>
<td>3-Tap</td>
<td>5.625</td>
<td>19</td>
<td>3377.8</td>
</tr>
<tr>
<td>Vector</td>
<td>1</td>
<td>5.625</td>
<td>10</td>
<td>1777.8</td>
</tr>
<tr>
<td>Scale</td>
<td>1</td>
<td>5.625</td>
<td>6</td>
<td>1066.7</td>
</tr>
</tbody>
</table>

Table 8.5: Parameters and bit rate allocation of CELP at 8 Kb/s.

<table>
<thead>
<tr>
<th>Segmental SNR (dB)</th>
<th>Male</th>
<th>Female</th>
<th>Overall</th>
</tr>
</thead>
<tbody>
<tr>
<td>11.47</td>
<td>12.51</td>
<td>11.96</td>
<td></td>
</tr>
</tbody>
</table>

Table 8.6: Segmental SNR performance of CELP at 8 Kb/s.
Table 8.7: Single tap pitch gain quantizer levels (4 bits)

<table>
<thead>
<tr>
<th>Input</th>
<th>Output</th>
<th>Input</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>0-0.1250</td>
<td>0.09375</td>
<td>0.6250-0.5625</td>
<td>0.59375</td>
</tr>
<tr>
<td>0.1875-0.1250</td>
<td>0.15625</td>
<td>0.6875-0.6250</td>
<td>0.65625</td>
</tr>
<tr>
<td>0.2500-0.1875</td>
<td>0.21875</td>
<td>0.7500-0.6875</td>
<td>0.71875</td>
</tr>
<tr>
<td>0.3125-0.2500</td>
<td>0.28125</td>
<td>0.8125-0.7500</td>
<td>0.78125</td>
</tr>
<tr>
<td>0.3750-0.3125</td>
<td>0.34375</td>
<td>0.8750-0.8125</td>
<td>0.84375</td>
</tr>
<tr>
<td>0.4375-0.3750</td>
<td>0.40625</td>
<td>1.0000-0.8750</td>
<td>0.95000</td>
</tr>
<tr>
<td>0.5000-0.4375</td>
<td>0.46875</td>
<td>1.2000-1.0000</td>
<td>1.15000</td>
</tr>
<tr>
<td>0.5625-0.5000</td>
<td>0.53125</td>
<td>&gt;1.2</td>
<td>1.3000</td>
</tr>
</tbody>
</table>

Table 8.8: General SegSNR performance of CELP.

In order to further evaluate the performance of CELP at around 8 Kb/s, its performance was tested with respect to the order of the pitch filter and the size of the code-book used in the coder. Results are tabulated in Table 8.8. When the single tap pitch filter was used the gain of the filter was quantized using 4 bits. Quantizer step sizes are given in Table 8.7.
8.2.2 4800 bits/sec CELP

We have shown in the previous section that the quality of a standard CELP is comparable to the original speech. However, it can only reduce the bit rate down to around 7 Kb/s. The reason for this is that the update rate of the model parameters (LPC and pitch) has to be frequent enough to improve quality, and consequently requires the transmission of much side information. In the previous section and also in section 7.4.2 we have shown that the update rate of pitch filter parameters is more important than the rate at which the LPC parameters are transmitted. In view of these results adjustment should be made to the overall bit rate of CELP in order to bring the total bit rate down to 4.8 Kb/s without causing much degradation to the speech quality.

During the adjustment of the dimensions of CELP to operate at 4.8 Kb/s it was assumed that the minimum rate at which the LPC parameters should be transmitted was every 256 samples (32 msec), which is the maximum time width that the speech was assumed to be stationary. Using 10 LPC parameters with 40 bits every 256 samples requires 1250 bits/sec, leaving 3550 bits/sec to transmit both pitch and excitation vector parameters. Assuming that the excitation vector rate is equal to the pitch parameter rate and also assuming 19 bits for pitch filter parameters, a 10 bit code-book and 5 bits for the optimum scale value, simple calculation shows that the minimum vector dimension should be 77 samples long. A CELP coder was tested with the above assumed dimensions. The processed speech was found to be very much worse than the original quality. It had excess amounts of quantization noise making the speech rough. Another important result observed was that the signal level (energy in the processed speech) was considerably lower than the 8 Kb/s CELP output which was very close to the original signal energy. The reason for this is the optimum scale calculation using equation (4.18). As the vector size tends to a larger value the average value of the term $\sum x_n f_n$ tends to a small value which makes the optimum scale value small. Hence, the output signal level drops below the original. This was expected because as the dimension of the excitation vector increases the correlation between its output response $f_n$ and the reference signal $x_n$ tends to reduce. Poor performance was also the result of less accurate pitch prediction. In order to assess the effect of the pitch prediction and excitation vector dimension on the speech quality separately, the available 3550 bits/sec were distributed amongst the pitch filter parameters and the excitation vector parameters as follows. The rate at which pitch parameters were updated was set to every 128 samples and gradually reduced to every 64 samples. Vector dimension on the other hand had an initial size of 50 samples which
was gradually increased to 100 samples. Results showed that the increase in the size of the vector dimension caused more distortion than the gain achieved by better pitch prediction. This again was not unexpected because poor quantization also reduces the efficiency of prediction in the synthesis filter.

8.2.2.1 Simulations

After initial simulations a 4.8 Kb/s CELP coder was finally simulated with the parameters given in Table 8.9. In order to save bits update rate of LPC, pitch and excitation vector parameters were reduced. To further save bits the optimum scale value was coded with 5 bits and the pitch parameters $b_2$ and $b_3$ were allocated 3 bits each.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Number</th>
<th>Update (msec)</th>
<th>Bits</th>
<th>Bit-rate</th>
</tr>
</thead>
<tbody>
<tr>
<td>LPC</td>
<td>10</td>
<td>31.25</td>
<td>40</td>
<td>1280</td>
</tr>
<tr>
<td>Pitch</td>
<td>3-Tap</td>
<td>15.625</td>
<td>17</td>
<td>1088</td>
</tr>
<tr>
<td>Vector</td>
<td>1</td>
<td>6.25</td>
<td>10</td>
<td>1600</td>
</tr>
<tr>
<td>Scale</td>
<td>1</td>
<td>6.25</td>
<td>5</td>
<td>800</td>
</tr>
</tbody>
</table>

Table 8.9: Parameters and bit rate allocation of CELP at 4.8 Kb/s.

The coded speech quality was not as good as the original speech. In fact quantization noise could still be perceived, even when noise shaping was used. Reducing the bits for the optimum scale was also one reason for slight roughness. Although, processed speech had no clicks and other annoying noise the overall speech quality may not be acceptable for telephone systems. Objective performance of the processed speech, which does perhaps not mean much, also confirmed the reduced quality in reducing from 8 Kb/s to 4.8 Kb/s. These are tabulated in Table 8.10.
Replacing the three tap pitch filter with a single tap filter and hence using a greater update rate did not improve the quality of the processed speech. This was because large quantization errors reduced the accuracy of a single tap pitch filter more than the accuracy of a three tap pitch filter. One possible method of improving the quality of processed speech may be achieved by minimizing the error between the predicted value of the pitch synthesis filter. Although, a pitch inverse filter is not used in CELP, the parameters of the pitch filter are calculated by minimizing the mean squared error between the LPC residual and its pitch predicted value.

### 8.2.2.2 Efficient Pitch Filter Implementation

Consider the example in Figure 8.1. In Figure 8.1, if the quantizer is considered such that there is no quantization error, \( e_q = 0 \) then,

\[
\hat{r}(i) = \hat{r}(i) \tag{8.1}
\]

and,

\[
P(i) = \hat{P}(i) \tag{8.2}
\]

However, the quantizer in CELP is far from being perfect and \( e_q \) is not zero. Therefore,

\[
\hat{r}(i) = [r(i) + e_q] + \hat{P}(i) \tag{8.3}
\]

and,

\[
\hat{P}(i) = P(i) + e_p \tag{8.4}
\]
Figure 8.1: A block diagram of independent inverse and synthesis filtering.

Figure 8.2: A block diagram of combined inverse and synthesis filtering.
where \( e_p \) is the error between the predicted value of pitch filter at the inverse and synthesis filter caused by the error in the previous block of output samples. Therefore,

\[
  r(i) - \hat{r}(i) = e_q + e_p
\]  

\( e_q \) depends on the quantizer and signal characteristics which we will assume to be fixed, \( e_p \) on the other hand is a function of previous \( e_p + e_q \) (depending on pitch period). If \( e_p \) can be set to zero then the difference between \( r(i) \) and \( \hat{r}(i) \) will only be \( e_q \). This can only be achieved if \( P(i) \) and \( \hat{P}(i) \) are made equal. Consider another example in Figure 8.2.

The configuration shown in Figure 8.2 eliminates \( e_p \) by using the same predicted signal in inverse filtering as in the synthesis filter. One problem with this configuration is that the pitch coefficients cannot be optimally calculated. If the coefficients are calculated using the unquantized LPC residual, \( r(i) \), \( \hat{P}(i) \) will have errors because \( \hat{P}(i) \) is determined by the quantized values of the LPC residual. If both \( r(i) \) and \( \hat{r}(i) \) are used to calculate the coefficients we are faced with another problem, that if the window size \( N \) is greater than the pitch period, we will need the quantized values of \( r(i) \) for \( p \leq i \leq N \) (\( p \) is the pitch period). There are two solutions to this problem. One is to limit the maximum window size \( N \) to minimum expected pitch period \( p \). However, assuming that the minimum pitch period is 20 samples, then the update of pitch filter parameters may be much more frequent than is necessary. The other solution is to limit the minimum pitch period to window size \( N \). This solution looks attractive as long as \( N \) is not too large to reduce the pitch filter effect (40 to 80 samples). The second version of the regular pulse excited linear prediction, which has been adopted as the GSM 16 Kb/s speech algorithm for mobile applications, uses a similar pitch filter to enhance the accuracy of the excitation pulses. In such systems (analysis and synthesis coders) decoding of the residual signal and the synthesis of the pitch filter is necessary to produce the \( \hat{r}(i) \)'s. However, in CELP (analysis by synthesis coders) no extra complexity is required.

For a single tap pitch filter; \( p = N \), and for a three tap pitch filter; \( p = N + 1 \). Using the above solution to pitch filter implementation the CELP coder was tested at both 8 and 4.8 Kb/s. Overall segmental SNR of both coders were increased by about 0.64 dB. However, when informal listening tests were conducted the improvement in quality was much more significant than the 0.64 dB reflected. Another and more important result was
that when the coders were tested using only 8 bit code-books the improvement in segmental SNR was 0.84 dB, higher than the 10 bit code-book. This showed that by eliminating $e_p$ and improving pitch prediction reduced the difference between the original speech and the synthesis filter memory. This of course means smaller signal energy is required in the excitation sequence and hence reduces the quantization noise. This was confirmed by measuring the SNR between the original input speech and the synthesis filter memory response. The results are tabulated in Table 8.11 and various corresponding waveforms are plotted in Figure 8.3.

<table>
<thead>
<tr>
<th>Memory Prediction (dB)</th>
</tr>
</thead>
<tbody>
<tr>
<td>8 Kbps</td>
</tr>
<tr>
<td>Pitch Tap</td>
</tr>
<tr>
<td>1</td>
</tr>
<tr>
<td>3</td>
</tr>
</tbody>
</table>

Table 8.11: Segmental prediction gain of CELP at 8 and 4.8 Kb/s.

Although the new pitch filter configuration improves the prediction and hence the overall speech quality, quantization noise and roughness could still be heard at low bit rates. The main reason for this is the rate at which the pitch parameters were updated. The difference in prediction for 8 and 4.8 Kb/s is about 2 dB. Another way of further improving the performance of CELP at 4.8 Kb/s is to find a means of updating the pitch filter parameters more frequently. This can only be done if savings in coding the pitch filter parameters and possibly the other parameters are made. This can be achieved by vector quantization of both LPC parameters and the three tap pitch filter coefficients.

8.2.2.3 Vector Quantization Of Short And Long Term Filter Parameters

We have made two attempts to vector quantize the LPC (short-term) parameters in sections 7.3.2.1.1 and 7.4.2. In the first attempt a 10 bit full search code-book with Itakura-Saito distortion measure was employed and in the second one, two cascaded full search code-books, one with 9 and the other 8 bits were used. In cascaded quantization
Figure 8.3: Typical waveforms of speech signals in CELP, (a) original, (b) 8 Kb/s CELP, (c) 4.8 Kb/s CELP, (d) filter memory response in 8 Kb/s CELP and (e) filter memory response in 4.8 Kb/s CELP.
the first code-book was trained using LAR's and the second code-book was populated with the error signal of the first stage. Both code-books employed the simple mean squared error measure. Results showed that if relatively good performance is the goal of vector quantization of LPC parameters as well as bit saving, the size and complexity of the code-books will be huge. In order to clarify this we briefly explain the function of the Itakura-Saito distortion measure. Itakura-Saito distance between two LPC vectors is given by,

\[ d \left( A, A, \right) = \frac{\left\| A, I, A, F \right\|}{\left\| A, I, A, F \right\|} - 1 \]  

where \( A \) and \( A, \) are the original and code-book LPC vectors and \( R_A \) is the autocorrelation matrix from which the original parameters \( A \) are calculated. In equation (8.6) terms \( \left\| A, I, A, F \right\| \) and \( \left\| A, I, A, F \right\| \) correspond to the LPC residual energy filtered by the code-book vector parameters and the original vector parameters respectively. Therefore, unless \( \left[ A, I, A, F \right] \equiv \left[ A, I, A, F \right] \) the ratio \( \left[ A, I, A, F \right] / \left[ A, I, A, F \right] \) will always have a value greater than unity. When the two earlier attempts are applied to coding of the LPC parameters of CELP, results showed that although fewer bits were used, more distortion could be heard. This was because, when the spectral parameters have large quantization errors they also affect the chosen excitation vector and hence cause more errors. There are several other new vector quantization techniques reported in the literature \[4\][5][6]. Their reported performances are good. However, here we will scalar quantize the LPC parameters and try to vector quantize the 3 tap pitch filter coefficients as a 3 dimensional vector.

The performance advantage of a three tap pitch filter over a single tap pitch filter is obvious. At low bit rates, therefore, it is almost essential to have a three tap pitch filter if good quality digital speech is to be achieved. However there are two major problems with the three tap pitch filter. Firstly, it requires more bits (about 4 bits for each transformed coefficient, \( b_1, b_2, b_3 \)) for transmission and hence increases the bit rate of the coder. Secondly, as was reported earlier \[7][8]\ it sometimes becomes unstable in the synthesis filtering. In order to maintain stability some correction terms have been inserted into its matrix solution \[7][8]. The coding capacity required by three tap pitch filter can be reduced by using vector quantization to code the three pitch filter coefficients as one vector. Vector quantization has an additional advantage over the scalar quantization of filter coefficients. By eliminating the unstable filter parameters from the code-book the
stability of the synthesis filter is guaranteed. The sufficient condition for stability is that
the sum of the absolute values of the 3 pitch filter coefficients should always be less than
unity \([9], (b_1 < 1)\).

Various size code-books have been simulated and compared with the scalar quanti-

zation performance. During performance comparison the prediction gain for both scalar
and vector quantization were computed as,

\[
G_{pr} = 10 \log_{10} \frac{\sum_{i=1}^{N} x^2(i)}{\sum_{i=1}^{N} y^2(i)}
\]

(8.7)

where \(x(i)\) is the LPC residual signal and \(y(i)\) is the pitch inverse filtered \(x(i)\). Dur-

ing scalar quantization transformed pitch coefficients \(b_1, b_2, b_3\) were quantized using 4 bits
each (see section 8.2.1.1). Results are tabulated in Table 8.12. Results of SNR's are given
relative to the scalar quantization.

<table>
<thead>
<tr>
<th>VQ (bits)</th>
<th>Relative SNR (dB)</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>-1.9</td>
</tr>
<tr>
<td>5</td>
<td>-0.8</td>
</tr>
<tr>
<td>6</td>
<td>-0.3</td>
</tr>
<tr>
<td>7</td>
<td>-0.1</td>
</tr>
</tbody>
</table>

Table 8.12: Prediction performance of VQ of 3 tap pitch filter
relative to 12 bits scalar quantization.

Results in Table 8.12 show that 6 or 7 bit code-books have very similar performance to
the scalar quantization case, with 6 and 5 bit savings respectively.

Using 6 bits to code the three pitch filter coefficients of CELP at 4.8 Kb/s two final
tests were conducted. In the first test the excitation vector size was assumed to be the
block size of pitch update and in the second test, the pitch update block was twice that
of excitation vector size. Results of both tests are tabulated in Table 8.13.

<table>
<thead>
<tr>
<th>Tests</th>
<th>Male</th>
<th>Female</th>
<th>Overall</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>7.5</td>
<td>8.3</td>
<td>7.8</td>
</tr>
<tr>
<td>2</td>
<td>7.7</td>
<td>8.5</td>
<td>8.0</td>
</tr>
</tbody>
</table>

Table 8.13: Segmental SNR performance of CELP with vector quantized pitch parameters at 4.8 Kb/s.

Simulated parameters of CELP in both tests are given in Table 8.14.

<table>
<thead>
<tr>
<th>Tests</th>
<th>First-Test</th>
<th>Second-Test</th>
</tr>
</thead>
<tbody>
<tr>
<td>Parameter</td>
<td>Bits</td>
<td>Update (msec)</td>
</tr>
<tr>
<td>LPC</td>
<td>40</td>
<td>32</td>
</tr>
<tr>
<td>Pitch</td>
<td>13</td>
<td>8</td>
</tr>
<tr>
<td>Gain</td>
<td>5</td>
<td>8</td>
</tr>
<tr>
<td>Vector</td>
<td>10</td>
<td>8</td>
</tr>
</tbody>
</table>

Table 8.14: Simulation parameters of CELP in two test cases at 4.8 Kb/s with vector quantized pitch coefficients.

The segmental SNR increases in Table 8.13 show that the digital speech quality is increased by vector quantizing the pitch parameters and hence updating them more often. Although, there were some differences in SNR of the two test cases the overall speech quality of both was very close. Because the pitch parameters were updated more often compared with the scalar quantized case, pitch prediction was improved and hence quantization noise was reduced. Subjective listening tests showed that the quality of
CELP with vector quantized pitch parameters was significantly better. Less quantization noise was audible.

8.2.3 Complexity Consideration Of CELP

So far, we have discussed the ways that the quality of CELP at low bit rates can be improved. Although, CELP seems to be the most promising coding technique for digital speech transmission at around 4.8 Kb/s, its very high complexity is a big disadvantage. Standard CELP as proposed in [10] requires 500 MIPS which makes it impossible to be implemented using current DSP chips. About 98% of CELP complexity is required during the code-book search. During the code-book search, the output response of each excitation sequence filtered through the recursive synthesis filter is calculated and then cross correlated with the reference signal to find the best matching excitation sequence. During filtering, convolution operations are required which are the main cause of high complexity. In the literature two types of simplification procedures have been suggested. One type of simplification assumes random Gaussian code-books and tries to simplify the convolution computations. The other simplification strategy tries to design structured code-books so that the search of the code-books becomes much simpler. In [11], Trancoso and Atal suggest 3 major simplification procedures for searching the Gaussian code-books. They suggest, singular-value decomposition, autocorrelation approach and frequency domain search, all of which are aimed at reducing the computation required by the filter convolution processes. In [12], Davidson and Gersho, and in [13][14], Adoul and others suggest structured code-book designs which may not be fully searched. Here, we will introduce two new simplification procedures in order to yield a real-time implementable CELP coder.

8.2.3.1 LPC Residual Matched Code-Book Search

A block diagram of the LPC residual matched CELP coder is shown in Figure 8.4. In CELP the mean squared error (weighted) is minimized between the original speech and the synthesized speech. Here as shown in Figure 8.4 we have tried to minimize the error between the LPC residual and the synthesized pitch residual. Rewriting equations (4.18) and (4.19),

\[ \alpha_k = \frac{\sum_{n=1}^{N} x_n f_n}{\sum_{n=1}^{N} f^2_n} \]  

(8.8)
Figure 8.4: A block diagram of LPC residual matched CELP.
In equations (8.8) and (8.9) $x_n$ represents the LPC residual with the pitch filter memory subtracted from it (reference signal) and $f_n$ is the response of the code-book sequences at the output of the pitch synthesis filter. Here output response of the pitch synthesis filter can be written in terms of the code-book sequences and the filter impulse response (truncated).

\[
E_k = \sum_{n=1}^{N} x_n^2 - \frac{(\sum_{n=1}^{N} x_n f_n)^2}{\sum_{n=1}^{N} f_n^2}
\]  

(8.9)

\[f_n = V_n(i) * P_F(i)\]  

(8.10)

$V_n(i)$ is the $n^{th}$ sequence of the unit variance Gaussian code-book and $P_F(i)$ is the pitch synthesis filter truncated impulse response. One important point to note here is that when the filter memory of the pitch synthesis filter is set to zero, $P_F(i)$ will have the first $p$ values set to zero, where $p$ is the pitch period and represents the delay in the filter. Therefore, computation of $f_n$ will involve only $N-p$ impulse response values of $P_F(i)$. In cases when $p \geq N$, $P_F(i)$ will have no effect on $V_n(i)$, i.e. $P_F(i)$ will contain zeros for $N$ values and $V_n(i)$ will be directly equal to $f_n$. Using the pitch filter discussed in section 8.2.2.2 we can limit minimum $p$ to be equal to $N$ and hence eliminate all the convolutions required. Equations (8.8) and (8.9) then becomes,

\[
\alpha_k = \frac{\sum_{n=1}^{N} x_n V_n}{\sum_{n=1}^{N} V_n^2}
\]  

(8.11)

\[
E_k = \sum_{n=1}^{N} x_n^2 - \frac{(\sum_{n=1}^{N} x_n V_n)^2}{\sum_{n=1}^{N} V_n^2}
\]  

(8.12)
If $V_n$ is a unit energy sequence then the terms $\sum_{n=1}^{N} V_n^2$ need not be calculated, which then leads to,

$$\alpha_k = \sum_{n=1}^{N} x_n V_n$$  \hspace{1cm} (8.13)$$

$$E_k = \sum_{n=1}^{N} x_n^2 - (\sum_{n=1}^{N} x_n V_n)^2$$  \hspace{1cm} (8.14)$$

Using the above solution the search of the code-book is reduced to about $2^b$ multiply-add operations per sample, where $b$ is the number of bits in the code-book.

By employing the above simplification for code-book searching, CELP at 8 and 4.8 Kb/s was tested. The subjective quality of CELP at around 8 Kb/s was not affected significantly. Although, slight tonal distortion as in RELP was heard, the overall simplification is worth this small loss of quality. However, at 4.8 Kb/s, as there was increased amounts of quantization and prediction error, the distortion barely heard at 8 Kb/s CELP was clearly audible at 4.8 Kb/s. Distortion could be reduced by updating pitch parameters more often but it led to higher overall transmission rates.

### 8.2.3.2 Multiple Gain Excitation Vector Error Minimization

In CELP each vector is associated with an optimum scale or gain factor. The two functions of this gain are that it first of all determines the sign of each sample in the excitation vector and takes into account the effect of the expected noise power in the synthesis. Therefore, if the number of gain factors are equal to the number of elements in each vector, then the error in the synthesis will be zero, because each value of the excitation sequence will be scaled to the optimum amplitude. This tells us that as the number of gain values are increased in a vector the performance of the coder will increase for any given code-book size. Therefore, one solution for code-book search simplification may be to use multiple gain vectors and decrease the size of the code-book. Similar conclusions and results were given in a recently published paper [15].

Here multiple gain errors can be written as,

$$E_k = \sum_{n=1}^{N_1} [x_n - \alpha_1 f_n] + \sum_{n=N_1+1}^{N_2} [x_n - \alpha_2 f_n] + ...$$  \hspace{1cm} (8.15)$$
where \( N_1 \) and \( N_2 \) are the boundaries of vector elements with which \( \alpha_1 \) and \( \alpha_2 \) are associated. Optimum gain values \( \alpha_i \)'s are calculated in the same way as before.

\[
\alpha_i = \frac{\sum_{n=N_i}^{N_i} x_n f_n}{\sum_{n=N_i}^{N_i} f_n^2}
\]

(8.16)

where \( N_j \) and \( N_i \) are the boundaries of elements that \( \alpha_i \) will apply. A sequence which maximizes the total correlation is selected as the optimum.

\[
E_k = \text{MAX} \left[ \frac{\left( \sum_{n=N_1}^{N_1} x_n f_n \right)^2}{\sum_{n=1}^{N_1} f_n^2} + \frac{\left( \sum_{n=N_1+1}^{N_2} x_n f_n \right)^2}{\sum_{n=N_1+1}^{N_2} f_n^2} + \ldots \right]
\]

(8.17)

Although, the multiple gain solution looks attractive in terms of simplification and also increasing quality for a given code-book, it requires extra capacity to code these multiple gains. Tests were carried out with 2 and 3 gains. The 2 gain vector representation with 7 or 8 bit code-books resulted in the same performance as single gain with 10 bit code-book. The 3 gain vector representation further reduced the code-book size to 6 or 7 bits. In [15] vector quantization of the gain factors have been used. However, we feel that more than 2 gain element vectors are difficult to vector quantize and to save bits, because the block length that they operate on becomes smaller as the number of gains increases, and consequently makes the dynamic range of elements in each vector greater. However, for further simplifications of CELP to those discussed in [11] the multiple gain approach may provide an alternative.

8.2.4 Discussions

Here, we have investigated the results of CELP operating at 8 and 4.8 Kb/s. At around 8 Kb/s transparent speech can be produced using standard CELP with an overall segmental SNR of about 12 dB. When the processed speech is compared with the original input using highly sensitive ear-phones no significant difference could be
detected. At 4.8 Kb/s on the other hand, speech quality was rough. Processed speech had large quantization errors. This is also reflected by the segmental SNR of less than 8 dB. The two major causes of quality degradation in CELP at 4.8 Kb/s are the rate at which the pitch filter parameters are updated and the large dimensions of excitation vectors. When the pitch filter parameters are updated less frequently, which is necessary to reduce the bit rate, pitch prediction gain falls. Smaller prediction gain results in larger energy in the reference signal and consequently causes larger quantization errors. Large excitation vector dimensions also contribute to the overall quantization noise, causing low signal level and roughness in the output digital speech.

CELP at 4.8 Kb/s can be improved by using VQ to quantize both the short and long term filter parameters. The rate at which the LPC parameters are updated is about 4 to 6 times less frequently than that of pitch filter parameters. Therefore, although any saving is useful at low bit rates, savings that can be made by vector quantization of the LPC parameters is not significant. For this reason in our simulations LPC parameters were scalar quantized. However, we have used VQ to quantize the 3 pitch coefficients of a 3 tap pitch filter. We have found that with 6, or maximum 7 bit code-books good performance of the pitch filter can be maintained. Saving of about 5 bits per pitch filter parameters update allowed us to to update the pitch parameters more often or to reduce the excitation vector size which improved the overall segmental SNR by about 0.3 to 0.5 dB. In section 8.2.2.2 we have also given a better method of pitch filter modelling. Although, in CELP there is no inverse filtering, when modelling the pitch filter parameters inverse filtering is assumed. The improvement of about 0.5 to 0.6 dB was due to optimum pitch coefficients calculations. Using the new pitch filter configuration any prediction gain achieved is not affected by the quantization noise of the current block of samples, and hence the achieved prediction gain is directly reflected at the output of the pitch synthesis filter. If the new pitch filter is used in coders where there is inverse filtering a larger increase in performance would be expected.

The overall performance of CELP can be further improved by vector quantization of LPC parameters and saving bits, but, it is not expected to yield toll quality at 4.8 Kb/s. Toll quality can be achieved at bit rates as low as about 6 Kb/s. Major limitations to the quality of CELP below 6 Kb/s is the lower pitch filter parameters update and larger excitation vector dimensions.

The complexity of CELP is another problem that seems to require solution. We have mentioned some results reported in the literature and also proposed two further
simplifications methods. Standard CELP requires about 500 MIPS but single chip implemented CELP with structured code-books have been reported in the literature, [13][14][12].

In the first of the following two sub sections of this chapter we will discuss a transform coder based on the principles of CELP which requires only about \( 2^b \) (\( b \) is the number of bits in the code-book) instructions per sample. Assuming a 10 bit code-book and 8 KHz sampling frequency the overall complexity will be in the region of 10 to 12 MIPS. Results obtained from the transform coder will then be directly compared with CELP at both 8 and 4.8 Kb/s.

In the final part of this chapter, a new base-band coder, which, again uses CELP principles to code the base-band signal will be discussed. Comparison will then be made with both the CELP and the transform coder at 8 and 4.8 Kb/s, in terms of quality and complexity.

8.3 Vector Quantized Transform Coder

The transform approach to speech coding has been established for some time and has been shown to be very efficient in controlling the bit allocation and the shape of the noise spectrum [16][17]. In chapters 6 and 7, we have designed and simulated various transform coders which produce high quality digital speech in the 16 to 9.6 Kb/s region. Although, these coders can maintain good quality down to about 9.6 Kb/s, they perform poorly at the lower bit rates. In section 7.4 we have discussed the performance of a new transform coder where the linear prediction residual was vector quantized using weighted mean squared error distortion measure. This technique was found to be capable of producing high quality speech at bit rates as low as 9.6 Kb/s. At rates below 9.6 Kb/s, coder performance gradually deteriorated. In fact the speech quality produced at around 7 Kb/s was unacceptable for telephone quality. We have suggested three more factors to be considered in the coder to improve its quality. These are efficient pitch filter implementation, consideration of the LPC filter gain and finally the effect of the synthesis filters memory. These are the reasons why CELP has performed better than any other coder at bit rates around 8 Kb/s and below. In the following sections we discuss the ways that the above mentioned improvements can be applied to enhance the quality of a vector quantized transform coder whilst keeping its complexity within the limits of current DSP capabilities.
8.3.1 Coder Description

A block diagram of the new vector quantized transform coder (VQTC) [21], is shown in Figure 8.5. First speech is analysed to calculate 10 linear prediction parameters. Quantized values of these LPC coefficients are then used to inverse filter the block of speech. The LPC residual signal is then used to detect the pitch filter parameters, which are used to pitch inverse filter the LPC residual to remove the remaining long term correlation. The remaining residual signal is frequency transformed using suitable size discrete cosine transform (DCT). The size of the DCT depends on the residual vector size which in turn depends on the specific bit rate for which the coder is designed. Using a suitable size FFT on the LPC filter impulse response, the envelope of the current block of speech is obtained. This is the transform approach of obtaining the speech envelope. Suitable filter bank can also be used to obtain a reasonable estimate of the speech envelope which is called the sub-band approach. Each vector of the transformed residual signal is then coded by minimizing the envelope weighted distance from a unit variance Gaussian code-book. Memory response of the synthesis filters clocked with zero value input is then subtracted from the original signal to produce the difference (reference) signal to be matched, as for CELP [10]. The single vector that was earlier selected is then used to produce the output synthesized signal, which is compared with the reference signal in order to calculate the optimum gain. The resultant decoder is fairly simple. The chosen sequence is scaled up by the optimum scale factor and filtered through the synthesis filters to recover the output speech. In the encoder two code-books are stored. One stores the time domain and the other the frequency domain representations of the residual sequences. Therefore, when encoding, only one DCT per residual vector is computed and the need for IDCT is eliminated. In the decoder no transformation is required for the code-book search. Only the time domain representations of the sequences are stored which make the decoder extremely simple.

8.3.1.1 Quantization And Implementation Of Short And Long Term Filters

Quantization of 10 LPC parameters was performed using 40 bits as was done in CELP. Quantization of the pitch filter parameters (single or three tap) was again performed as in CELP.

Implementation of the pitch filter as discussed in section 8.2.2.2 was applied. Here unlike CELP, there are two advantages to be gained from the new pitch filter. One is the optimal calculation of the pitch filter parameters which was the only gain achieved in
Figure 8.5: A block diagram of VQTC.
CELP. The other is that in VQTC it is necessary to have a pitch inverse filter and the use of the same predicted value at both inverse and synthesis filtering ensures the prediction gain achieved at the inverse filtering is not affected by the quantization noise of the current residual vector. The quantization noise of the current residual vector can affect the prediction of the next vector of residual samples. However, when calculating the pitch filter parameters of the next vector residual samples the quantization noise of the current vector is known and hence the effect of quantization noise is minimized by adjusting the parameters accordingly.

8.3.1.2 Vector Quantization And Noise Shaping Of The Residual Vectors

In recently developed coders for use at 8 Kb/s and below (MPLPC and CELP) the residual signal is quantized in an analysis by synthesis procedure which is extremely complex [10]. The reason for the analysis by synthesis coding is to consider the effect of the LPC synthesis filter on the coded residual in terms of its filter gain, spectral shape and memory carried over from the previous block. This makes it possible to compare the original speech and the coded speech rather than comparing the residual vectors with the code-book entries.

Here we have overcome these complexities by considering the spectral shape of the LPC filter whilst code-book searching and considering the memory and the gain of the synthesis filter whilst calculating the amplitude scale factor of the chosen vector. Errors are minimized between the residual vectors and code-book entries as,

$$ E_r = \sum_{i=1}^{N} [x(i) - C V(i)]W(i) $$

(8.18)

where $x(i)$ and $V(i)$ are the unit variance transformed residual and code-book vectors respectively, and $W(i)$ is the noise shaping vector (spectral envelope). In equation (8.18) factor $C$ is a measure of correlation between $x(i)$ and $V(i)$ without the effect of the weighting vector $W(i)$ and is given by,

$$ C = \sum_{i=1}^{N} x(i)V(i) $$

(8.19)
As can be seen from equation (8.18) all convolution processes needed in the search of CELP are replaced by multiplications. After this very much simplified search, optimum amplitudes are calculated as follows: Firstly the memory of the synthesis filter is subtracted from the original speech, as this cannot be changed. Then the chosen code-book sequence (time domain equivalent) is used to produce the output response, which is then compared with the memory subtracted original as,

\[
a_n = \frac{\sum_{i=1}^{N} S(i)P(i)}{\sum_{i=1}^{N} P^2(i)}
\]  

(8.20)

where \(a_n\) is the optimum amplitude scale, \(S(i)\) and \(P(i)\) are the reference vector and the output response vector produced by the selected unit variance sequence.

8.3.2 8 Kb/s Vector Quantized Transform Coder

Like CELP, VQTC was simulated at 8 and 4.8 Kb/s. Before simulating the complete VQTC, tests were conducted to determine the noise shaping or the weighting vector. There are three ways of forming a weighting vector. Using the original data and suitable size FFT, using the original speech data and suitable filter bank or applying FFT to the impulse response of the LPC inverse filter. The resolution of the weighting vector depends on the residual vector size and hence the overall bit rate of the coder. At 8 Kb/s expected vector size is about 32 samples. This size is expected to be doubled at 4.8 Kb/s. Speech is assumed to be stationary up to about 32 msec (256 samples). Stationary in this case means that speech spectral characteristics do not change significantly within 32 msec. Therefore, calculating the speech envelope for every 32 or 64 samples is not necessary. In order to find the speech envelope using the original input data, an FFT or filter bank can be used. If an FFT is used the size of the FFT should be as large as the block length of the data, which is about 256 samples. Resulting spectral coefficients are then decimated by averaging the neighbouring values to obtain as many points on the spectrum as the size of the excitation vectors to be used. This solution will be costly because of the size of the FFT required. Another solution is the filter bank implementation. As with the sub-band coder a filter bank may be used to split the signal into 16 or 32 bands, and each band energy will give a point on the spectrum. For use in cases where the
residual vector dimensions are larger than the number of sub-bands, up-sampled weighting vectors are used, i.e. each value of the weighting vector is used to weight the corresponding two or more elements of the residual vector. Finally, FFT can be applied to the truncated impulse response of the LPC inverse filter. The size of the FFT in this case needs not be equal to the data block, but is not expected to be less than 128 points. The complexity of this approach is much simpler than applying FFT directly to the speech data. Only the first 11 values of the impulse response of 10 tap LPC filter is non-zero, the rest of the values are all zeros. Therefore, only an 11 by 128 matrix calculation is required. As the size of the residual vector increases, the accuracy of the envelope increases. The sub band approach also has moderate complexity but its accuracy is usually limited by the number of bands. For a 32 element residual vector, which is the expected vector size at 8 Kb/s, a 16 band sub-band approach produces reasonable noise shaping. However, at lower bit rates, where the vector size of the residual signal is expected to be of the order of 64 samples, a sub-band approach with only 16 bands is not as good as the FFT approach using the LPC filter impulse response. Therefore, in the following simulations the FFT approach using the LPC filter impulse response will be used.

8.3.2.1 Simulations

An 8 Kb/s VQTC was simulated with the specific parameters given in Table 8.15.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Number</th>
<th>Update (msec)</th>
<th>Bits</th>
<th>Bit-rate</th>
</tr>
</thead>
<tbody>
<tr>
<td>LPC</td>
<td>10</td>
<td>32</td>
<td>40</td>
<td>1250</td>
</tr>
<tr>
<td>Pitch</td>
<td>1-Tap</td>
<td>4</td>
<td>11</td>
<td>2750</td>
</tr>
<tr>
<td>Vector</td>
<td>1</td>
<td>4</td>
<td>10</td>
<td>2500</td>
</tr>
<tr>
<td>Scale</td>
<td>1</td>
<td>4</td>
<td>6</td>
<td>1500</td>
</tr>
</tbody>
</table>

Table 8.15: Bit allocation of VQTC at 8 Kb/s.

The parameter specifications given in Table 8.15 are only one possible combination, other combinations are possible. For example a three tap pitch filter may be used to
replace the one tap and increase the update time from 4 msec to 8 msec. The size of the residual vector may also be modified. In this case it was chosen to be 32 samples long, because size 32 is large enough to yield an overall bit rate of 8 Kb/s and is an integer power of 2. The use of multiple gain however, has not been successful in this case. The reason for this was that the correlation factor used in equation (8.18) and given by equation (8.19) did not always carry the same sign, when calculated using the signal at the output of the synthesis filters. This meant that the correlation factors between the vectors, whilst searching for the optimum sequence, did not always reflect the correlation between the reference signal vector and the signal produced by the selected vector at the output of the synthesis filters. Results obtained with respect to various code-book sizes using the parameter specifications given in Table 8.15 are tabulated in Table 8.16. The coder was also simulated with the same parameters, but replacing the one tap pitch filter with a 3 tap pitch filter. These results are given in Table 8.17.

<table>
<thead>
<tr>
<th>Bits</th>
<th>Bit-Rate</th>
<th>Male (dB)</th>
<th>Female (dB)</th>
<th>Overall (dB)</th>
</tr>
</thead>
<tbody>
<tr>
<td>8</td>
<td>7500</td>
<td>9.38</td>
<td>9.86</td>
<td>9.56</td>
</tr>
<tr>
<td>9</td>
<td>7750</td>
<td>9.78</td>
<td>10.67</td>
<td>10.15</td>
</tr>
<tr>
<td>10</td>
<td>8000</td>
<td>10.25</td>
<td>10.73</td>
<td>10.45</td>
</tr>
</tbody>
</table>

Table 8.16: SegSNR performance of VQTC with single tap pitch filter.

Informal subjective listening tests showed that using an 8 bit code-book, good quality speech can be obtained. However, occasional quantization noise was heard when tested using highly sensitive ear-phones. The quality of speech for a 9 bit code-book was very close to the original quality. Finally, the quality at 8 Kb/s, where a 10 bit code-book was employed was as good as the original speech quality. These comparisons were made using a pair of very sensitive high quality ear-phones. In a real telephone environment the difference between the 8 and the 10 bit code-books would not be noticeable. This was confirmed when the comparison was made using a pair of less sensitive speakers. Although, these speakers were still much more sensitive than a typical telephone hand set there was no difference detected in the subjective quality of the 8, 9 or 10 bit code-
books, all of which produced high quality.

<table>
<thead>
<tr>
<th>Bits</th>
<th>Bit-Rate</th>
<th>Male (dB)</th>
<th>Female (dB)</th>
<th>Overall (dB)</th>
</tr>
</thead>
<tbody>
<tr>
<td>8</td>
<td>9500</td>
<td>10.33</td>
<td>10.58</td>
<td>10.45</td>
</tr>
<tr>
<td>9</td>
<td>9750</td>
<td>11.06</td>
<td>11.26</td>
<td>11.13</td>
</tr>
<tr>
<td>10</td>
<td>10000</td>
<td>11.32</td>
<td>12.01</td>
<td>11.58</td>
</tr>
</tbody>
</table>

Table 8.17: SegSNR performance of VQTC with three tap pitch filter.

It can be seen from Table 8.17 that when a three tap pitch filter was used the performance of VQTC increases by about 1 dB at the expense of 2 Kb/s extra information rate. As expected from the segmental SNR performances of all three code-books (8, 9 and 10 bits) the speech quality produced from 9.5 to 10 Kb/s was comparable to the original even when compared using very sensitive ear-phones. However, as we were interested in the 8 to 4.8 Kb/s overall bit rate, the rate at which the pitch parameters were updated was reduced in order to bring down the overall bit rate to below 8 Kb/s. Results given in Table 8.18 are for the same coder as in Table 8.17, but with the pitch parameters update rate reduced by a factor of 2.

<table>
<thead>
<tr>
<th>Bits</th>
<th>Bit-Rate</th>
<th>Male (dB)</th>
<th>Female (dB)</th>
<th>Overall (dB)</th>
</tr>
</thead>
<tbody>
<tr>
<td>8</td>
<td>7125</td>
<td>10.12</td>
<td>10.35</td>
<td>10.32</td>
</tr>
<tr>
<td>9</td>
<td>7375</td>
<td>10.98</td>
<td>11.05</td>
<td>11.01</td>
</tr>
<tr>
<td>10</td>
<td>7625</td>
<td>10.20</td>
<td>11.98</td>
<td>11.46</td>
</tr>
</tbody>
</table>

Table 8.18: SegSNR performance of VQTC with three tap pitch filter updated every 64 samples.
The results in Table 8.18 are not significantly different from those given in Table 8.17. The reason for this is that, although, the pitch filter parameters are updated every 64 samples rather than every 32, it is not expected to have more than one pitch period in every 64 samples. This enables the pitch filter to maintain its efficient prediction. In section 7.4.2 it was shown that when pitch filter parameters were updated every 64 samples, the remaining signal was very close to a Gaussian random signal, which shows the effectiveness of the pitch filter. Subjective quality of all three bit rates from 7125 bits/sec to 7625 bits/sec where 8, 9 and 10 bit code-books were employed was comparable to the original speech quality. Results in Table 8.16 and 8.18 also show that better performance can be achieved if a three tap pitch filter is employed for a given bit rate. In all of the simulations three tap pitch filter parameters were coded with 19 bits as discussed in section 8.2.1.1.

8.3.3 4.8 Kb/s Vector Quantized Transform Coder

The basic principles of the VQTC is the same as that of CELP. Here we proposed VQTC because of its much easier implementation. Therefore, the quality performance of VQTC is expected to be equal to CELP performance. At 4.8 Kb/s the quality of CELP is not comparable with the original speech quality.

This means that the expected quality of VQTC also will not be comparable to the original quality at 4.8 Kb/s. Unless, of course, a very efficient method of quantizing the LPC and pitch parameters is found and hence the rate at which the residual vectors and the pitch parameters are updated is increased. As the main aim of the VQTC is to reduce the complexity of CELP whilst maintaining its quality we found it useful to test the VQTC at 4.8 Kb/s in order to compare it with CELP at the same bit rate.

8.3.3.1 Simulations

A 4.8 Kb/s VQTC was simulated with the specific parameters given in Table 8.19. Parameters given in Table 8.19, again may be modified according to a specific application. The vector size chosen here is 64 samples long which is an integer power of two. Results obtained with respect to various size code-books using the parameters given in Table 8.19 are tabulated in Table 8.20. The same coder was also simulated with a three tap pitch filter. These results are tabulated in Table 8.21.
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Number</th>
<th>Update (msec)</th>
<th>Bits</th>
<th>Bit-Rate</th>
</tr>
</thead>
<tbody>
<tr>
<td>LPC</td>
<td>10</td>
<td>32</td>
<td>40</td>
<td>1250</td>
</tr>
<tr>
<td>Pitch</td>
<td>1-Tap</td>
<td>8</td>
<td>11</td>
<td>1375</td>
</tr>
<tr>
<td>Vector</td>
<td>1</td>
<td>8</td>
<td>10</td>
<td>1250</td>
</tr>
<tr>
<td>Scale</td>
<td>1</td>
<td>8</td>
<td>5</td>
<td>625</td>
</tr>
</tbody>
</table>

Table 8.19: Bit allocation of VQTC at 4.8 Kb/s.

<table>
<thead>
<tr>
<th>Bits</th>
<th>Bit-Rate</th>
<th>Male (dB)</th>
<th>Female (dB)</th>
<th>Overall (dB)</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>4500</td>
<td>5.84</td>
<td>6.5</td>
<td>6.14</td>
</tr>
<tr>
<td>11</td>
<td>4625</td>
<td>6.06</td>
<td>7.02</td>
<td>6.47</td>
</tr>
<tr>
<td>12</td>
<td>4750</td>
<td>6.91</td>
<td>7.44</td>
<td>7.15</td>
</tr>
</tbody>
</table>

Table 8.20: SegSNR performance of VQTC with a single tap pitch filter.

<table>
<thead>
<tr>
<th>Bits</th>
<th>Bit-Rate</th>
<th>Male (dB)</th>
<th>Female (dB)</th>
<th>Overall (dB)</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>5500</td>
<td>7.01</td>
<td>7.40</td>
<td>7.15</td>
</tr>
<tr>
<td>11</td>
<td>5625</td>
<td>7.16</td>
<td>7.68</td>
<td>7.38</td>
</tr>
<tr>
<td>12</td>
<td>5750</td>
<td>7.65</td>
<td>8.04</td>
<td>7.81</td>
</tr>
</tbody>
</table>

Table 8.21: SegSNR performance of VQTC with a three tap pitch filter.

The subjective quality of VQTC using 10, 11 and 12 bit code-books as listed in Table 8.20, contained noticeable quantization noise. None of these bit rates were
acceptable. However, the coder with 4750 bits/sec transmission rate which used 12 bits was very close to being acceptable. Overall, the performance of VQTC was comparable to CELP with similar bit rates.

Using a three tap pitch filter again improves the coder performance by about 1 dB at the expense of 1 Kb/s extra information rate. The performance of the coder at 5.5 Kb/s was better than any of the 4.5 to 4.75 Kb/s single tap pitch filter VQTC's. This was of course expected because of the relative overall transmission rates of the coders. When a 12 bit code-book was used (with a three tap pitch filter) the coder had an acceptable quality at 5.5 Kb/s. This was about the lowest bit rate that could be achieved by CELP with acceptable quality. Reducing the pitch filter parameters update rate by a factor of 2 however, in this case showed significant degradation in quality. Reduction in either the pitch filter parameters rate or the residual vector rate was necessary to bring the total bit rate down to 4.8 Kb/s. Reducing the vector rate (increasing the vector size) introduces more degradation than the degradation caused by reducing the pitch filter parameters update rate. This was also observed in the case of CELP. Therefore it was preferred to reduce the pitch filter parameters update rate. When the pitch filter parameters was reduced from every 32 samples to every 64 in 8 Kb/s VQTC, reduction in pitch filter performance and hence in overall quality was not significant. However, reducing the pitch filter parameters update rate to every 128 samples, clearly causes more degradation. This is because the data size over which the pitch filter parameters are optimized is large and more importantly a data size of 128 samples long is likely to have more than one pitch period which reduces the effectiveness of the filter.

The pitch filter coefficients (three tap) can be vector quantized as was done in CELP to reduce the pitch information and hence increase the update rate. The coder was tested using the same code-book as was used in CELP. Results showed similar improvements to CELP. Although, the quality of VQTC at 4.8 Kb/s may be acceptable for some applications it is necessary to improve its quality further by adaptive post filtering as applied in similar coders [15][18], which will make it attractive for a wider range of applications.

8.3.4 Comparison Of VQTC With CELP

Here, we discuss the advantages and disadvantages of VQTC and CELP, in terms of complexity and quality. In this discussion we do not include delay, because delay is flexible in both VQTC and CELP, and is expected to be comparable.
The complexity of both CELP and VQTC can be divided into two areas. One is the computations required to search the code-book or vector quantization of the residual signal, and the other is the computations required to obtain the residual and to synthesize the speech after vector quantization of residual.

Obtaining the residual and synthesizing the speech in both CELP and VQTC requires the same amount of computation. VQTC requires one subtraction per sample more computation for pitch inverse filtering. No extra multiplication is required for pitch inverse filtering because the predicted value at both inverse and synthesis filter is the same, as discussed in section 8.2.2.2. Before searching the code-book, in VQTC, residual vectors are frequency transformed using DCT. The size of DCT is equal to the residual vector size; i.e., 32 and 64 for 8 and 4.8 Kb/s VQTC respectively. The computations required for 64 point DCT is 4 times that required for the 32 point DCT. This means that the DCT complexity increases when the vector size is increased, i.e., when the bit rate is reduced. For 8 Kb/s the frequency transformation requires 32 multiply-adds per sample and for 4.8 Kb/s this increases to 64 multiply-adds for each sample. In general this is $N$ multiply-adds per sample, $N$ being the DCT size. In VQTC, it is necessary to compute the weighting vectors. Assuming a 128 point FFT is used on the impulse response of the LPC inverse filter, a complex matrix of size $11 \times 128$ needs to be computed and then neighbouring samples are averaged to obtain a number of points on the spectrum equal to the size of the residual vectors, (assuming 10 LPC parameters). After vector quantizing the frequency transformed residual vectors, synthesis is performed in the time domain which requires the inverse DCT transformation of the code-book sequences. However, this can be done off-line, and store the time domain equivalent of the frequency domain code-book. This does not require any real time computation, but it requires extra memory to store the time domain code-book. Assuming that the noise shaping in CELP is performed whilst obtaining the reference signal and by modifying the LPC synthesis filter while searching the code-book [12], the remaining computations of CELP and VQTC (not including code-book search) are exactly the same.

So far we have discussed the difference between CELP and VQTC before and after code-book search. In CELP code-book search requires more than 95% of the overall computations. Therefore, the extra computations VQTC requires when compared with CELP, before and after code-book search, are not significant. In standard CELP code-book search is very complex. The reason for this is the computations required to compute the
convolutions of the synthesis filter response and the code-book sequences. Atal's design [10] has an overall computation estimate of about 500 MIPS. At least 95% of this complexity measure is used for code-book searching.

In VQTC, however, code-book search is simplified by an enormous amount and all of the convolutions required by the synthesis filters are eliminated. For each sequence in the code-book, equations (8.19) and (8.18) are computed and the sequence which minimizes equation (8.18) is selected. After selection of the optimum sequence, the optimum scale factor is calculated in exactly the same way as in CELP. Therefore, overall complexity difference between CELP and VQTC for a code-book search is that VQTC does not require synthesis filter convolutions. Searching the code-book in VQTC can be reduced to computing only one equation as follows,

\[ E_{\text{min}} = \max \left[ \sum_{i=1}^{N} x(i)V(i)W(i)^2 \right] \]

(8.21)

which can be interpreted as searching for maximum weighted correlation.

Equation (8.21) requires two multiply-adds per sample per sequence in the code-book. Assuming a 10 bit code-book the computations required for code-book search is about 16 MIPS. Further simplifications to code-book searching can be made if only a certain number of vector components are included in the search. For example, for a 64 sample vector only 32 of the most important elements may be considered (formant regions elements) which halves the complexity. The overall complexity of VQTC is about 28 to 30 times less than the complexity of standard CELP.

8.3.4.2 Quality

The quality of CELP and VQTC as discussed earlier was very similar at 8 and 4.8 Kb/s. However, vector dimensions of CELP and VQTC were not the same. Therefore, this may not be the true comparison of the two code-book search techniques.
Table 8.22: SegSNR performance of VQTC with 10 bit code-book and 10 LPC parameters.

<table>
<thead>
<tr>
<th>Pitch-Tap</th>
<th>Vector-Size</th>
<th>Male (dB)</th>
<th>Female (dB)</th>
<th>Overall (dB)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>32</td>
<td>10.25</td>
<td>10.73</td>
<td>10.45</td>
</tr>
<tr>
<td>1</td>
<td>64</td>
<td>5.84</td>
<td>6.51</td>
<td>6.14</td>
</tr>
<tr>
<td>3</td>
<td>32</td>
<td>11.32</td>
<td>12.01</td>
<td>11.58</td>
</tr>
<tr>
<td>3</td>
<td>64</td>
<td>7.01</td>
<td>7.37</td>
<td>7.15</td>
</tr>
</tbody>
</table>

In order to have a better comparison of the two coders the size of the vectors, update rates of the parameters and the number of bits in the code-book were kept at the same values. Signal to noise ratios of VQTC and CELP with one and three tap pitch filters are tabulated in Table 8.22 and 8.23 respectively.

Table 8.23: SegSNR performance of CELP with 10 bit code-book and 10 LPC parameters.

<table>
<thead>
<tr>
<th>Pitch-Tap</th>
<th>Vector-Size</th>
<th>Male (dB)</th>
<th>Female (dB)</th>
<th>Overall (dB)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>32</td>
<td>10.94</td>
<td>12.29</td>
<td>11.46</td>
</tr>
<tr>
<td>1</td>
<td>64</td>
<td>6.21</td>
<td>7.19</td>
<td>6.60</td>
</tr>
<tr>
<td>3</td>
<td>32</td>
<td>12.76</td>
<td>13.48</td>
<td>13.01</td>
</tr>
<tr>
<td>3</td>
<td>64</td>
<td>7.45</td>
<td>8.38</td>
<td>7.82</td>
</tr>
</tbody>
</table>

Although, there are some differences in the objective performance of CELP and VQTC, both have very similar subjective quality. Using one and three tap pitch filters and 32 element residual vectors both produced comparable quality to the original speech. However, when the vector size was increased to 64, both contained quantization noise and had similar roughness in the processed speech. In order to further evaluate the
quality of CELP and VQTC. The correlation produced by each selected vector and the minimized error for both coders were investigated. As shown in Figure 8.6 and 8.7 both CELP and VQTC achieve similar correlation and hence error patterns. As the vector size gets smaller the correlation achieved by CELP increases more than that of VQTC. This is also reflected by the SNR performances given in Table 8.22 and 8.23. However, when the overall quality of CELP and VQTC was compared, they were found to be very similar. This makes the VQTC a better coder because of its much simpler implementation.

8.3.5 Discussions

We have explained the principles of a new vector quantized transform coder, and compared it with CELP. VQTC has high quality, which is comparable to the original speech at around 8 Kb/s. Its quality gradually deteriorates as the bit rate reduces, and becomes unacceptable at around 5.5 Kb/s. Below 5.5 Kb/s the speech produced contains large quantization noise which causes roughness. Apart from this roughness no other noise such as clicks could be heard. The quality of VQTC followed the same deterioration pattern as CELP which also becomes unacceptable below about 5.5 Kb/s. This proved the efficiency of the new code-book search method. Because of the new simplified search of the code-book, the complexity of VQTC was about 28 to 30 times simpler, which made it a strong competitor to CELP. One set back with VQTC is that it requires two code-books at the encoder. Although, current DSP chips seem to be providing more and more memory there may be problems in some applications requiring more storage for other tasks in the channel.

Here we have shown that the complexity of CELP can be reduced to a level which can be implemented by current DSP’s without reducing the quality performance. However, both CELP and VQTC are still not capable of producing high quality speech at 4.8 Kb/s. VQTC only reduces the complexity of CELP and makes no attempt to improve on the quality of speech at 4.8 Kb/s. The major quality degrading factors in VQTC are exactly the same ones as in CELP. The most important of all is the large residual vector sizes which causes excess amounts of quantization noise and hence roughness. The second most important quality reducing factor is the reduced pitch prediction and hence the performance of Gaussian code-books (see section 7.4.2). In the following section we will discuss a new CELP base-band (CELP-BB) coder which further simplifies CELP and VQTC, and improves the speech quality below 6 Kb/s.
Figure 8.6: Typical vector (a) correlation and (b) error patterns of CELP (solid) and VQTC (dotted) at 8 Kb/s.
Figure 8.7: Typical vector (a) correlation and (b) error patterns of CELP (solid) and VQTC (dotted) at 4.8 Kbps.
8.4 CELP Base-Band (CELP-BB) Coding Of Speech

In section 8.2 we have discussed one of the most promising low rate speech coders which is the code excited linear prediction (CELP). CELP seemed to be producing high quality speech at bit rates as low as 6 Kb/s. Below 6 Kb/s however, although producing intelligible speech, the amount of quantization noise and roughness makes it unacceptable to be used in any telephone network. In order to offer the possibility of carrying digital speech over a single analogue voice channel it is necessary to bring the high quality speech coding rate down to about 4.8 Kb/s. The other disadvantage of CELP is its high complexity. Following the discussions on CELP, we proposed a new vector quantized transform coder (VQTC) which has similar performance to CELP and yet has a possible single chip implementable complexity. Other simplifications have been reported in the literature which make it possible to have single chip implementation \[12\][13]. From what we have achieved using VQTC, and from reported simplification procedures, it seems that the complexity problem of CELP can be solved and it is possible to have a single chip compact implementation. However, the quality improvement at around 4.8 Kb/s remains to be solved and this is the most important step now to be taken. Here we propose a new CELP base-band (CELP-BB) coding scheme for speech in order to improve the speech quality at around 4.8 Kb/s. As the name suggests, CELP coding is applied to the base-band residual to reduce the bit rate of the base-band coder down to 4.8 Kb/s and below. The algorithm is very similar to replacing APC with RELP at bit rates below 16 Kb/s. APC transmits the full-band residual signal whereas RELP transmits only a base-band and hence requires less transmission capacity. A similar procedure can be followed to reduce the transmission capacity of CELP and hence improve the residual vector and pitch filter parameters update rates which are the two major causes of quality degradation at 4.8 Kb/s.

8.4.1 Base-Band Coding Of Speech

We have briefly described the basic principles of base-band coding in section 5.3. The LPC residual is first low-pass filtered and decimated by a factor given by the ratio of speech bandwidth over the base-band width, which has to be an integer for easy time domain implementation. A decimated base-band residual signal is then coded and transmitted. At the decoder, the received base-band signal is up-sampled by inserting zeros between the samples and then the up-sampled signal is filtered through the LPC synthesis filter. The LPC synthesis filter interpolates the zero valued samples to produce
good quality output. There are two major causes for speech quality degradation in base-band coding. These are the quantization of the base-band signal and the high frequency regeneration (HFR) noise. HFR noise depends on the ratio of decimation. Higher decimation ratios cause higher HFR noise. Both the quantization noise and the HFR noise depend on the overall bit rate of the coder. In practice a compromise is made between the base-band quantization noise and HFR noise by choosing a suitable base-band width. We have discussed in section 7.4.1.3 how pitch filtering can be used to improve both quantization and HFR noise. Although, the design discussed in section 7.5.1.3 produces good quality speech, HFR noise cannot be completely eliminated. For female and child speech, for example, where the pitch is at higher frequencies, folding the base-band spectrum to produce the higher frequencies, breaks the pitch harmonics and causes tonal and aliasing distortions. More information about RELP with pitch prediction can be found in [19].

Multi-pulse excited linear predictive coding (MPLPC) has some similarities with base-band coding. In MPLPC a number of pulses (number depends on the bit-rate) are optimized both in terms of locations and amplitude to minimize the overall error between the original and synthetic speech. During synthesizing, zero valued samples (where there is no pulse) are interpolated by the LPC synthesis filter. Reported speech quality of MPLPC with only 25% of the excitation samples optimized (75% zero valued samples) is better than an ordinary base-band coder. MPLPC speech does not contain aliasing and tonal distortion, however, it is not transparent at only 25% pulse rate. The reason that MPLPC does not have tonal or aliasing distortion is due to its difference from the base-band coder (RELP). In the base-band coder, pulse amplitudes and positions are assumed to be optimal. However, in MPLPC both pulse amplitudes and positions are optimized.

The regular pulse excited (RPE) approach to MPLPC combines the ideas of base-band coding and MPLPC coding [20]. This coder, combined with a pitch predictor, has been chosen for the GSM 16 Kbps speech coding algorithm. In an RPE coder the low-pass filter is replaced by a filter which is called a weighting filter or smoother. Decimated sequences are then compared in terms of their energy and the sequence which has the maximum energy is selected for transmission. The position of the selected sequence is also transmitted. The combination of the weighting filter and selection of maximum energy sequence is equivalent to optimizing the MPLPC pulse amplitudes. In the usual base-band coder the first sequence is chosen. RPE eliminates the tonal distortion and the aliasing effect seen in female speech, and produces transparent speech at bit rates as low
as 12 Kb/s. Below 12 Kb/s the coder allocates fewer bits to code the pulses which cause roughness in the recovered speech.

CELP-BB coding of speech is based on RPE and vector quantization, with a pitch filter operating on the decimated base-band signal.

8.4.2 CELP-BB Coder Description

A block diagram of CELP-BB [22] is shown in Figure 8.8. The input speech is inverse filtered to obtain the LPC residual which is then divided into sub-blocks. Each sub-block is filtered by the weighting filter separately. Filtered sub-blocks are split into a number of sequences equal to the decimation factor. These sequences are compared in terms of their energies, one with the highest energy is selected for transmission. The position of the selected sequence in each sub-block is transmitted to the decoder to place the pulses in the correct locations. Selected sequences are then stored in a buffer, side by side, to form a decimated continuous signal. In RPE, sequences are quantized using scalar quantizers, and transmitted separately. Here vector quantization is applied to code the continuous decimated signal. The principles of the vector quantization is based on CELP which works as follows: The decimated signal is analysed to obtain its pitch period and pitch filter coefficients. The pitch synthesis filter is then clocked with zero value input to determine the memory response which is subtracted from the decimated signal, so as to form the reference signal. Gaussian code-book sequences are then searched one by one to match the reference signal. The index of the optimum sequence together with the scale factor is transmitted to the decoder. At the decoder, code-book sequences are scaled up by the optimum scale and passed through the pitch synthesis filter to obtain the continuous decimated signal. The recovered signal is then sub-segmented and shifted to the correct positions with zeros inserted in between the pulses to form the excitation sequence. The LPC synthesis filter is then excited to recover the output speech. A description of the coder up to the selection of the maximum energy sequence after weighting filtering is given in [20]. Therefore, we will only concentrate on the vector quantization of the selected sequences.

8.4.3 Vector Quantization Of The Decimated Signal

In order to have an integer number of pulses in each sequence the LPC residual is divided into a number of sub-blocks each containing a number of samples which are an integer multiple of the decimation factor. Weighted sub-blocks are split into a number of
CELP-BB Encoder

Input
\[ \text{LPC Inverse Filter} \rightarrow \text{Low-Pass Filter And Decimate} \rightarrow \text{CELP Encoder} \]

CELP-BB Decoder

\[ \text{CELP Decoder} \rightarrow \text{Upsample} \rightarrow \text{LPC Synth. Filter} \rightarrow \text{Output} \]

Figure 8.8: A block diagram of CELP-BB.

Figure 8.9: A block diagram of CELP used in CELP-BB.
sequences equal to the decimation factor \( d \) as,

\[
S_j(i) = W(i \cdot d + j)
\]

(8.22)

\[i = 0, 1, \ldots, \frac{N}{d} - 1 \quad \text{and} \quad j = 1, 2, \ldots, d\]

where \( S_j(i) \) is the \( j^{th} \) sequence, \( W(i) \) is the weighted sub-block and \( N \) is the number of samples in each sub-block. The energies of each sequence are calculated as,

\[
\sigma_j^2 = \sum_{i=1}^{N} S_j^2(i)
\]

(8.23)

The sequence with maximum \( \sigma_j^2 \) is selected for coding and transmission. In a frame, all of the selected sequences are placed sequentially to form a continuous signal \( y(n) \) which contains \( d \) times less samples than the original LPC residual signal \( x(n) \). This means that the upper and lower limits of the expected pitch period are reduced by a factor of \( d \). The continuous decimated signal is then used as the input to an analysis by synthesis vector quantizer or CELP coder as shown in Figure 8.9. The input to the CELP coder in this case does not contain short-time correlation, it however, has a much stronger long-time correlation. Therefore, in the CELP coder both the LPC synthesis filter and the noise shaping filter are excluded leaving only the pitch synthesis filter. An analysis by synthesis procedure operating around the pitch synthesis filter vector quantizes the decimated signal \( y(n) \). The dimension of the code-book sequences are set to be equal to the number of samples in each decimated sequence. This is not of course a restriction. The error is minimized using equation (8.9) and an optimum scale is calculated using equation (8.8) where \( x_n \) and \( f_n \) are replaced by vectors formed from \( y(n) \) and the impulse response of the pitch synthesis filter convolved with unit variance code-book sequences respectively. The pitch filter implementation discussed in section 8.2.2.2 can also be applied in this case and this reduces the complexity of the search. The output response of the pitch synthesis filter excited by the unit variance code-book sequences can be written as,

\[
f_n = V(i) \ast P(i)
\]

(8.24)
where \( V(i) \) is the code-book sequence and \( P(i) \) is the pitch synthesis impulse response. When the delay in the pitch synthesis filter is at least as large as the vector size then the truncated impulse response \( P(i) \) has a value of 1 at the first location and zeros everywhere else. This makes \( V(i) = V(i) \ast P(i) \) and hence \( f_n = V(i) \). Therefore, equations (8.9) and (8.8) can be written as equations (8.14) and (8.13) respectively where a similar procedure was applicable to LPC residual matching in CELP as discussed in section 8.2.3.1.

The synthesized decimated signal \( \hat{y}(n) \) is split into sequences which are put together to form \( y(n) \). With the help of the corresponding position index \( j \) associated with each sequence, received sequences are shifted to the correct positions with the necessary zeros inserted in between the samples to form the final excitation signal at the decoder.

The LPC and pitch filter parameters are coded in the same way as discussed in relation to CELP. The overall bit rate of CELP-BB is simply determined by the vector size of the decimated signal and the pitch filter parameter update rate. By varying the vector dimensions it is possible to achieve a range of bit rates from 10 Kb/s down to 2.4 Kb/s. In order to compare CELP-BB with CELP and VQTC it was tested both at 8 and 4.8 Kb/s.

### 8.4.4 8 Kb/s CELP-BB

Although, CELP-BB is a base-band coder, most of the transmission capacity is occupied by the CELP coder which operates in the base-band. It is therefore necessary to adjust the parameters of CELP as was done in the previous sections to achieve a given overall bit rate. The flexibility of CELP-BB lies in its decimation factor which enables smaller residual vector dimensions and more frequent pitch filter parameter updates. Pulse position coding and LPC parameter coding are fixed for various bit rates from 8 to 4.8 Kb/s.

An 8 Kb/s CELP-BB was simulated in order to achieve high quality speech by finding the optimum update rate for the residual vectors and pitch parameters.

#### 8.4.4.1 Simulations

An 8 Kb/s CELP-BB was simulated with the parameters given in Table 8.24.
Table 8.24: Bit allocation of CELP-BB at 7 Kb/s.

A decimation factor of 3 was used. 4.875 msec corresponds to 39 samples when sampled at 8 KHz, which is a sub-block of the 195 sample long frame. There are 5 sub-blocks in each frame and 3 sequences in each sub-block. Each sequence has 13 samples. This means that the vector size is 13 samples (decimated) long and the pitch parameters are updated every sequence, i.e., every 13 samples of the decimated signal.

Table 8.25: Performance of CELP-BB with single tap pitch filter.

CELP-BB at about 7 Kb/s was tested by calculating its objective SNR's and conducting informal listening tests. As CELP-BB is a base-band coder, SNR's relating to both base-band quantization performance and the overall coder performance was calculated. We have also tested the coder with a three tap pitch filter. In Table 8.25 and 8.26 SNR performances of CELP-BB with one and three tap pitch filters are tabulated respectively. In the tables base-band prediction refers to the segmental SNR of original base-band and

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Number</th>
<th>Update (msec)</th>
<th>Bits</th>
<th>Bit-Rate</th>
</tr>
</thead>
<tbody>
<tr>
<td>LPC</td>
<td>10</td>
<td>24.375</td>
<td>40</td>
<td>1641</td>
</tr>
<tr>
<td>Pitch</td>
<td>1-Tap</td>
<td>4.875</td>
<td>10</td>
<td>2051.3</td>
</tr>
<tr>
<td>Vector</td>
<td>1</td>
<td>4.875</td>
<td>8</td>
<td>1641</td>
</tr>
<tr>
<td>Scale</td>
<td>1</td>
<td>4.875</td>
<td>6</td>
<td>1231</td>
</tr>
<tr>
<td>Position</td>
<td>4</td>
<td>4.875</td>
<td>2</td>
<td>410</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>SNR</th>
<th>SegSNR (dB)</th>
<th>Usual-SNR (dB)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Base-Band Quantization</td>
<td>13.06</td>
<td>12.20</td>
</tr>
<tr>
<td>Base-Band Prediction</td>
<td>8.40</td>
<td>7.66</td>
</tr>
<tr>
<td>Overall coder</td>
<td>8.47</td>
<td>7.88</td>
</tr>
</tbody>
</table>
the pitch filter memory response.

<table>
<thead>
<tr>
<th>SNR</th>
<th>SegSNR (dB)</th>
<th>Usual-SNR (dB)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Base-Band Quantization</td>
<td>14.24</td>
<td>12.34</td>
</tr>
<tr>
<td>Base-Band Prediction</td>
<td>9.77</td>
<td>8.07</td>
</tr>
<tr>
<td>Overall Coder</td>
<td>9.06</td>
<td>8.32</td>
</tr>
</tbody>
</table>

Table 8.26: Performance of CELP-BB with three tap pitch filter.

Using a three tap pitch filter requires 8 bits per sequence more information and hence the overall bit rate of the coder in Table 8.26 is about 9.6 Kb/s.

The subjective quality of CELP-BB using both one tap and three tap pitch filters was comparable to the original speech quality. In section 7.5 we have shown that 3 bits per sample quantization was the optimum number of bits/sample when scalar quantizers were used. The choice of 4 bits/sample did not improve the quality of the base-band coder. The performance of the 3 bit APCM quantizer was found to be around 13 dB. Here, using an 8 bit code-book and one tap pitch filter we have achieved a similar performance. Introducing the regular pulse approach helps to eliminate high frequency distortion and enables high quality speech. The success of the vector quantization using only 8 bits comes from the very effective pitch synthesis filter implementation. It can be seen from Tables 8.25 and 8.26 that 8.40 dB and 9.77 dB prediction was achieved using one and three tap pitch filters respectively. Only the remaining 4.66 dB and 4.48 dB base-band quantization performance was achieved by the vector quantizers for the one and three tap pitch filter cases. This also shows that the SNR increase of base-band quantization using a three tap pitch filter over the base-band with one tap pitch filter was solely due to better prediction and is about 1.2 dB.

The overall coder SNR performance is in the region of 8.5 dB at 7 Kb/s. Although, 8.5 dB SegSNR is less than that of CELP at 7 Kb/s, it is comparable with VQTC at 7 Kb/s. Also, as CELP-BB is only a base-band coder it would be expected to have lower SNR values than the full-band coders. Base-band coders are best compared using subjective listening tests. Therefore, we have also compared CELP, VQTC and CELP-BB all at 7
Kb/s using processed speech sentences. Informal listening tests amongst the people in the speech lab showed that all three coders had very similar quality which was comparable to the original speech.

8.4.5 4.8 Kb/s CELP-BB

The prime objective of CELP-BB was to enhance speech quality below 6 Kb/s, since we have shown earlier that both CELP and VQTC can produce high quality speech at rates as low as 6 Kb/s. The overall bit rate of CELP-BB can be reduced down below 6 Kb/s simply by choosing larger dimensions for the residual vectors. Although, it will not reduce the bit rate very much, the update rate of the LPC parameters can be reduced by increasing the frame size to about 30 msec. By adjusting the parameters of CELP-BB along the above lines it was simulated at 4.8 Kb/s.

8.4.5.1 Simulations

The overall parameters of CELP-BB at 4.8 Kb/s are tabulated in Table 8.27.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Number</th>
<th>Update (msec)</th>
<th>Bits</th>
<th>Bit-Rate</th>
</tr>
</thead>
<tbody>
<tr>
<td>LPC</td>
<td>10</td>
<td>30</td>
<td>40</td>
<td>1333.33</td>
</tr>
<tr>
<td>Pitch</td>
<td>1-Tap</td>
<td>7.5</td>
<td>10</td>
<td>1333.33</td>
</tr>
<tr>
<td>Vector</td>
<td>1</td>
<td>7.5</td>
<td>9</td>
<td>1200.00</td>
</tr>
<tr>
<td>Scale</td>
<td>1</td>
<td>7.5</td>
<td>5</td>
<td>666.67</td>
</tr>
<tr>
<td>Position</td>
<td>4</td>
<td>7.5</td>
<td>2</td>
<td>266.67</td>
</tr>
</tbody>
</table>

Table 8.27: Bit allocation of CELP-BB at 4.8 Kb/s.

Here, again a decimation factor of 3 was used. The frame size was increased to 30 msec (240 samples) which contained 4 sub-blocks of 60 samples. Each sequence had 20 samples which meant that the vector size was also 20 samples long and the pitch filter parameters were updated every 20 samples (decimated signal).
Objective and subjective performance of CELP-BB at 4.8 Kb/s was evaluated in exactly the same way as was done for the 7 Kb/s CELP-BB. SNR performances using one and three tap pitch filters are tabulated in Tables 8.28 and 8.29 respectively.

<table>
<thead>
<tr>
<th></th>
<th>SegSNR (dB)</th>
<th>Usual-SNR (dB)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Base-Band Quantization</td>
<td>8.16</td>
<td>7.41</td>
</tr>
<tr>
<td>Base-Band Prediction</td>
<td>5.42</td>
<td>4.67</td>
</tr>
<tr>
<td>Overall Coder</td>
<td>6.10</td>
<td>5.78</td>
</tr>
</tbody>
</table>

Table 8.28: Performance of CELP-BB with a single tap pitch filter.

<table>
<thead>
<tr>
<th></th>
<th>SegSNR (dB)</th>
<th>Usual-SNR (dB)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Base-Band Quantization</td>
<td>8.47</td>
<td>7.16</td>
</tr>
<tr>
<td>Base-Band Prediction</td>
<td>5.63</td>
<td>4.38</td>
</tr>
<tr>
<td>Overall Coder</td>
<td>6.35</td>
<td>5.62</td>
</tr>
</tbody>
</table>

Table 8.29: Performance of CELP-BB with a three tap pitch filter.

Here, again using a three tap pitch filter requires 8 bits per sequence more information which is just over 1 Kb/s. It can be seen from Table 8.28 and 8.29 that the prediction achieved by the one and three tap pitch filters is not significantly different. This is because the pitch filter is applied to the smoothed and decimated signal which increases the prediction of the single tap pitch filtering. The performance of the vector quantizer also showed reduction from 4 dB to just under 3 dB. This of course was because of the increased vector dimension. Overall segmental SNR of the complete coder was comparable to CELP and VQTC. This shows that the performance reduction of CELP-BB, as the bit rate is reduced, is much slower than both CELP and VQTC.
The subjective quality of CELP-BB at 4.8 Kb/s was tested by informal listening tests. The results showed that CELP-BB maintains its high quality even at 4.8 Kb/s and of course outperforms both CELP and VQTC at bit rates below 6 Kb/s. CELP-BB at 4.8 Kb/s was also compared with CELP-BB at 7 Kb/s. It was noticed that there were slight differences between the two bit rates. However, these differences did not affect the quality of the 4.8 Kb/s algorithm because the differences could not be described as quantization or any other form of noise. One conclusion made was that the 7 Kb/s algorithm was more refined than the 4.8 Kb/s algorithm and although it had high quality it was not as clean and refined as the 7 Kb/s algorithm. No roughness or disturbing quantization noise was present in any of the processed speech. This of course was not the case for CELP and VQTC below 6 Kb/s.

CELP-BB was also simulated at 2.4 Kb/s to provide an alternative to the traditional LPC-10 vocoder. However, we will not discuss this here. It will be discussed later, because here the objective was to produce good quality speech at rates around 4.8 Kb/s.

8.4.6 Comparison Of CELP-BB With CELP And VQTC

As with the comparison of VQTC with CELP discussed earlier, comparison of CELP-BB with CELP and VQTC can be discussed in terms of complexity and quality.

8.4.6.1 Complexity

We have discussed some of the simplification methods of CELP and also compared VQTC with CELP. It seems that CELP cannot be implemented without some form of simplifications or structural designed code-books. VQTC on the other hand can be implemented with one or at maximum two DSP 32 AT&T chips. The reason for VQTC being simpler than CELP was that the synthesis filter convolutions were eliminated from the code-book search. However, a few other extra computations were required. These were the computation of weighting vectors, and frequency transformation of the residual vectors. It also required more memory to store both time and frequency domain representations of the random Gaussian code-book sequences. CELP-BB does not need any frequency transformation or extra memory for storage. In fact, because of smaller vector dimensions, and hence using 8 or 9 bit code-books, it requires smaller storage. LPC inverse filtering of all three coders have equal complexity. After LPC inverse filtering, CELP-BB has a weighting filter which has an impulse response of 11 samples long (see Table 8.30), and has a grid selector where the energies of the sequences are compared.
Synthesis of speech at the decoder requires similar amount of computations as required for CELP and VQTC. In VQTC and CELP, the pitch filter operates on the full-band LPC residual which has a range of up to 160 samples. In CELP-BB this is reduced to 55 which means reduced pitch filter complexity.

<table>
<thead>
<tr>
<th>i</th>
<th>W(i)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 &amp; 10</td>
<td>-0.016356</td>
</tr>
<tr>
<td>1 &amp; 9</td>
<td>-0.045649</td>
</tr>
<tr>
<td>2 &amp; 8</td>
<td>0.000000</td>
</tr>
<tr>
<td>3 &amp; 7</td>
<td>0.250793</td>
</tr>
<tr>
<td>4 &amp; 6</td>
<td>0.700790</td>
</tr>
<tr>
<td>5</td>
<td>1.000000</td>
</tr>
</tbody>
</table>

Table 8.30: Impulse response of 11 tap weighting filter.

The code-book search in CELP-BB requires no convolution. First the pitch filter memory is subtracted from the original base-band signal to form the reference signal. Then, each sequence in the code-book is used to compute the correlation with the reference signal and the sequence which maximizes the squared correlation is selected for transmission. The optimum scale factor is equal to the correlation which is automatically calculated and requires no extra computation. In VQTC, although the code-book search was simplified, for optimum scale calculation, it still required the convolution of the selected sequence with the synthesis filter. Therefore, the computation required for code-book search in CELP-BB needs 1 multiply-add per sample per sequence, which is $8000/d \times 2^b$ multiply-adds per second. Because of the decimation factor $d$ the complexity of the code-book search in CELP-BB is further reduced. For 7 Kb/s and 4.8 Kb/s CELP-BB using 8 and 9 bit code-books respectively and a decimation factor of 3 the overall search computations are.

$\frac{8000}{3} \times 2^8 = 0.69$ MIPS and
8000/3 \times 2^9 = 1.4 \text{ MIPS.}

Even if 10 bit code-books are used the code-book search computation will be below 3 MIPS. Hence, the overall complexity of CELP-BB at 7 and 4.8 Kb/s is well within the capabilities of a single DSP 32 AT&T chip. CELP-BB is about 3 times simpler than VQTC which is 28 times simpler than standard CELP using 10 bit Gaussian code-books in all three.

8.4.6.2 Quality

As was mentioned earlier, at around 7 Kb/s all three coders (CELP, VQTC and CELP-BB) have quality comparable to the original speech quality. Reduction in quality of CELP and VQTC as the bit rate reduces is much more rapid than CELP-BB. At around 6 Kb/s the CELP-BB quality starts to be significantly better. At 4.8 Kb/s CELP-BB still maintains its high quality whereas CELP and VQTC both have unacceptable quality below about 5.5 Kb/s.

The improved quality of CELP-BB has been achieved by smaller vector sizes and better pitch prediction. Various waveforms such as memory response, base-band signal and coded base-band etc for CELP-BB at 7 and 4.8 Kb/s are shown in Figure 8.10 and 8.11 respectively. From Figures 8.10 and 8.11, it can be seen that the filter memory response matches the original base-band signal very well leaving only a small portion of the original base-band to be matched by the vector quantizer. This enables high quality even at 4.8 Kb/s.

8.4.7 Discussions

We have proposed a new low bit rate coder which was denoted as CELP-BB. CELP-BB has been proposed for bit rates between 8 to 4.8 Kb/s but it can operate at lower bit rates than 4.8 Kb/s. CELP-BB at rates below 4.8 Kb/s will be discussed in the next section. CELP-BB has two major advantages and has no disadvantages when compared with CELP and VQTC at 8 to 4.8 Kb/s. These are, its lower complexity and its high quality performance below 6 Kb/s. As well as its simplicity its high quality at 4.8 Kb/s is the most important factor which makes CELP-BB the best coder when compared with CELP and VQTC. Its high quality is due to the smaller base-band residual vectors and more frequent pitch filter parameter update rates which enable accurate quantization of the base-band signal. Also, the usual high frequency regeneration distortions seen in
Figure 8.10: Typical speech waveforms in CELP-BB. (a) original (b) 7 Kb/s CELP-BB, (c) original base-band, (d) pitch filter memory response and (e) coded base-band.
Figure 8.11: Typical speech waveforms in CELP-BB, (a) original, (b) 4.8 Kb/s CELP-BB, (c) original base-band, (d) pitch filter memory response and (e) coded base-band.
ordinary base-band coders have been eliminated by the use of a weighting filter combined with grid selection (selection of maximum energy sequence) as suggested in [20].

8.4.8 2.4 Kb/s CELP-BB

Results presented in section 8.4.5.1 showed that there was still the possibility of reducing the bit rate of CELP-BB below 4.8 Kb/s. The most important bit rate of interest below 4.8 Kb/s is at 2.4 Kb/s. At 2.4 Kb/s the traditional LPC-10 vocoder has become standard. LPC-10 cannot produce natural quality speech and its performance is judged by the intelligibility of the received speech. Here, we have simulated a 2.4 Kb/s CELP-BB in order to provide an alternative or a replacement to LPC-10.

8.4.8.1 Simulations

CELP-BB with the parameters given in Table 8.31 was simulated at an overall bit rate of 2.4 Kb/s.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Number</th>
<th>Update (msec)</th>
<th>Bits</th>
<th>Bit-Rate</th>
</tr>
</thead>
<tbody>
<tr>
<td>LPC</td>
<td>10</td>
<td>30</td>
<td>10</td>
<td>333.33</td>
</tr>
<tr>
<td>Pitch</td>
<td>3-Tap</td>
<td>15</td>
<td>13</td>
<td>866.67</td>
</tr>
<tr>
<td>Vector</td>
<td>1</td>
<td>15</td>
<td>10</td>
<td>666.67</td>
</tr>
<tr>
<td>Scale</td>
<td>1</td>
<td>15</td>
<td>5</td>
<td>333.33</td>
</tr>
<tr>
<td>Position</td>
<td>4</td>
<td>15</td>
<td>2</td>
<td>133.33</td>
</tr>
</tbody>
</table>

Table 8.31: Bit allocation of CELP-BB at 2.4 Kb/s.

The frame length was chosen to be the same as that of 4.8 Kb/s of 240 samples. 10 LPC parameters were vector quantized with a 10 bit code-book using the Itakura-Saito distortion measure. A decimation factor of 4 was used to split the two sub-blocks of 120 samples into 4 sequences of 30 samples. Selected sequences were put together to produce the continuous decimated signal for vector quantization. A three tap pitch filter was used in the analysis by synthesis loop to code the decimated vectors. Pitch parameters were
coded with 6 bits for the pitch period and 7 bit code-book for the 3 coefficients and were updated every sequence, i.e. 30 samples of decimated signal. Vectors of 30 samples long were coded with 10 bits and their scale values were coded with 5 bits. The SNR performance of the coder can be seen in Table 8.32.

<table>
<thead>
<tr>
<th>SNR</th>
<th>SegSNR (dB)</th>
<th>Usual-SNR (dB)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Base-Band Quantization</td>
<td>3.26</td>
<td>3.75</td>
</tr>
<tr>
<td>Base-Band Prediction</td>
<td>1.58</td>
<td>1.96</td>
</tr>
<tr>
<td>Overall Coder</td>
<td>2.94</td>
<td>2.89</td>
</tr>
</tbody>
</table>

Table 8.32: SNR performance of CELP-BB at 2.4 Kb/s with a three tap pitch filter.

The coder was also tested with a single tap pitch filter which required 200 bits/sec less overall transmission rate. Results are tabulated in Table 8.33.

<table>
<thead>
<tr>
<th>SNR</th>
<th>SegSNR (dB)</th>
<th>Usual-SNR (dB)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Base-Band Quantization</td>
<td>3.20</td>
<td>3.95</td>
</tr>
<tr>
<td>Base-Band Prediction</td>
<td>1.50</td>
<td>2.10</td>
</tr>
<tr>
<td>Overall Coder</td>
<td>2.74</td>
<td>2.87</td>
</tr>
</tbody>
</table>

Table 8.33: SNR performance of CELP-BB at 2.2 Kb/s with one tap pitch filter.

From Tables 8.32 and 8.33 it can be seen that there is no significant difference in the performance of base-band quantization, base-band prediction and the overall coder performance for the one and three tap pitch filters. The subjective quality of both coders was not very smooth. Although, there was no high frequency regeneration noise, as heard in ordinary base-band coders, speech quality was found to be slightly rougher than
expected. This means that the grid selection (selection of maximum energy sequence) combined with the weighting filter can maintain its performance in eliminating the high frequency regeneration noise but causes slightly higher roughness. The roughness caused was due firstly to poor quantization of the base-band signal. The performance of the quantization was less than 1 bit/sample scalar quantization. The second reason for roughness in the recovered speech is more important than that caused by quantization. As the length of the sequences gets bigger the energies in the sequences tend to be equal or very close, and hence the function of the weighting filter cannot be exploited. The use of 2 bits per sequence to code the position of the selected sequence does not produce the desired advantage as it did in the 7 and 4.8 Kb/s CELP-BB's.

Replacing the weighting filter with a low-pass filter and discarding the grid selection, i.e. implementing an ordinary base-band coder was tested with the parameters given in Table 8.34.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Number</th>
<th>Update (msec)</th>
<th>Bits</th>
<th>Bit-Rate</th>
</tr>
</thead>
<tbody>
<tr>
<td>LPC</td>
<td>10</td>
<td>25</td>
<td>10</td>
<td>400</td>
</tr>
<tr>
<td>Pitch</td>
<td>1-Tab</td>
<td>12.5</td>
<td>10</td>
<td>800</td>
</tr>
<tr>
<td>Vector</td>
<td>1</td>
<td>12.5</td>
<td>10</td>
<td>800</td>
</tr>
<tr>
<td>Scale</td>
<td>1</td>
<td>12.5</td>
<td>5</td>
<td>400</td>
</tr>
</tbody>
</table>

Table 8.34: Bit allocation of CELP-BB at 2.4 Kb/s with no grid selection.

As can be seen from Table 8.33 the three tap pitch filter is replaced by a one tap and the update rate of LPC parameters, pitch filter parameters and residual vector rate are increased. Increased update rate of the pitch parameters improved the prediction by 0.48 dB and also smaller vector sizes helped to improve the base-band quantization performance by 0.84 dB. When a three tap pitch filter was used the improvement obtained in prediction and base-band quantization were 0.99 dB and 1.48 dB respectively.

Replacing the weighting filter by a low-pass filter caused slight high frequency regeneration noise, however, the overall speech quality was smoother. Overall speech
quality was natural, intelligible and smooth. It did not contain any undesirable clicks or annoying energy level variation noise present in LPC-10. The scheme compares very well with LPC-10 and could form the basis for a replacement.

8.5 Discussions

In this chapter we have first explained one of the most promising low bit rate coder to date, CELP, and then proposed two alternatives. CELP produces good quality speech down to about 6 or 5.5 Kb/s. Below 5.5 Kb/s its speech quality suffers from quantization noise and roughness. The major causes of these are the reduced update rate of pitch filter parameters and the increased sizes of the excitation vectors. Vector quantizing the pitch coefficients (3-Tap) improved the speech quality slightly. It is necessary to vector quantize the LPC parameters so as to reduce the side information further if good quality speech is to be produced at around 4.8 Kb/s.

CELP is very complex. In order to reduce its complexity we proposed the vector quantized transform coder (VQTC) which is an improved version of the linear predictive coding of speech with VQ and frequency domain noise shaping as discussed in section 7.4. VQTC produced very similar quality to CELP with about 28 times less complexity. The quality of VQTC however, followed the same pattern as CELP and became unacceptable below about 5.5 Kb/s. Implementation of VQTC requires one or two DSP 32 AT&T chips.

In the final part of this chapter we have discussed another alternative to CELP which was called CELP base-band coder (CELP-BB). This produced comparable speech quality to both CELP and VQTC above 6 Kb/s. At 6 Kb/s and below its quality was much better. Its quality at 4.8 Kb/s was very good. It was quantization noise free and had no roughness or any other unpleasant noise. It produced good quality speech at bit rates as low as 3.5 Kb/s. Its performance at 2.4 Kb/s outperforms the traditional LPC-10 because of its natural and smooth quality. However, we feel that its quality at 2.4 Kb/s can still be improved.

As well as its quality advantage over CELP and VQTC it also has complexity advantage. Using the new pitch filter implementation discussed in section 8.2.2.2, the coder was simplified. It is about 3 or 4 times less complex than VQTC and can easily be implemented on one DSP 32 AT&T chip. If required the coder can be further simplified by designing structured code-books, but this does not seem to be necessary.
8.6 References


22. A. M. Kondoz, B. G. Evans, "CELP Base-Band coder for high quality speech coding at 9.6 to 2.4 KBPS" To be presented at ICASSP-88 in New-York.
9.1 Introduction

Subjective testing is very important when comparing speech coders. Speech coders such as PCM, APCM and ADPCM which operate in the region of 64 Kb/s to 32 Kb/s may be compared in terms of signal to noise ratio of the recovered speech. This is because:

(i) Noise in PCM, APCM, and ADPCM contains only quantization error. Hence, they all have similar types of distortion which enables the overall signal to noise ratio to reflect the subjective quality of the coders within the limits of small tolerance.

(ii) Coders operating in the 64 Kb/s to 32 Kb/s region have fairly high signal to noise ratios (above 25 dB) which makes it difficult to detect any error at all.

For the above two reasons, SNR comparison of coders operating in 64 - 32 Kb/s region may be assumed to be equivalent to their expected subjective quality.

For the coders operating below 32 Kb/s neither of the above two claims are true. Firstly the noise does not contain only quantization error and coders operating at around 16 Kb/s can only produce about 20 dB signal to noise ratio. It is therefore, much more important to conduct subjective tests to compare the coders working at 16 Kb/s and below. Noise in these coders may vary from high frequency tonal noise (RELP), quantization noise (APC), band limitation and aliasing noise (SBC and ATC), back ground noise, granular noise (APC), etc.

Subjective testing methods were developed long before low bit rate speech coders (below 16 Kb/s) were in demand. It is extremely important to satisfy the conditions which are necessary for subjective testing and to conduct the test in a fair way for all the coders under test. Currently, there is not a subjective testing method which has been developed especially for the low bit rate coders. At the present moment major speech coding research companies and institutes are trying to developed a better subjective testing method or analysis for coders below 16 Kb/s. In the following section we briefly explain the existing subjective listening test methods.
9.2 Listening Tests

Listening tests are concerned with assessing the effect of noise and various other distortions that cause difficulty in listening. Several methods have been used for subjective listening testing such as articulation tests, immediate appreciation tests, opinion scale tests, pair comparison tests, pair comparison ranking tests, Youden square rank ordering tests and quantal response threshold tests.

Articulation tests are performed by reading standardized sounds of speech material over the speech systems under test. The percentage of the material recognized correctly by a group of listeners is a measure of articulation that reflects the speech quality of the system under test [1][2]. Various articulation tests differ in the kind of speech material used, such as words or sentences etc.

Immediate appreciation tests have the same structure as articulation tests [1], where a number of sentences, unrelated in meaning are read, and the listener subject is required for each to indicate whether he understood the meaning of the sentence without reasonable effort [3]. This kind of test differs from articulation tests in that it also considers the effort needed. Immediate appreciation tests suffer from poor sensitivity in the range of quality of existing telephone networks.

Opinion scale listening test methods are widely used. It is a description scale method consisting of a limited number of discrete ascending or decending steps, which are usually assigned numerical values to quantify the measured criterion, such as the mean opinion scores (M.O.S) that express the opinion distribution with one parameter. Randomised speech material is used in such tests.

In pair comparison listening tests, two test conditions at a time are presented to the subjects who are asked to choose the one which best satisfies a predefined criterion. The test conditions could be impaired speech or noise. Pair comparison tests can also be used for ranking a number of test conditions. In this case the possible pairs of the test conditions involved are presented to the subjects in order to indicate which of the pair of test conditions he prefers [4]. The order of presentation of the pairs should be randomized in order to avoid any undesirable bias.

Youden square rank order testing methods have been introduced for telecommunication applications, as a means of evaluating equivalent noise. This method is similar in basic principles to the pair comparison ranking method. The test conditions are presented in groups of three or more instead of pairs, as in the pair comparison ranking method.
Finally, in the quantal response threshold tests [2], the subject listens to one test condition at a time and in each case is required to give a quantal response. The test conditions in this case is any parameter whose threshold is to be found.

In the following, subjective tests results of various 16 and 9.6 Kb/s coders will be discussed. In our testing the widely used mean opinion score subjective testing method was used.

9.3 Subjective Testing And Results

The details of the test are given in [5]. Thirty subjects from the University, secretaries, lecturers and students were individually asked to give a score for each of the test sentences in a sound proof room. As the test material, two male and two female sentences were used.

Male
"They kept her running about"
"Jar was full of water".
and,

Female
"Are you going to be nice to me"
"You know my out going life".

After listening to each of the sentences which were presented in random order to each subject, they were asked to mark a point on a five point scale for each sentence. Each sentence was presented only once for each subject. They were asked to assume that the sentences were coming from a telephone system and to judge by considering sentence intelligibility, understandability, and the distortions that were bothering them. In order to make it easy for the subjects, the 5 point scale was also marked as BAD, POOR, FAIR, GOOD and EXCELLENT in ascending order.

It is important to keep the subjects in the sound proof room for as short a time as possible. After a long time of listening fatigue sets in and the results may become biased. The duration of tests depends on the test conditions, length of the test material and the number of coders. In order to keep the test time down only 6 coders with one condition (no errors) were tested. Three of the coders were at 16 Kb/s and the other three were at 9.6 Kb/s. The 16 Kb/s coders were ATC (Zelisky and Noll's), SBC and PDTC as
discussed in sections 6.3.1.1, 6.2.4 and 6.3.1.3 respectively, and the 9.6 Kb/s coders were ATC with vector quantized side information, SBC with vector quantized side information, and HTC as discussed in sections 7.3.1, 7.2.1.2 and 7.3.3 respectively.

Scores for each coder varied within certain levels. Variations were with respect to the coder, bit rate and test material, i.e., male or female. Results of variations for 16 and 9.6 Kb/s coders with respect to the test material are tabulated in Table 9.1 and 9.2 respectively.

<table>
<thead>
<tr>
<th>Data</th>
<th>ATC</th>
<th>SBC</th>
<th>PDTC</th>
</tr>
</thead>
<tbody>
<tr>
<td>Male</td>
<td>3-5</td>
<td>4-5</td>
<td>4-5</td>
</tr>
<tr>
<td>Female</td>
<td>3-5</td>
<td>3-5</td>
<td>3-5</td>
</tr>
</tbody>
</table>

Table 9.1: Maximum and minimum scores for the 16 Kb/s coders with respect to the test material.

<table>
<thead>
<tr>
<th>Data</th>
<th>ATC-VQ</th>
<th>SBC-VQ</th>
<th>HTC</th>
</tr>
</thead>
<tbody>
<tr>
<td>Male</td>
<td>3-5</td>
<td>3-5</td>
<td>3-5</td>
</tr>
<tr>
<td>Female</td>
<td>2-3</td>
<td>2-4</td>
<td>2-4</td>
</tr>
</tbody>
</table>

Table 9.2: Maximum and minimum scores for the 9.6 Kb/s coders with respect to the test material.

As can be seen from Table 9.1 and 9.2 results obtained for each coder did not vary much. This means that the subjects were consistent with their decisions and hence we can assume that the results were not biased in any way. Average values or the mean opinion scores (MOS) for each of the coders as listed in Table 9.1 and 9.2 are given in Table 9.3 and 9.4.
Table 9.3: MOS's of 16 Kb/s coders (scores out of 5) with respect to the test material.

<table>
<thead>
<tr>
<th>Data</th>
<th>ATC</th>
<th>SBC</th>
<th>PDTC</th>
</tr>
</thead>
<tbody>
<tr>
<td>Male</td>
<td>4.35</td>
<td>4.11</td>
<td>4.33</td>
</tr>
<tr>
<td>Female</td>
<td>3.97</td>
<td>4.31</td>
<td>4.00</td>
</tr>
</tbody>
</table>

Table 9.4: MOS's of 9.6 Kb/s coders (scores out of 5) with respect to the test material.

<table>
<thead>
<tr>
<th>Data</th>
<th>ATC-VQ</th>
<th>SBC-VQ</th>
<th>HTC</th>
</tr>
</thead>
<tbody>
<tr>
<td>Male</td>
<td>3.76</td>
<td>3.38</td>
<td>4.23</td>
</tr>
<tr>
<td>Female</td>
<td>2.72</td>
<td>3.21</td>
<td>3.14</td>
</tr>
</tbody>
</table>

The average MOS's (average of male and female test material) representing the overall coder performance with respect to the transmission rate are tabulated in Table 9.5 and 9.6.

Table 9.5: MOS's of the 16 Kb/s coders for mixed test material.

<table>
<thead>
<tr>
<th>ATC</th>
<th>SBC</th>
<th>PDTC</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.160</td>
<td>4.210</td>
<td>4.165</td>
</tr>
</tbody>
</table>

Table 9.6: MOS's of the 9.6 Kb/s coders for mixed test material.

<table>
<thead>
<tr>
<th>ATC-VQ</th>
<th>SBC-VQ</th>
<th>HTC</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.240</td>
<td>3.295</td>
<td>3.685</td>
</tr>
</tbody>
</table>
The results in Table 9.5 show that all three 16 Kb/s coders have similar MOS's when mixed test data is used. Scores of Table 9.5 mean that all three coders have qualities between GOOD and EXCELLENT. If one is asked to choose one coder out of the listed three it would be very difficult because they all appeared to have similar scores. Here, the results given in Table 9.3 may be used to compare the coders further. From Table 9.3 the performance of ATC across the male and female speakers is not robust. It has an EXCELLENT performance for male speech but FAIR to GOOD performance for the female speech. Similar variations were noticed in PDTC but not as much as with ATC. SBC on the other hand showed much more consistent results for male and female speech. Therefore, the best coder at 16 Kb/s was SBC, which was followed by PDTC and ATC. The reason that ATC and PDTC do not perform as well for female speech is simply because of the pitch variations. In ATC although, a 128 point DCT is used female speech can still produce well defined pitch harmonics. This of course, as was discussed earlier in relation to transform coders, causes clipping and granular errors in quantization. Consequently, this results in aliasing noise as well. Aliasing noise is one of the most annoying distortions heard in frequency domain coders. Well defined pitch harmonics die away as the pitch frequency decreases. This is more serious for female speech than male speech. This is the major cause for ATC and PDTC performing better for male, and not so well for female.

In the sub-band coder, the speech spectrum is split into 16 bands as opposed to 128 and 256 for ATC and PDTC respectively. This does not allow the pitch harmonics to be well defined. Therefore, the signal in each band approximates to random Gaussian for which the APCM quantizers are designed. As a result, SBC has a much more robust performance across the speakers.

In the 9.6 Kb/s coders HTC scored significantly higher MOS for mixed test data. ATC-VQ and SBC-VQ on the other hand were very close. When we look at Table 9.4 however, we can see that SBC is again the most consistent or robust across mixed talkers. Female SBC-VQ scored slightly higher than female HTC, however, the difference was only 0.07. For the male speech on the other hand, HTC scored 0.85 higher than SBC-VQ. Although, for male speech ATC-VQ came closer to HTC its performance for female speech was FAIR and produced the lowest score for females. Therefore, the ordering of the coders by their overall performances, can be listed as HTC, SBC-VQ and ATC-VQ. The reason ATC-VQ came last is exactly the same for that of 16 Kb/s ATC.
9.4 Discussions

We have briefly reviewed subjective testing methods and explained the results of listening tests applied to 6 coders, three at 16 Kb/s and the other three at 9.6 Kb/s. This was a very simple subjective test because it involved similar type coders (all frequency domain) under no error conditions. Therefore, the results obtained are only valid if the coders are used in applications where there is almost no errors in the transmission channel. For applications where the expected error rate can change and at times become severe, the results are not applicable and further tests are needed to evaluate the performance of the coders. For example, in PDTC and HTC the pitch period will need to be protected because the principles of both coders depend on the correct reception of the pitch period.

Another important factor is that the distortion heard in all of the coders tested are very similar because they are all frequency domain coders. The distortions heard are aliasing, quantization noise and band limitation. If there were other coders in which the distortion may be different, the results given above might have been different. Subjects might have given higher or lower scores. These are some of the factors which should be considered for future, wider and more realistic subjective tests.

However, the results obtained confirmed the arguments made whilst designing the coders in chapters 6 and 7. The results give an indication of the coders general performance. At both 16 and 9.6 Kb/s SBC seemed to be very robust across wide range of speakers. At 9.6 Kb/s HTC seemed to be the best coder. ATC on the other hand was not very successful because of its poor performance to female speech.

9.5 References

10.1 Introduction

Following the introduction, in chapter 2 we have briefly discussed digital speech coding as applied to major applications. As the aim of this research program was to investigate the ways of producing high quality speech in the region of 16 to 4.8 Kb/s, in chapter 3, 4 and 5 we have discussed the principles of the existing reduced bit rate speech coding algorithms. The objectives of the research program have been achieved in three steps. In the first we have simulated two 16 Kb/s coders and suggested ways of improving their quality as explained in chapter 6. Second part of the work was to reduce the bit rate one step further whilst maintaining low complexity and high quality. In chapter 7 we discussed various coders which can achieve low to moderate complexity and yet maintain good quality in the 12 to 9.6 Kb/s region. Finally, in chapter 8 we explained how the bit rate can be reduced further to operate in the 8 to 4.8 Kb/s region with good quality and implementable complexity.

10.2 Conclusions

At 16 Kb/s we have concentrated on two frequency domain algorithms, namely the sub-band coder (SBC) and the adaptive transform coder (ATC). In SBC, both forward and backward adaptive quantization of the sub-band signals have been investigated. As the number of bands was increased, the correlation between the samples of the sub-band signals was reduced. Therefore, forward adaptive quantization of the sub-band signals was preferred to backward adaptive quantization. The use of variable bit allocation to the sub-bands enabled the coder to preserve the short term variations in speech better than the fixed bit allocation where the bits were fixed from long time observations. The other important variable in SBC was the number of bands. Although, it was desirable to have as many bands as possible the required side information was increased with the increase in the number of bands (requirement for transmission of band energies for forward adaptive quantization and variable bit allocation). Therefore, compromise was made. 16 bands were found to be the best choice. Some of the bands which were beyond
the signal bandwidth were not transmitted. A 13 band coder (input was filtered at 3.2
KHz) with variable bit allocation and forward adaptive quantizers in each sub-band pro-
duced toll quality 16 Kb/s speech.

In ATC for 16 Kb/s the two well developed Zelinsky and Noll's and the speech
specific (vocoder driven) adaptation strategies have been investigated. Zelinsky and Noll's
approach seemed to be producing excellent 16 Kb/s digital speech for male talkers. How-
ever, the results were not as good for the female talkers. Female speech had its pitch
harmonics well defined even when a 128 point DCT was used and hence caused larger
quantization error. In the recovered speech this caused, back ground noise, aliasing noise
and DCT transform block edge effects. Using a better adaptation method which included
the pitch modelling together with the envelope modelling, i.e. speech specific adaptation,
toll quality was achieved at 16 Kb/s. Although, the speech specific transform coder was
capable of producing toll quality 16 Kb/s digital speech, it was very complex. We pro-
posed an alternative to the very complex speech specific transform coder which we called
the pitch driven transform coder (PDTC). PDTC eliminated the FFT's needed in the
speech specific transform coder and still produced toll quality. It can be said that both
the sub-band coder and various transform coders are capable of producing toll quality
digital speech at 16 Kb/s.

Coders simulated at 16 Kb/s tend to break down at around 12 Kb/s. They start to
have quantization noise, band limitation (due to variable bit allocation), and slight alias-
ing noise. The next bit rate of interest was at 9.6 Kb/s. In order to bridge the gap
between 12 Kb/s and 9.6 Kb/s, in the second stage of the research program we looked at
various algorithms which would produce good quality speech in the 12 to 9.6 Kb/s
region. These are discussed in chapter 7. In chapter 7 we firstly discussed the possible
ways of reducing the bit rate of 16 Kb/s coders to operate in the 12 to 9.6 Kb/s region
and then we proposed two new approaches to the transform coding which produced high
quality speech at bit rates as low as 9.6 Kb/s.

One of the most obvious ways of improving the quality of 16 Kb/s coders (SBC and
ATC) at lower bit rates is to reduce the side information using vector quantization for
coding the band energies. In the vocoder driven ATC and PDTC similar coding can be
applied to the LPC parameters. Using VQ to code the side information in SBC and various
ATC schemes improved the coder performances by about 1 Kb/s. This was expected
because the reduction in the side information was about 1 Kb/s. Vector quantization of
the side information was not sufficient to achieve high quality at 9.6 Kb/s, however, it
may be used in the 16 Kb/s coders to make room for error detection and correction. Vector quantization was also applied to code the residual signal. In SBC we simulated parallel and serial vector quantization. Parallel VQ was preferred to serial because it did not require trained code-books and the bit allocation was achieved by simple weighting whilst vector selection. Parallel VQ used only one code-book for all bands which may be a big advantage under channel errors. Vector quantized SBC produced high quality speech down to about 10 Kb/s. A similar procedure was applied to ATC, but the results obtained were not as good. Vector quantized vocoder driven ATC produced the best result, good quality being achieved at 9.6 Kb/s. No significant improvement was achieved when PDTC was fully vector quantized. Background noise and aliasing noise were the two major distortions heard in PDTC below 12 Kb/s. Therefore, toll quality could not be achieved at 9.6 Kb/s using the 16 Kb/s coders combined with VQ. Vocoder driven ATC seemed to be the best technique producing good quality at 9.6 but with increased complexity. We proposed a new hybrid transform coder (HTC) which was an enhanced version of the PDTC. The major cause of degradation in PDTC was poor quantization of the residual signal. In HTC, 1.5 KHz base-band was quantized accurately, with the help of a created pitch pattern as used in PDTC. High frequencies were effectively regenerated by comparing the base-band signal with the higher frequencies and transmitting side information for the best matching points. High quality speech was achieved at around 9.6 Kb/s, with low complexity. The other new coder which was proposed for 9.6 Kb/s speech coding was weighted vector quantization of the speech residual. This again was a follow-on from PDTC. In PDTC, pitch filter parameters were updated every 256 samples. This was necessary because a 256 point DCT was used to define the pitch harmonics in the frequency domain. In a block of 256 pitch period is not constant. Slight variation in the pitch period reduces the performance of bit allocation and also does not remove the peaks in the spectrum very well. In the new coder, better pitch filtering was achieved in the time domain by calculating the parameters more frequently. Also noise shaping was introduced by means of envelope weighting whilst selecting the optimum vector from the code-book. High quality speech was achieved at 9.6 Kb/s with fairly low complexity.

From the simulation results discussed in chapter 7, it is now fairly obvious that better speech modelling is required if the bit rate is to be reduced below 12 Kb/s. Also important is efficient noise shaping. The most important reason behind the SBC's degraded performance below about 10 Kb/s was that it had no pitch modelling. The
importance of better modelling and noise shaping was demonstrated by the poor performance of PDTC at lower bit rates. Weighted vector quantization of the residual signal with better pitch modelling in the time domain produced high quality speech at 9.6 Kb/s. The differences between the PDTC and weighted vector quantization of the residual signal are better pitch modelling and noise shaping.

Although both the HTC and frequency domain weighted quantization of the residual signal produced high quality speech at around 9.6 Kb/s, they cannot maintain their high quality below 8 Kb/s. They tended to break down at around 8 Kb/s. Speech transmission at 8 Kb/s is very attractive because it is half of the 16 Kb/s transmission rate which is being standardized. Also important is the rate at 4.8 Kb/s. Therefore, the final part of the work was concentrated on algorithms capable of producing high quality speech in the region of 8 to 4.8 Kb/s. Simulation results relating to the 8 to 4.8 Kb/s region were discussed in chapter 8. Here we initially discussed the most promising coder of the last few years, the code excited linear prediction (CELP). Then we proposed two new alternative coders. CELP has produced toll quality speech at 8 Kb/s. However, as the bit rate was reduced quality was also reduced. At around 6 Kb/s CELP was shown to be critical. Below 6 Kb/s its quality was not very good and is not expected to be acceptable for any application at 4.8 Kb/s in its original state. The two major causes of degradation at 4.8 Kb/s were the insufficient update rate of the pitch filter parameters and the large dimensions of the excitation vectors, which caused roughness in the recovered speech. CELP is very complex and cannot be implemented without some simplifications. For this purpose we proposed a new vector quantized transform coder (VQTC) which produced comparable speech quality to CELP but was about 28 to 30 times simpler. VQTC is a further developed version of frequency domain weighted vector quantization of the residual signal. In VQTC the gain and memory of the synthesis filters were considered making it possible to minimize the error between the original and the synthetic speech rather than minimizing the error between the residual vectors and the code-book entries. Although, VQTC produced comparable quality to CELP with lower complexity, its quality was not acceptable at 4.8 Kb/s. Here again the main causes for degradations and poor quality below 6 Kb/s were reduced update rate of the pitch filter parameters and increased residual vector dimensions.

The third coder discussed in chapter 8 was the new CELP base-band coder (CELP-BB). In CELP-BB a suitable base-band was coded and spectral folding was used to create the high frequencies. This allowed capacity for more frequent transmission of the pitch filter
parameters and enabled smaller excitation vectors. CELP-BB produced high quality speech from 8 Kb/s to about 4 Kb/s. The complexity of CELP-BB is well within the capabilities of a single DSP 32 AT&T chip. CELP-BB produced intelligible, natural speech even at 2.4 Kb/s which showed its potential as a replacement for LPC-10.

As the final conclusion to the whole research work we can say that the sub-band coder at 16 Kb/s, Hybrid trasform coder and the weighted frequency domain vector quantization of the speech residual at 9.6 Kb/s and finally, CELP-BB at 8 to 4.8 Kb/s are the most promising coders. They all produce high quality speech at their corresponding bit rates with relatively low complexities.

10.3 Future Work

The work discussed in this thesis may be extended in the following areas:

(i) Although, SBC did not have pitch modelling it produced good quality speech at around 10 Kb/s. Therefore, if an effective way of implementing pitch modelling in SBC can be found, it is almost certain that toll quality speech at 9.6 Kb/s will be achieved.

In CELP, toll quality speech at 8 Kb/s was achieved by minimizing the error between the original speech and the reproduced speech rather than minimizing the error between the residual vectors and the code-book entries. Similar analysis by synthesis implementation may be achieved by sub-band coding with pitch prediction. This may reduce the complexity of CELP. When implementing an SBC in an anlysis by synthesis coding, it may be important to control the delay, filter memory and filter response. These are best controlled by TDAC implementation [1].

(ii) The quality of CELP and VQTC may be improved at 4.8 Kb/s by improving pitch prediction and with better representation of the excitation vectors. Firstly by reducing other information in the coder and allocating it to improve the pitch filter parameters update rate and shortening the excitation vector sizes. Reducing other information may come in the form of vector quantization of both LPC and 3 tap pitch filter parameters or implementation of self excited CELP [8]. Some work has already been reported [2][3][4][5][6][8].

Finally, the quality of CELP type coders may be improved by producing more pulse like LPC residuals when LPC inverse filtering. This will require better LPC analysis. More pulse like residuals will be modelled better by the pitch filter and
hence the increased prediction gain may enable high quality speech at low bit rates. Using a multi-pulse coder high quality speech at low bit rates may be achieved even without using a pitch filter. Some early work has already been reported and looks encouraging [7].

10.4 References


APPENDIX A

PARALLEL FILTER COEFFICIENTS FOR A 16 BAND SUB-BAND CODER.

<table>
<thead>
<tr>
<th>Coefficients</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>$h_1(0)$, $h_1(63)$</td>
<td>-4.0719498E-04</td>
</tr>
<tr>
<td>$h_1(1)$, $h_1(62)$</td>
<td>1.4163303E-04</td>
</tr>
<tr>
<td>$h_1(2)$, $h_1(61)$</td>
<td>-3.0097019E-04</td>
</tr>
<tr>
<td>$h_1(3)$, $h_1(60)$</td>
<td>1.7704129E-04</td>
</tr>
<tr>
<td>$h_1(4)$, $h_1(59)$</td>
<td>-1.4163303E-03</td>
</tr>
<tr>
<td>$h_1(5)$, $h_1(58)$</td>
<td>1.5579634E-03</td>
</tr>
<tr>
<td>$h_1(6)$, $h_1(57)$</td>
<td>-2.5316905E-03</td>
</tr>
<tr>
<td>$h_1(7)$, $h_1(56)$</td>
<td>-2.8857731E-03</td>
</tr>
<tr>
<td>$h_1(8)$, $h_1(55)$</td>
<td>-4.9748602E-03</td>
</tr>
<tr>
<td>$h_1(9)$, $h_1(54)$</td>
<td>-6.2141493E-03</td>
</tr>
<tr>
<td>$h_1(10)$, $h_1(53)$</td>
<td>-7.4357344E-03</td>
</tr>
<tr>
<td>$h_1(11)$, $h_1(52)$</td>
<td>-7.8606335E-03</td>
</tr>
<tr>
<td>$h_1(12)$, $h_1(51)$</td>
<td>-8.7635443E-03</td>
</tr>
<tr>
<td>$h_1(13)$, $h_1(50)$</td>
<td>-9.0822186E-03</td>
</tr>
<tr>
<td>$h_1(14)$, $h_1(49)$</td>
<td>-8.6396150E-03</td>
</tr>
<tr>
<td>$h_1(15)$, $h_1(48)$</td>
<td>-7.4357344E-03</td>
</tr>
<tr>
<td>$h_1(16)$, $h_1(47)$</td>
<td>-5.6830253E-03</td>
</tr>
<tr>
<td>$h_1(17)$, $h_1(46)$</td>
<td>-3.2929680E-03</td>
</tr>
<tr>
<td>$h_1(18)$, $h_1(45)$</td>
<td>4.4260322E-04</td>
</tr>
<tr>
<td>$h_1(19)$, $h_1(44)$</td>
<td>5.0987895E-03</td>
</tr>
<tr>
<td>$h_1(20)$, $h_1(43)$</td>
<td>1.0799519E-02</td>
</tr>
<tr>
<td>$h_1(21)$, $h_1(42)$</td>
<td>1.7049076E-02</td>
</tr>
<tr>
<td>$h_1(22)$, $h_1(41)$</td>
<td>2.4130728E-02</td>
</tr>
<tr>
<td>$h_1(23)$, $h_1(40)$</td>
<td>3.1495646E-02</td>
</tr>
<tr>
<td>$h_1(24)$, $h_1(39)$</td>
<td>3.8984492E-02</td>
</tr>
<tr>
<td>$h_1(25)$, $h_1(38)$</td>
<td>4.6207778E-02</td>
</tr>
<tr>
<td>$h_1(26)$, $h_1(37)$</td>
<td>5.3041570E-02</td>
</tr>
<tr>
<td>$h_1(27)$, $h_1(36)$</td>
<td>5.9149496E-02</td>
</tr>
<tr>
<td>$h_1(28)$, $h_1(35)$</td>
<td>6.4301394E-02</td>
</tr>
<tr>
<td>$h_1(29)$, $h_1(34)$</td>
<td>6.8320237E-02</td>
</tr>
<tr>
<td>$h_1(30)$, $h_1(33)$</td>
<td>7.1097808E-02</td>
</tr>
<tr>
<td>$h_1(31)$, $h_1(32)$</td>
<td>7.2516114E-02</td>
</tr>
</tbody>
</table>

Transmit and receive bandpass filter coefficients for bandpass filter no. 1; even symmetry.
Transmit and receive bandpass filter coefficients for bandpass filter no. 2; odd symmetry.

<table>
<thead>
<tr>
<th>Coefficients</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>$h_2(0)$, $h_2(63)$</td>
<td>-7.2306942E-04</td>
</tr>
<tr>
<td>$h_2(1)$, $h_2(62)$</td>
<td>-1.6992131E-03</td>
</tr>
<tr>
<td>$h_2(2)$, $h_2(61)$</td>
<td>-2.0788244E-03</td>
</tr>
<tr>
<td>$h_2(3)$, $h_2(60)$</td>
<td>-2.4042057E-03</td>
</tr>
<tr>
<td>$h_2(4)$, $h_2(59)$</td>
<td>-1.4461388E-03</td>
</tr>
<tr>
<td>$h_2(5)$, $h_2(58)$</td>
<td>-1.8799804E-03</td>
</tr>
<tr>
<td>$h_2(6)$, $h_2(57)$</td>
<td>-6.8691594E-04</td>
</tr>
<tr>
<td>$h_2(7)$, $h_2(56)$</td>
<td>-3.6153471E-04</td>
</tr>
<tr>
<td>$h_2(8)$, $h_2(55)$</td>
<td>1.5184457E-03</td>
</tr>
<tr>
<td>$h_2(9)$, $h_2(54)$</td>
<td>1.6088294E-03</td>
</tr>
<tr>
<td>$h_2(10)$, $h_2(53)$</td>
<td>9.7614370E-04</td>
</tr>
<tr>
<td>$h_2(11)$, $h_2(52)$</td>
<td>-1.5545991E-03</td>
</tr>
<tr>
<td>$h_2(12)$, $h_2(51)$</td>
<td>-5.2060997E-03</td>
</tr>
<tr>
<td>$h_2(13)$, $h_2(50)$</td>
<td>-9.7433599E-03</td>
</tr>
<tr>
<td>$h_2(14)$, $h_2(49)$</td>
<td>-1.5582145E-02</td>
</tr>
<tr>
<td>$h_2(15)$, $h_2(48)$</td>
<td>-2.1059396E-02</td>
</tr>
<tr>
<td>$h_2(16)$, $h_2(47)$</td>
<td>-2.5867807E-02</td>
</tr>
<tr>
<td>$h_2(17)$, $h_2(46)$</td>
<td>-2.7603174E-02</td>
</tr>
<tr>
<td>$h_2(18)$, $h_2(45)$</td>
<td>-2.6283571E-02</td>
</tr>
<tr>
<td>$h_2(19)$, $h_2(44)$</td>
<td>-2.6625554E-02</td>
</tr>
<tr>
<td>$h_2(20)$, $h_2(43)$</td>
<td>-1.0972578E-02</td>
</tr>
<tr>
<td>$h_2(21)$, $h_2(42)$</td>
<td>2.6392033E-03</td>
</tr>
<tr>
<td>$h_2(22)$, $h_2(41)$</td>
<td>1.8944418E-02</td>
</tr>
<tr>
<td>$h_2(23)$, $h_2(40)$</td>
<td>3.6225777E-02</td>
</tr>
<tr>
<td>$h_2(24)$, $h_2(39)$</td>
<td>5.2404452E-02</td>
</tr>
<tr>
<td>$h_2(25)$, $h_2(38)$</td>
<td>6.5311246E-02</td>
</tr>
<tr>
<td>$h_2(26)$, $h_2(37)$</td>
<td>7.2993852E-02</td>
</tr>
<tr>
<td>$h_2(27)$, $h_2(36)$</td>
<td>7.4042305E-02</td>
</tr>
<tr>
<td>$h_2(28)$, $h_2(35)$</td>
<td>6.7733526E-02</td>
</tr>
<tr>
<td>$h_2(29)$, $h_2(34)$</td>
<td>5.4429047E-02</td>
</tr>
<tr>
<td>$h_2(30)$, $h_2(33)$</td>
<td>3.5231557E-02</td>
</tr>
<tr>
<td>$h_2(31)$, $h_2(32)$</td>
<td>1.2219872E-02</td>
</tr>
<tr>
<td>Coefficients</td>
<td>Value</td>
</tr>
<tr>
<td>--------------</td>
<td>---------------</td>
</tr>
<tr>
<td>$h_0$, $h_{63}$</td>
<td>$-9.8303577E-04$</td>
</tr>
<tr>
<td>$h_1$, $h_{62}$</td>
<td>$-1.3762502E-03$</td>
</tr>
<tr>
<td>$h_2$, $h_{61}$</td>
<td>$-1.9660716E-05$</td>
</tr>
<tr>
<td>$h_3$, $h_{60}$</td>
<td>$3.3423217E-04$</td>
</tr>
<tr>
<td>$h_4$, $h_{59}$</td>
<td>$1.2582858E-03$</td>
</tr>
<tr>
<td>$h_5$, $h_{58}$</td>
<td>$1.9070895E-03$</td>
</tr>
<tr>
<td>$h_6$, $h_{57}$</td>
<td>$9.2405366E-04$</td>
</tr>
<tr>
<td>$h_7$, $h_{56}$</td>
<td>$5.3089323E-04$</td>
</tr>
<tr>
<td>$h_8$, $h_{55}$</td>
<td>$-1.0026966E-03$</td>
</tr>
<tr>
<td>$h_9$, $h_{54}$</td>
<td>$-1.3762501E-04$</td>
</tr>
<tr>
<td>$h_{10}$, $h_{53}$</td>
<td>$2.0447145E-03$</td>
</tr>
<tr>
<td>$h_{11}$, $h_{52}$</td>
<td>$5.9375362E-03$</td>
</tr>
<tr>
<td>$h_{12}$, $h_{51}$</td>
<td>$1.0439840E-02$</td>
</tr>
<tr>
<td>$h_{13}$, $h_{50}$</td>
<td>$1.2504215E-02$</td>
</tr>
<tr>
<td>$h_{14}$, $h_{49}$</td>
<td>$1.1619483E-02$</td>
</tr>
<tr>
<td>$h_{15}$, $h_{48}$</td>
<td>$5.0724647E-03$</td>
</tr>
<tr>
<td>$h_{16}$, $h_{47}$</td>
<td>$-5.9571969E-03$</td>
</tr>
<tr>
<td>$h_{17}$, $h_{46}$</td>
<td>$-1.9994948E-02$</td>
</tr>
<tr>
<td>$h_{18}$, $h_{45}$</td>
<td>$-3.0496655E-02$</td>
</tr>
<tr>
<td>$h_{19}$, $h_{44}$</td>
<td>$-3.9989966E-02$</td>
</tr>
<tr>
<td>$h_{20}$, $h_{43}$</td>
<td>$-3.7021308E-02$</td>
</tr>
<tr>
<td>$h_{21}$, $h_{42}$</td>
<td>$-2.218306E-02$</td>
</tr>
<tr>
<td>$h_{22}$, $h_{41}$</td>
<td>$2.3592860E-03$</td>
</tr>
<tr>
<td>$h_{23}$, $h_{40}$</td>
<td>$3.1004949E-02$</td>
</tr>
<tr>
<td>$h_{24}$, $h_{39}$</td>
<td>$5.5915076E-02$</td>
</tr>
<tr>
<td>$h_{25}$, $h_{38}$</td>
<td>$6.9382668E-02$</td>
</tr>
<tr>
<td>$h_{26}$, $h_{37}$</td>
<td>$6.6138647E-02$</td>
</tr>
<tr>
<td>$h_{27}$, $h_{36}$</td>
<td>$4.5435917E-02$</td>
</tr>
<tr>
<td>$h_{28}$, $h_{35}$</td>
<td>$1.1462198E-02$</td>
</tr>
<tr>
<td>$h_{29}$, $h_{34}$</td>
<td>$-2.7407039E-02$</td>
</tr>
<tr>
<td>$h_{30}$, $h_{33}$</td>
<td>$-6.1085846E-02$</td>
</tr>
<tr>
<td>$h_{31}$, $h_{32}$</td>
<td>$-8.0530293E-02$</td>
</tr>
</tbody>
</table>

Transmit and receive bandpass filter coefficients for bandpass filter no. 3; even symmetry.
<table>
<thead>
<tr>
<th>Coefficients</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>$h_4(0)$, $h_4(63)$</td>
<td>1.4798430E-03</td>
</tr>
<tr>
<td>$h_4(1)$, $h_4(62)$</td>
<td>2.4339524E-03</td>
</tr>
<tr>
<td>$h_4(2)$, $h_4(61)$</td>
<td>1.8108964E-03</td>
</tr>
<tr>
<td>$h_4(3)$, $h_4(60)$</td>
<td>1.1682971E-03</td>
</tr>
<tr>
<td>$h_4(4)$, $h_4(59)$</td>
<td>-1.3435418E-03</td>
</tr>
<tr>
<td>$h_4(5)$, $h_4(58)$</td>
<td>-2.0250485E-03</td>
</tr>
<tr>
<td>$h_4(6)$, $h_4(57)$</td>
<td>-1.1293593E-03</td>
</tr>
<tr>
<td>$h_4(7)$, $h_4(56)$</td>
<td>1.7524458E-04</td>
</tr>
<tr>
<td>$h_4(8)$, $h_4(55)$</td>
<td>1.9471620E-03</td>
</tr>
<tr>
<td>$h_4(9)$, $h_4(54)$</td>
<td>1.5577296E-04</td>
</tr>
<tr>
<td>$h_4(10)$, $h_4(53)$</td>
<td>-3.7774942E-03</td>
</tr>
<tr>
<td>$h_4(11)$, $h_4(52)$</td>
<td>-7.8275912E-03</td>
</tr>
<tr>
<td>$h_4(12)$, $h_4(51)$</td>
<td>-8.985302E-03</td>
</tr>
<tr>
<td>$h_4(13)$, $h_4(50)$</td>
<td>-2.657619E-03</td>
</tr>
<tr>
<td>$h_4(14)$, $h_4(49)$</td>
<td>8.6648706E-03</td>
</tr>
<tr>
<td>$h_4(15)$, $h_4(48)$</td>
<td>2.0386785E-02</td>
</tr>
<tr>
<td>$h_4(16)$, $h_4(47)$</td>
<td>2.4767900E-02</td>
</tr>
<tr>
<td>$h_4(17)$, $h_4(46)$</td>
<td>1.5479937E-02</td>
</tr>
<tr>
<td>$h_4(18)$, $h_4(45)$</td>
<td>-5.7830708E-03</td>
</tr>
<tr>
<td>$h_4(19)$, $h_4(44)$</td>
<td>-3.0804101E-02</td>
</tr>
<tr>
<td>$h_4(20)$, $h_4(43)$</td>
<td>-4.6069853E-02</td>
</tr>
<tr>
<td>$h_4(21)$, $h_4(42)$</td>
<td>-4.0500969E-02</td>
</tr>
<tr>
<td>$h_4(22)$, $h_4(41)$</td>
<td>-1.2792854E-02</td>
</tr>
<tr>
<td>$h_4(23)$, $h_4(40)$</td>
<td>2.6831891E-02</td>
</tr>
<tr>
<td>$h_4(24)$, $h_4(39)$</td>
<td>5.9466325E-02</td>
</tr>
<tr>
<td>$h_4(25)$, $h_4(38)$</td>
<td>6.7313388E-02</td>
</tr>
<tr>
<td>$h_4(26)$, $h_4(37)$</td>
<td>4.324667E-02</td>
</tr>
<tr>
<td>$h_4(27)$, $h_4(36)$</td>
<td>-3.9722105E-03</td>
</tr>
<tr>
<td>$h_4(28)$, $h_4(35)$</td>
<td>-5.2982278E-02</td>
</tr>
<tr>
<td>$h_4(29)$, $h_4(34)$</td>
<td>-7.9755753E-02</td>
</tr>
<tr>
<td>$h_4(30)$, $h_4(33)$</td>
<td>-7.0214559E-02</td>
</tr>
<tr>
<td>$h_4(31)$, $h_4(32)$</td>
<td>-2.7824944E-02</td>
</tr>
</tbody>
</table>

Transmit and receive bandpass filter coefficients for bandpass filter no. 4; odd symmetry.
<table>
<thead>
<tr>
<th>Coefficients</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>$h_5^<em>(0)$, $h_5^</em>(63)$</td>
<td>$1.5736382E-03$</td>
</tr>
<tr>
<td>$h_5^<em>(1)$, $h_5^</em>(62)$</td>
<td>$9.2115405E-04$</td>
</tr>
<tr>
<td>$h_5^<em>(2)$, $h_5^</em>(61)$</td>
<td>$-1.3625404E-03$</td>
</tr>
<tr>
<td>$h_5^<em>(3)$, $h_5^</em>(60)$</td>
<td>$-2.5139828E-03$</td>
</tr>
<tr>
<td>$h_5^<em>(4)$, $h_5^</em>(59)$</td>
<td>$-1.4776846E-03$</td>
</tr>
<tr>
<td>$h_5^<em>(5)$, $h_5^</em>(58)$</td>
<td>$-6.5248413E-04$</td>
</tr>
<tr>
<td>$h_5^<em>(6)$, $h_5^</em>(57)$</td>
<td>$5.5653055E-04$</td>
</tr>
<tr>
<td>$h_5^<em>(7)$, $h_5^</em>(56)$</td>
<td>$6.5248413E-04$</td>
</tr>
<tr>
<td>$h_5^<em>(8)$, $h_5^</em>(55)$</td>
<td>$-1.7271639E-04$</td>
</tr>
<tr>
<td>$h_5^<em>(9)$, $h_5^</em>(54)$</td>
<td>$2.1685502E-03$</td>
</tr>
<tr>
<td>$h_5^<em>(10)$, $h_5^</em>(53)$</td>
<td>$5.5461149E-03$</td>
</tr>
<tr>
<td>$h_5^<em>(11)$, $h_5^</em>(52)$</td>
<td>$5.3158263E-03$</td>
</tr>
<tr>
<td>$h_5^<em>(12)$, $h_5^</em>(51)$</td>
<td>$-8.8277261E-04$</td>
</tr>
<tr>
<td>$h_5^<em>(13)$, $h_5^</em>(50)$</td>
<td>$-1.1629570E-02$</td>
</tr>
<tr>
<td>$h_5^<em>(14)$, $h_5^</em>(49)$</td>
<td>$-1.7329210E-02$</td>
</tr>
<tr>
<td>$h_5^<em>(15)$, $h_5^</em>(48)$</td>
<td>$-9.1155870E-03$</td>
</tr>
<tr>
<td>$h_5^<em>(16)$, $h_5^</em>(47)$</td>
<td>$1.0900323E-02$</td>
</tr>
<tr>
<td>$h_5^<em>(17)$, $h_5^</em>(46)$</td>
<td>$2.9707218E-02$</td>
</tr>
<tr>
<td>$h_5^<em>(18)$, $h_5^</em>(45)$</td>
<td>$2.9227450E-02$</td>
</tr>
<tr>
<td>$h_5^<em>(19)$, $h_5^</em>(44)$</td>
<td>$2.8977972E-03$</td>
</tr>
<tr>
<td>$h_5^<em>(20)$, $h_5^</em>(43)$</td>
<td>$-3.3430215E-02$</td>
</tr>
<tr>
<td>$h_5^<em>(21)$, $h_5^</em>(42)$</td>
<td>$-5.0778616E-02$</td>
</tr>
<tr>
<td>$h_5^<em>(22)$, $h_5^</em>(41)$</td>
<td>$-2.9726408E-02$</td>
</tr>
<tr>
<td>$h_5^<em>(23)$, $h_5^</em>(40)$</td>
<td>$1.9785622E-02$</td>
</tr>
<tr>
<td>$h_5^<em>(24)$, $h_5^</em>(39)$</td>
<td>$6.1371899E-02$</td>
</tr>
<tr>
<td>$h_5^<em>(25)$, $h_5^</em>(38)$</td>
<td>$5.9875011E-02$</td>
</tr>
<tr>
<td>$h_5^<em>(26)$, $h_5^</em>(37)$</td>
<td>$1.1092230E-02$</td>
</tr>
<tr>
<td>$h_5^<em>(27)$, $h_5^</em>(36)$</td>
<td>$-5.0970525E-02$</td>
</tr>
<tr>
<td>$h_5^<em>(28)$, $h_5^</em>(35)$</td>
<td>$-7.8605145E-02$</td>
</tr>
<tr>
<td>$h_5^<em>(29)$, $h_5^</em>(34)$</td>
<td>$-4.8091918E-02$</td>
</tr>
<tr>
<td>$h_5^<em>(30)$, $h_5^</em>(33)$</td>
<td>$1.9881574E-02$</td>
</tr>
<tr>
<td>$h_5^<em>(31)$, $h_5^</em>(32)$</td>
<td>$7.4613474E-02$</td>
</tr>
</tbody>
</table>

Transmit and receive bandpass filter coefficients for bandpass filter no. 5; even symmetry.
<table>
<thead>
<tr>
<th>Coefficients</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>h_{6} (0), h_{6} (63)</td>
<td>-1.1043159E-03</td>
</tr>
<tr>
<td>h_{6} (1), h_{6} (62)</td>
<td>-2.2086317E-03</td>
</tr>
<tr>
<td>h_{6} (2), h_{6} (61)</td>
<td>-2.4094165E-04</td>
</tr>
<tr>
<td>h_{6} (3), h_{6} (60)</td>
<td>1.5861992E-03</td>
</tr>
<tr>
<td>h_{6} (4), h_{6} (59)</td>
<td>1.8773763E-03</td>
</tr>
<tr>
<td>h_{6} (5), h_{6} (58)</td>
<td>5.4211874E-04</td>
</tr>
<tr>
<td>h_{6} (6), h_{6} (57)</td>
<td>-1.3653360E-03</td>
</tr>
<tr>
<td>h_{6} (7), h_{6} (56)</td>
<td>-6.2243262E-04</td>
</tr>
<tr>
<td>h_{6} (8), h_{6} (55)</td>
<td>7.8306039E-04</td>
</tr>
<tr>
<td>h_{6} (9), h_{6} (54)</td>
<td>-1.8271408E-03</td>
</tr>
<tr>
<td>h_{6} (10), h_{6} (53)</td>
<td>-4.8991470E-03</td>
</tr>
<tr>
<td>h_{6} (11), h_{6} (52)</td>
<td>-1.2850221E-03</td>
</tr>
<tr>
<td>h_{6} (12), h_{6} (51)</td>
<td>7.9309959E-03</td>
</tr>
<tr>
<td>h_{6} (13), h_{6} (50)</td>
<td>1.2187632E-02</td>
</tr>
<tr>
<td>h_{6} (14), h_{6} (49)</td>
<td>1.3051006E-03</td>
</tr>
<tr>
<td>h_{6} (15), h_{6} (48)</td>
<td>-1.7950153E-02</td>
</tr>
<tr>
<td>h_{6} (16), h_{6} (47)</td>
<td>-2.1885533E-02</td>
</tr>
<tr>
<td>h_{6} (17), h_{6} (46)</td>
<td>1.7669054E-03</td>
</tr>
<tr>
<td>h_{6} (18), h_{6} (45)</td>
<td>3.1964924E-02</td>
</tr>
<tr>
<td>h_{6} (19), h_{6} (44)</td>
<td>3.2085396E-02</td>
</tr>
<tr>
<td>h_{6} (20), h_{6} (43)</td>
<td>-7.1077785E-03</td>
</tr>
<tr>
<td>h_{6} (21), h_{6} (42)</td>
<td>-4.7967467E-02</td>
</tr>
<tr>
<td>h_{6} (22), h_{6} (41)</td>
<td>-4.1180942E-02</td>
</tr>
<tr>
<td>h_{6} (23), h_{6} (40)</td>
<td>1.5018696E-02</td>
</tr>
<tr>
<td>h_{6} (24), h_{6} (39)</td>
<td>6.3367650E-02</td>
</tr>
<tr>
<td>h_{6} (25), h_{6} (38)</td>
<td>4.6622209E-02</td>
</tr>
<tr>
<td>h_{6} (26), h_{6} (37)</td>
<td>-2.4636284E-02</td>
</tr>
<tr>
<td>h_{6} (27), h_{6} (36)</td>
<td>-7.5575367E-02</td>
</tr>
<tr>
<td>h_{6} (28), h_{6} (35)</td>
<td>-4.7144249E-02</td>
</tr>
<tr>
<td>h_{6} (29), h_{6} (34)</td>
<td>3.4514893E-02</td>
</tr>
<tr>
<td>h_{6} (30), h_{6} (33)</td>
<td>8.2241416E-02</td>
</tr>
<tr>
<td>h_{6} (31), h_{6} (32)</td>
<td>4.2726986E-02</td>
</tr>
</tbody>
</table>

Transmit and receive bandpass filter coefficients for bandpass filter no. 6; odd symmetry.
<table>
<thead>
<tr>
<th>Coefficients</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>$h_7(0)$</td>
<td>-1.2447799E-03</td>
</tr>
<tr>
<td>$h_7(1)$</td>
<td>5.9275230E-04</td>
</tr>
<tr>
<td>$h_7(2)$</td>
<td>2.2327004E-03</td>
</tr>
<tr>
<td>$h_7(3)$</td>
<td>1.1064709E-03</td>
</tr>
<tr>
<td>$h_7(4)$</td>
<td>-1.4818808E-03</td>
</tr>
<tr>
<td>$h_7(5)$</td>
<td>-1.3435719E-03</td>
</tr>
<tr>
<td>$h_7(6)$</td>
<td>7.9033640E-04</td>
</tr>
<tr>
<td>$h_7(7)$</td>
<td>7.5081957E-04</td>
</tr>
<tr>
<td>$h_7(8)$</td>
<td>2.1734252E-04</td>
</tr>
<tr>
<td>$h_7(9)$</td>
<td>2.7859358E-03</td>
</tr>
<tr>
<td>$h_7(10)$</td>
<td>3.2403793E-03</td>
</tr>
<tr>
<td>$h_7(11)$</td>
<td>-4.0899911E-03</td>
</tr>
<tr>
<td>$h_7(12)$</td>
<td>-9.8792054E-03</td>
</tr>
<tr>
<td>$h_7(13)$</td>
<td>-3.7540979E-04</td>
</tr>
<tr>
<td>$h_7(14)$</td>
<td>1.6043829E-02</td>
</tr>
<tr>
<td>$h_7(15)$</td>
<td>1.2368765E-02</td>
</tr>
<tr>
<td>$h_7(16)$</td>
<td>-1.5233735E-02</td>
</tr>
<tr>
<td>$h_7(17)$</td>
<td>-2.8669454E-02</td>
</tr>
<tr>
<td>$h_7(18)$</td>
<td>1.3040551E-03</td>
</tr>
<tr>
<td>$h_7(19)$</td>
<td>3.9042618E-02</td>
</tr>
<tr>
<td>$h_7(20)$</td>
<td>2.3749609E-02</td>
</tr>
<tr>
<td>$h_7(21)$</td>
<td>-3.3865914E-02</td>
</tr>
<tr>
<td>$h_7(22)$</td>
<td>-5.0601289E-02</td>
</tr>
<tr>
<td>$h_7(23)$</td>
<td>8.6937007E-03</td>
</tr>
<tr>
<td>$h_7(24)$</td>
<td>6.4511210E-02</td>
</tr>
<tr>
<td>$h_7(25)$</td>
<td>2.9756166E-02</td>
</tr>
<tr>
<td>$h_7(26)$</td>
<td>-5.4217078E-02</td>
</tr>
<tr>
<td>$h_7(27)$</td>
<td>-6.5499127E-02</td>
</tr>
<tr>
<td>$h_7(28)$</td>
<td>1.9264450E-02</td>
</tr>
<tr>
<td>$h_7(29)$</td>
<td>8.0930449E-02</td>
</tr>
<tr>
<td>$h_7(30)$</td>
<td>2.7681533E-02</td>
</tr>
<tr>
<td>$h_7(31)$</td>
<td>-6.6743910E-02</td>
</tr>
</tbody>
</table>

Transmit and receive bandpass filter coefficients for bandpass filter no. 7; even symmetry.
Transmit and receive bandpass filter coefficients for bandpass filter no. 8; odd symmetry.

<table>
<thead>
<tr>
<th>Coefficients</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>$h(0), h(63)$</td>
<td>$1.5949077E-03$</td>
</tr>
<tr>
<td>$h(1), h(62)$</td>
<td>$1.1095010E-03$</td>
</tr>
<tr>
<td>$h(2), h(61)$</td>
<td>$-2.3230177E-03$</td>
</tr>
<tr>
<td>$h(3), h(60)$</td>
<td>$-1.8549470E-03$</td>
</tr>
<tr>
<td>$h(4), h(59)$</td>
<td>$1.6295796E-03$</td>
</tr>
<tr>
<td>$h(5), h(58)$</td>
<td>$8.6679764E-04$</td>
</tr>
<tr>
<td>$h(6), h(57)$</td>
<td>$-1.8722829E-03$</td>
</tr>
<tr>
<td>$h(7), h(56)$</td>
<td>$6.9343812E-05$</td>
</tr>
<tr>
<td>$h(8), h(55)$</td>
<td>$1.1615088E-03$</td>
</tr>
<tr>
<td>$h(9), h(54)$</td>
<td>$-3.6232141E-03$</td>
</tr>
<tr>
<td>$h(10), h(53)$</td>
<td>$-2.6177289E-03$</td>
</tr>
<tr>
<td>$h(11), h(52)$</td>
<td>$7.6624914E-03$</td>
</tr>
<tr>
<td>$h(12), h(51)$</td>
<td>$5.6515206E-03$</td>
</tr>
<tr>
<td>$h(13), h(50)$</td>
<td>$-1.2412542E-02$</td>
</tr>
<tr>
<td>$h(14), h(49)$</td>
<td>$-1.1337713E-02$</td>
</tr>
<tr>
<td>$h(15), h(48)$</td>
<td>$1.6555835E-02$</td>
</tr>
<tr>
<td>$h(16), h(47)$</td>
<td>$1.9606963E-02$</td>
</tr>
<tr>
<td>$h(17), h(46)$</td>
<td>$-1.9208236E-02$</td>
</tr>
<tr>
<td>$h(18), h(45)$</td>
<td>$-3.0511277E-02$</td>
</tr>
<tr>
<td>$h(19), h(44)$</td>
<td>$1.8861517E-02$</td>
</tr>
<tr>
<td>$h(20), h(43)$</td>
<td>$4.3235868E-02$</td>
</tr>
<tr>
<td>$h(21), h(42)$</td>
<td>$-1.3903434E-02$</td>
</tr>
<tr>
<td>$h(22), h(41)$</td>
<td>$-5.5509720E-02$</td>
</tr>
<tr>
<td>$h(23), h(40)$</td>
<td>$4.3513244E-03$</td>
</tr>
<tr>
<td>$h(24), h(39)$</td>
<td>$6.5391213E-02$</td>
</tr>
<tr>
<td>$h(25), h(38)$</td>
<td>$9.1880511E-03$</td>
</tr>
<tr>
<td>$h(26), h(37)$</td>
<td>$-7.0886709E-02$</td>
</tr>
<tr>
<td>$h(27), h(36)$</td>
<td>$-2.4946436E-02$</td>
</tr>
<tr>
<td>$h(28), h(35)$</td>
<td>$7.1008064E-02$</td>
</tr>
<tr>
<td>$h(29), h(34)$</td>
<td>$4.0964857E-02$</td>
</tr>
<tr>
<td>$h(30), h(33)$</td>
<td>$-6.5512568E-02$</td>
</tr>
<tr>
<td>$h(31), h(32)$</td>
<td>$-5.5058986E-02$</td>
</tr>
<tr>
<td>Coefficients</td>
<td>Value</td>
</tr>
<tr>
<td>--------------</td>
<td>------------</td>
</tr>
<tr>
<td>h'<em>(0), h'</em>(63)</td>
<td>1.5949077E-03</td>
</tr>
<tr>
<td>h'<em>(1), h'</em>(62)</td>
<td>-1.1095010E-03</td>
</tr>
<tr>
<td>h'<em>(2), h'</em>(61)</td>
<td>-2.3230177E-03</td>
</tr>
<tr>
<td>h'<em>(3), h'</em>(60)</td>
<td>1.8549470E-03</td>
</tr>
<tr>
<td>h'<em>(4), h'</em>(59)</td>
<td>1.6295796E-03</td>
</tr>
<tr>
<td>h'<em>(5), h'</em>(58)</td>
<td>-8.6679764E-04</td>
</tr>
<tr>
<td>h'<em>(6), h'</em>(57)</td>
<td>-1.8722829E-03</td>
</tr>
<tr>
<td>h'<em>(7), h'</em>(56)</td>
<td>-6.9343812E-05</td>
</tr>
<tr>
<td>h'<em>(8), h'</em>(55)</td>
<td>1.1615088E-03</td>
</tr>
<tr>
<td>h'<em>(9), h'</em>(54)</td>
<td>3.6232141E-03</td>
</tr>
<tr>
<td>h'<em>(10), h'</em>(53)</td>
<td>-2.6177289E-03</td>
</tr>
<tr>
<td>h'<em>(11), h'</em> (52)</td>
<td>-7.6624914E-03</td>
</tr>
<tr>
<td>h'<em>(12), h'</em> (51)</td>
<td>5.6515206E-03</td>
</tr>
<tr>
<td>h'<em>(13), h'</em> (50)</td>
<td>1.2412542E-02</td>
</tr>
<tr>
<td>h'<em>(14), h'</em>(49)</td>
<td>-1.133713E-02</td>
</tr>
<tr>
<td>h'<em>(15), h'</em> (48)</td>
<td>-1.6555835E-02</td>
</tr>
<tr>
<td>h'<em>(16), h'</em> (47)</td>
<td>1.9606963E-02</td>
</tr>
<tr>
<td>h'<em>(17), h'</em> (46)</td>
<td>1.9208236E-02</td>
</tr>
<tr>
<td>h'<em>(18), h'</em> (45)</td>
<td>-3.0511277E-02</td>
</tr>
<tr>
<td>h'<em>(19), h'</em> (44)</td>
<td>-1.8861517E-02</td>
</tr>
<tr>
<td>h'<em>(20), h'</em> (43)</td>
<td>4.3235868E-02</td>
</tr>
<tr>
<td>h'<em>(21), h'</em>(42)</td>
<td>1.3903434E-02</td>
</tr>
<tr>
<td>h'<em>(22), h'</em> (41)</td>
<td>-5.5509720E-02</td>
</tr>
<tr>
<td>h'<em>(23), h'</em>(40)</td>
<td>-4.3513244E-03</td>
</tr>
<tr>
<td>h'<em>(24), h'</em> (39)</td>
<td>6.5391213E-02</td>
</tr>
<tr>
<td>h'<em>(25), h'</em> (38)</td>
<td>-9.1880515E-03</td>
</tr>
<tr>
<td>h'<em>(26), h'</em>(37)</td>
<td>-7.0886709E-02</td>
</tr>
<tr>
<td>h'<em>(27), h'</em> (36)</td>
<td>2.4946436E-02</td>
</tr>
<tr>
<td>h'<em>(28), h'</em> (35)</td>
<td>7.1008064E-02</td>
</tr>
<tr>
<td>h'<em>(29), h'</em> (34)</td>
<td>-4.0964857E-02</td>
</tr>
<tr>
<td>h'<em>(30), h'</em> (33)</td>
<td>-6.5512568E-02</td>
</tr>
<tr>
<td>h'<em>(31), h'</em> (32)</td>
<td>5.5058986E-02</td>
</tr>
</tbody>
</table>

Transmit and receive bandpass filter coefficients for bandpass filter no. 9; even symmetry.
<table>
<thead>
<tr>
<th>Coefficients</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>$h'_0(0)$, $h'_0(63)$</td>
<td>-1.2447799E-03</td>
</tr>
<tr>
<td>$h'_0(1)$, $h'_0(62)$</td>
<td>-5.9275230E-04</td>
</tr>
<tr>
<td>$h'_0(2)$, $h'_0(61)$</td>
<td>2.2327004E-03</td>
</tr>
<tr>
<td>$h'_0(3)$, $h'_0(60)$</td>
<td>-1.1064709E-03</td>
</tr>
<tr>
<td>$h'_0(4)$, $h'_0(59)$</td>
<td>-1.4818808E-03</td>
</tr>
<tr>
<td>$h'_0(5)$, $h'_0(58)$</td>
<td>1.3435719E-03</td>
</tr>
<tr>
<td>$h'_0(6)$, $h'_0(57)$</td>
<td>7.9033640E-04</td>
</tr>
<tr>
<td>$h'_0(7)$, $h'_0(56)$</td>
<td>-7.5081957E-04</td>
</tr>
<tr>
<td>$h'_0(8)$, $h'_0(55)$</td>
<td>2.1734252E-04</td>
</tr>
<tr>
<td>$h'_0(9)$, $h'_0(54)$</td>
<td>-2.7859358E-03</td>
</tr>
<tr>
<td>$h'_0(10)$, $h'_0(53)$</td>
<td>3.2403793E-03</td>
</tr>
<tr>
<td>$h'_0(11)$, $h'_0(52)$</td>
<td>4.0899911E-03</td>
</tr>
<tr>
<td>$h'_0(12)$, $h'_0(51)$</td>
<td>-9.8792054E-03</td>
</tr>
<tr>
<td>$h'_0(13)$, $h'_0(50)$</td>
<td>3.7540979E-04</td>
</tr>
<tr>
<td>$h'_0(14)$, $h'_0(49)$</td>
<td>1.6043829E-02</td>
</tr>
<tr>
<td>$h'_0(15)$, $h'_0(48)$</td>
<td>-1.2368765E-02</td>
</tr>
<tr>
<td>$h'_0(16)$, $h'_0(47)$</td>
<td>-1.5233735E-02</td>
</tr>
<tr>
<td>$h'_0(17)$, $h'_0(46)$</td>
<td>2.8669454E-02</td>
</tr>
<tr>
<td>$h'_0(18)$, $h'_0(45)$</td>
<td>1.3040551E-03</td>
</tr>
<tr>
<td>$h'_0(19)$, $h'_0(44)$</td>
<td>-3.9042618E-02</td>
</tr>
<tr>
<td>$h'_0(20)$, $h'_0(43)$</td>
<td>2.3749609E-02</td>
</tr>
<tr>
<td>$h'_0(21)$, $h'_0(42)$</td>
<td>3.3865914E-02</td>
</tr>
<tr>
<td>$h'_0(22)$, $h'_0(41)$</td>
<td>-5.0601289E-02</td>
</tr>
<tr>
<td>$h'_0(23)$, $h'_0(40)$</td>
<td>-8.6937007E-03</td>
</tr>
<tr>
<td>$h'_0(24)$, $h'_0(39)$</td>
<td>6.4511210E-02</td>
</tr>
<tr>
<td>$h'_0(25)$, $h'_0(38)$</td>
<td>-2.9756166E-02</td>
</tr>
<tr>
<td>$h'_0(26)$, $h'_0(37)$</td>
<td>6.5499127E-02</td>
</tr>
<tr>
<td>$h'_0(27)$, $h'_0(36)$</td>
<td>1.9264450E-02</td>
</tr>
<tr>
<td>$h'_0(28)$, $h'_0(35)$</td>
<td>-8.0930449E-02</td>
</tr>
<tr>
<td>$h'_0(29)$, $h'_0(34)$</td>
<td>2.7681533E-02</td>
</tr>
<tr>
<td>$h'_0(30)$, $h'_0(33)$</td>
<td>6.6743910E-02</td>
</tr>
</tbody>
</table>

Transmit and receive bandpass filter coefficients for bandpass filter no. 10; odd symmetry.
Transmit and receive bandpass filter coefficients for bandpass filter no. 11; even symmetry.

<table>
<thead>
<tr>
<th>Coefficients</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>( h(0), h(63) )</td>
<td>(-1.1043159E-03)</td>
</tr>
<tr>
<td>( h'(1), h'(62) )</td>
<td>(2.2086317E-03)</td>
</tr>
<tr>
<td>( h''(2), h''(61) )</td>
<td>(-2.4094165E-04)</td>
</tr>
<tr>
<td>( h''(3), h''(60) )</td>
<td>(-1.5861992E-03)</td>
</tr>
<tr>
<td>( h''(4), h''(59) )</td>
<td>(1.8873763E-03)</td>
</tr>
<tr>
<td>( h''(5), h''(58) )</td>
<td>(-5.4211874E-04)</td>
</tr>
<tr>
<td>( h''(6), h''(57) )</td>
<td>(-1.3653360E-03)</td>
</tr>
<tr>
<td>( h''(7), h''(56) )</td>
<td>(6.2243262E-04)</td>
</tr>
<tr>
<td>( h''(8), h''(55) )</td>
<td>(7.8306039E-04)</td>
</tr>
<tr>
<td>( h''(9), h''(54) )</td>
<td>(1.8271408E-03)</td>
</tr>
<tr>
<td>( h''(10), h''(53) )</td>
<td>(-4.8991470E-03)</td>
</tr>
<tr>
<td>( h''(11), h''(52) )</td>
<td>(1.2850221E-03)</td>
</tr>
<tr>
<td>( h''(12), h''(51) )</td>
<td>(7.9309959E-03)</td>
</tr>
<tr>
<td>( h''(13), h''(50) )</td>
<td>(-1.2187632E-02)</td>
</tr>
<tr>
<td>( h''(14), h''(49) )</td>
<td>(1.3051006E-03)</td>
</tr>
<tr>
<td>( h''(15), h''(48) )</td>
<td>(1.7950153E-02)</td>
</tr>
<tr>
<td>( h''(16), h''(47) )</td>
<td>(-2.1885533E-02)</td>
</tr>
<tr>
<td>( h''(17), h''(46) )</td>
<td>(-1.7669054E-03)</td>
</tr>
<tr>
<td>( h''(18), h''(45) )</td>
<td>(3.1964924E-02)</td>
</tr>
<tr>
<td>( h''(19), h''(44) )</td>
<td>(-3.2085396E-02)</td>
</tr>
<tr>
<td>( h''(20), h''(43) )</td>
<td>(-7.1077785E-03)</td>
</tr>
<tr>
<td>( h''(21), h''(42) )</td>
<td>(4.7967467E-02)</td>
</tr>
<tr>
<td>( h''(22), h''(41) )</td>
<td>(-4.1180942E-02)</td>
</tr>
<tr>
<td>( h''(23), h''(40) )</td>
<td>(-1.5018696E-02)</td>
</tr>
<tr>
<td>( h''(24), h''(39) )</td>
<td>(6.3367650E-02)</td>
</tr>
<tr>
<td>( h''(25), h''(38) )</td>
<td>(-4.6622209E-02)</td>
</tr>
<tr>
<td>( h''(26), h''(37) )</td>
<td>(-2.4636284E-02)</td>
</tr>
<tr>
<td>( h''(27), h''(36) )</td>
<td>(7.5575367E-02)</td>
</tr>
<tr>
<td>( h''(28), h''(35) )</td>
<td>(-4.7144249E-02)</td>
</tr>
<tr>
<td>( h''(29), h''(34) )</td>
<td>(-3.4514893E-02)</td>
</tr>
<tr>
<td>( h''(30), h''(33) )</td>
<td>(8.2241416E-02)</td>
</tr>
<tr>
<td>( h''(31), h''(32) )</td>
<td>(-4.2726986E-02)</td>
</tr>
</tbody>
</table>
Coefficients

<table>
<thead>
<tr>
<th>Coefficients</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>$h_{12}^{2}(0), h_{12}^{2}(63)$</td>
<td>$1.5736382E-03$</td>
</tr>
<tr>
<td>$h_{12}^{2}(1), h_{12}^{2}(62)$</td>
<td>$-9.2115405E-04$</td>
</tr>
<tr>
<td>$h_{12}^{2}(2), h_{12}^{2}(61)$</td>
<td>$-1.3625404E-03$</td>
</tr>
<tr>
<td>$h_{12}^{2}(3), h_{12}^{2}(60)$</td>
<td>$2.5139828E-03$</td>
</tr>
<tr>
<td>$h_{12}^{2}(4), h_{12}^{2}(59)$</td>
<td>$-1.4776846E-03$</td>
</tr>
<tr>
<td>$h_{12}^{2}(5), h_{12}^{2}(58)$</td>
<td>$6.5248413E-04$</td>
</tr>
<tr>
<td>$h_{12}^{2}(6), h_{12}^{2}(57)$</td>
<td>$5.5653055E-04$</td>
</tr>
<tr>
<td>$h_{12}^{2}(7), h_{12}^{2}(56)$</td>
<td>$-6.5248413E-04$</td>
</tr>
<tr>
<td>$h_{12}^{2}(8), h_{12}^{2}(55)$</td>
<td>$-1.7271639E-04$</td>
</tr>
<tr>
<td>$h_{12}^{2}(9), h_{12}^{2}(54)$</td>
<td>$-2.1685502E-03$</td>
</tr>
<tr>
<td>$h_{12}^{2}(10), h_{12}^{2}(53)$</td>
<td>$5.5461149E-03$</td>
</tr>
<tr>
<td>$h_{12}^{2}(11), h_{12}^{2}(52)$</td>
<td>$-5.318263E-03$</td>
</tr>
<tr>
<td>$h_{12}^{2}(12), h_{12}^{2}(51)$</td>
<td>$-8.277261E-04$</td>
</tr>
<tr>
<td>$h_{12}^{2}(13), h_{12}^{2}(50)$</td>
<td>$1.1629570E-02$</td>
</tr>
<tr>
<td>$h_{12}^{2}(14), h_{12}^{2}(49)$</td>
<td>$-1.7329210E-02$</td>
</tr>
<tr>
<td>$h_{12}^{2}(15), h_{12}^{2}(48)$</td>
<td>$9.1155870E-03$</td>
</tr>
<tr>
<td>$h_{12}^{2}(16), h_{12}^{2}(47)$</td>
<td>$1.0900323E-02$</td>
</tr>
<tr>
<td>$h_{12}^{2}(17), h_{12}^{2}(46)$</td>
<td>$-2.9707218E-02$</td>
</tr>
<tr>
<td>$h_{12}^{2}(18), h_{12}^{2}(45)$</td>
<td>$2.9227450E-02$</td>
</tr>
<tr>
<td>$h_{12}^{2}(19), h_{12}^{2}(44)$</td>
<td>$-2.8977972E-03$</td>
</tr>
<tr>
<td>$h_{12}^{2}(20), h_{12}^{2}(43)$</td>
<td>$-3.3430215E-02$</td>
</tr>
<tr>
<td>$h_{12}^{2}(21), h_{12}^{2}(42)$</td>
<td>$5.0778616E-02$</td>
</tr>
<tr>
<td>$h_{12}^{2}(22), h_{12}^{2}(41)$</td>
<td>$-2.9726048E-02$</td>
</tr>
<tr>
<td>$h_{12}^{2}(23), h_{12}^{2}(40)$</td>
<td>$-1.9785622E-02$</td>
</tr>
<tr>
<td>$h_{12}^{2}(24), h_{12}^{2}(39)$</td>
<td>$6.1371895E-02$</td>
</tr>
<tr>
<td>$h_{12}^{2}(25), h_{12}^{2}(38)$</td>
<td>$-5.9875011E-02$</td>
</tr>
<tr>
<td>$h_{12}^{2}(26), h_{12}^{2}(37)$</td>
<td>$1.1092230E-02$</td>
</tr>
<tr>
<td>$h_{12}^{2}(27), h_{12}^{2}(36)$</td>
<td>$5.0970525E-02$</td>
</tr>
<tr>
<td>$h_{12}^{2}(28), h_{12}^{2}(35)$</td>
<td>$-7.8605145E-02$</td>
</tr>
<tr>
<td>$h_{12}^{2}(29), h_{12}^{2}(34)$</td>
<td>$4.8091918E-02$</td>
</tr>
<tr>
<td>$h_{12}^{2}(30), h_{12}^{2}(33)$</td>
<td>$1.9881574E-02$</td>
</tr>
<tr>
<td>$h_{12}^{2}(31), h_{12}^{2}(32)$</td>
<td>$-7.4613474E-02$</td>
</tr>
</tbody>
</table>

Transmit and receive bandpass filter coefficients for bandpass filter no. 12; odd symmetry.
Coefficients

<table>
<thead>
<tr>
<th>Coefficients</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>$h_{13}(0)$, $h_{13}(63)$</td>
<td>2.3488600E-03</td>
</tr>
<tr>
<td>$h_{13}(1)$, $h_{13}(62)$</td>
<td>-3.6882926E-03</td>
</tr>
<tr>
<td>$h_{13}(2)$, $h_{13}(61)$</td>
<td>3.1253425E-03</td>
</tr>
<tr>
<td>$h_{13}(3)$, $h_{13}(60)$</td>
<td>-2.1547393E-03</td>
</tr>
<tr>
<td>$h_{13}(4)$, $h_{13}(59)$</td>
<td>-9.3177916E-04</td>
</tr>
<tr>
<td>$h_{13}(5)$, $h_{13}(58)$</td>
<td>2.1547393E-03</td>
</tr>
<tr>
<td>$h_{13}(6)$, $h_{13}(57)$</td>
<td>-1.6112014E-03</td>
</tr>
<tr>
<td>$h_{13}(7)$, $h_{13}(56)$</td>
<td>3.6882926E-04</td>
</tr>
<tr>
<td>$h_{13}(8)$, $h_{13}(55)$</td>
<td>1.5917894E-03</td>
</tr>
<tr>
<td>$h_{13}(9)$, $h_{13}(54)$</td>
<td>-7.7648263E-05</td>
</tr>
<tr>
<td>$h_{13}(10)$, $h_{13}(53)$</td>
<td>-3.6300563E-03</td>
</tr>
<tr>
<td>$h_{13}(11)$, $h_{13}(52)$</td>
<td>7.5707058E-03</td>
</tr>
<tr>
<td>$h_{13}(12)$, $h_{13}(51)$</td>
<td>-8.6771930E-03</td>
</tr>
<tr>
<td>$h_{13}(13)$, $h_{13}(50)$</td>
<td>2.4653324E-03</td>
</tr>
<tr>
<td>$h_{13}(14)$, $h_{13}(49)$</td>
<td>8.9101382E-03</td>
</tr>
<tr>
<td>$h_{13}(15)$, $h_{13}(48)$</td>
<td>-2.0770911E-02</td>
</tr>
<tr>
<td>$h_{13}(16)$, $h_{13}(47)$</td>
<td>2.5332745E-02</td>
</tr>
<tr>
<td>$h_{13}(17)$, $h_{13}(46)$</td>
<td>-1.6131427E-02</td>
</tr>
<tr>
<td>$h_{13}(18)$, $h_{13}(45)$</td>
<td>-5.2218456E-03</td>
</tr>
<tr>
<td>$h_{13}(19)$, $h_{13}(44)$</td>
<td>3.0515768E-02</td>
</tr>
<tr>
<td>$h_{13}(20)$, $h_{13}(43)$</td>
<td>-4.6181306E-02</td>
</tr>
<tr>
<td>$h_{13}(21)$, $h_{13}(42)$</td>
<td>4.0998284E-02</td>
</tr>
<tr>
<td>$h_{13}(22)$, $h_{13}(41)$</td>
<td>-1.3510798E-02</td>
</tr>
<tr>
<td>$h_{13}(23)$, $h_{13}(40)$</td>
<td>2.6050992E-02</td>
</tr>
<tr>
<td>$h_{13}(24)$, $h_{13}(39)$</td>
<td>5.8857385E-02</td>
</tr>
<tr>
<td>$h_{13}(25)$, $h_{13}(38)$</td>
<td>-6.6971630E-02</td>
</tr>
<tr>
<td>$h_{13}(26)$, $h_{13}(37)$</td>
<td>4.3172434E-02</td>
</tr>
<tr>
<td>$h_{13}(27)$, $h_{13}(36)$</td>
<td>3.8241770E-03</td>
</tr>
<tr>
<td>$h_{13}(28)$, $h_{13}(35)$</td>
<td>-5.2723169E-02</td>
</tr>
<tr>
<td>$h_{13}(29)$, $h_{13}(34)$</td>
<td>7.9511821E-02</td>
</tr>
<tr>
<td>$h_{13}(30)$, $h_{13}(33)$</td>
<td>-7.0038736E-02</td>
</tr>
<tr>
<td>$h_{13}(31)$, $h_{13}(32)$</td>
<td>2.7778666E-02</td>
</tr>
</tbody>
</table>

Transmit and receive bandpass filter coefficients for bandpass filter no. 13; even symmetry.
<table>
<thead>
<tr>
<th>Coefficients</th>
<th>( h(0), h(63) )</th>
<th>( 1.0984605\times10^{-3} )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( h(1), h(62) )</td>
<td>(-1.2494989\times10^{-3} )</td>
<td></td>
</tr>
<tr>
<td>( h(2), h(61) )</td>
<td>( 2.2518442\times10^{-3} )</td>
<td></td>
</tr>
<tr>
<td>( h(3), h(60) )</td>
<td>(-1.8811137\times10^{-3} )</td>
<td></td>
</tr>
<tr>
<td>( h(4), h(59) )</td>
<td>( 8.9249923\times10^{-4} )</td>
<td></td>
</tr>
<tr>
<td>( h(5), h(58) )</td>
<td>( 9.1996073\times10^{-4} )</td>
<td></td>
</tr>
<tr>
<td>( h(6), h(57) )</td>
<td>(-4.4075730\times10^{-3} )</td>
<td></td>
</tr>
<tr>
<td>( h(7), h(56) )</td>
<td>( 7.1399938\times10^{-3} )</td>
<td></td>
</tr>
<tr>
<td>( h(8), h(55) )</td>
<td>(-9.6801836\times10^{-3} )</td>
<td></td>
</tr>
<tr>
<td>( h(9), h(54) )</td>
<td>( 8.9661842\times10^{-3} )</td>
<td></td>
</tr>
<tr>
<td>( h(10), h(53) )</td>
<td>(-4.9293418\times10^{-3} )</td>
<td></td>
</tr>
<tr>
<td>( h(11), h(52) )</td>
<td>(-2.1557289\times10^{-3} )</td>
<td></td>
</tr>
<tr>
<td>( h(12), h(51) )</td>
<td>( 1.1465183\times10^{-3} )</td>
<td></td>
</tr>
<tr>
<td>( h(13), h(50) )</td>
<td>(-1.8920982\times10^{-3} )</td>
<td></td>
</tr>
<tr>
<td>( h(14), h(49) )</td>
<td>( 2.2916634\times10^{-3} )</td>
<td></td>
</tr>
<tr>
<td>( h(15), h(48) )</td>
<td>(-2.0074368\times10^{-3} )</td>
<td></td>
</tr>
<tr>
<td>( h(16), h(47) )</td>
<td>( 1.0270606\times10^{-3} )</td>
<td></td>
</tr>
<tr>
<td>( h(17), h(46) )</td>
<td>( 4.8194956\times10^{-3} )</td>
<td></td>
</tr>
<tr>
<td>( h(18), h(45) )</td>
<td>(-2.1859365\times10^{-3} )</td>
<td></td>
</tr>
<tr>
<td>( h(19), h(44) )</td>
<td>( 3.4848660\times10^{-2} )</td>
<td></td>
</tr>
<tr>
<td>( h(20), h(43) )</td>
<td>(-3.9572041\times10^{-2} )</td>
<td></td>
</tr>
<tr>
<td>( h(21), h(42) )</td>
<td>( 3.2555625\times10^{-2} )</td>
<td></td>
</tr>
<tr>
<td>( h(22), h(41) )</td>
<td>(-1.4678179\times10^{-2} )</td>
<td></td>
</tr>
<tr>
<td>( h(23), h(40) )</td>
<td>(-9.6801836\times10^{-3} )</td>
<td></td>
</tr>
<tr>
<td>( h(24), h(39) )</td>
<td>( 3.3736471\times10^{-2} )</td>
<td></td>
</tr>
<tr>
<td>( h(25), h(38) )</td>
<td>(-4.9746532\times10^{-2} )</td>
<td></td>
</tr>
<tr>
<td>( h(26), h(37) )</td>
<td>( 5.2492686\times10^{-2} )</td>
<td></td>
</tr>
<tr>
<td>( h(27), h(36) )</td>
<td>(-4.0148735\times10^{-2} )</td>
<td></td>
</tr>
<tr>
<td>( h(28), h(35) )</td>
<td>( 1.5735447\times10^{-2} )</td>
<td></td>
</tr>
<tr>
<td>( h(29), h(34) )</td>
<td>( 1.4032833\times10^{-2} )</td>
<td></td>
</tr>
<tr>
<td>( h(30), h(33) )</td>
<td>(-4.0643040\times10^{-2} )</td>
<td></td>
</tr>
<tr>
<td>( h(31), h(32) )</td>
<td>( 5.6241181\times10^{-2} )</td>
<td></td>
</tr>
</tbody>
</table>

Transmit and receive bandpass filter
coefficients for bandpass filter no. 14;
odd symmetry.
### Parallel Filter Coefficients for a 16 Band Sub-Band Coder with Two Point FFT.

**Filter 0 → 7**

<table>
<thead>
<tr>
<th>Filter</th>
<th>Parallel Filter Coefficients</th>
</tr>
</thead>
<tbody>
<tr>
<td>MC 9</td>
<td>0.21326E-02 0.13595E-02 0.80935E-03 0.10525E-02 -0.25101E-02 0.39621E-02 0.37750E-02 -0.42420E-02</td>
</tr>
<tr>
<td>MC 7</td>
<td>0.30542E-02 0.27535E-02 0.37696E-02 0.30949E-02 0.1529E-02 0.73892E-02 0.79375E-02 -0.11060E-02</td>
</tr>
<tr>
<td>MC 6</td>
<td>0.30543E-02 0.26743E-02 0.32527E-02 0.33149E-02 0.22281E-02 0.29418E-02 0.58543E-03 0.13806E-02</td>
</tr>
<tr>
<td>MC 4</td>
<td>0.23414E-02 0.26435E-02 0.4398E-03 0.11869E-02 0.23054E-02 0.19629E-02 0.15576E-02 -0.13860E-02</td>
</tr>
<tr>
<td>MC 2</td>
<td>0.19025E-02 0.23156E-02 0.7376E-03 0.29341E-04 -0.11923E-02 0.18169E-02 -0.12871E-02 0.63119E-03</td>
</tr>
<tr>
<td>MC 0</td>
<td>0.14390E-02 0.19118E-02 0.13769E-02 0.87529E-03 0.38200E-03 0.30020E-03 -0.17401E-02 0.17312E-02</td>
</tr>
<tr>
<td>K 11</td>
<td>0.28225E-03 0.85144E-03 0.44399E-02 -0.5189E-02 -0.30769E-02 0.10632E-02 -0.25748E-02 0.41291E-02</td>
</tr>
<tr>
<td>K 10</td>
<td>0.10515E-02 0.17610E-02 0.52843E-02 -0.48042E-02 -0.18697E-02 -0.36700E-02 -0.50953E-02 0.45231E-02</td>
</tr>
<tr>
<td>K 9</td>
<td>0.21326E-02 0.13595E-02 -0.80935E-03 -0.10525E-02 -0.25101E-02 0.39621E-02 0.37750E-02 -0.42420E-02</td>
</tr>
<tr>
<td>K 8</td>
<td>0.13389E-02 -0.25976E-03 0.20343E-02 -0.36727E-02 -0.50051E-02 0.48086E-02 0.37666E-02 -0.22995E-02</td>
</tr>
<tr>
<td>K 7</td>
<td>0.27100E-02 0.22397E-02 -0.29530E-02 0.1529E-02 0.17320E-02 -0.25209E-03 -0.13777E-02 0.22975E-03</td>
</tr>
<tr>
<td>K 6</td>
<td>0.30542E-02 0.27535E-02 -0.37696E-02 0.30949E-02 0.37891E-02 -0.34371E-02 -0.35317E-02 0.34585E-02</td>
</tr>
<tr>
<td>K 5</td>
<td>0.27667E-02 -0.28630E-02 -0.19362E-02 0.24870E-02 -0.81853E-03 -0.11592E-02 0.24279E-02 -0.19383E-02</td>
</tr>
<tr>
<td>K 4</td>
<td>0.32936E-02 0.29244E-02 -0.32527E-02 0.31349E-02 0.22818E-02 -0.29418E-02 -0.58543E-02 0.13806E-02</td>
</tr>
<tr>
<td>K 3</td>
<td>0.19025E-02 0.23156E-02 0.7376E-03 0.29341E-04 -0.11923E-02 0.18169E-02 -0.12871E-02 0.63119E-03</td>
</tr>
<tr>
<td>K 2</td>
<td>0.14390E-02 0.19118E-02 0.13769E-02 0.87529E-03 0.38200E-03 0.30020E-03 -0.17401E-02 0.17312E-02</td>
</tr>
<tr>
<td>K 1</td>
<td>0.10117E-02 -0.15552E-02 0.14627E-02 -0.12629E-02 0.12951E-02 -0.10206E-02 0.52157E-03 0.18289E-03</td>
</tr>
<tr>
<td>K 0</td>
<td>0.70329E-03 -0.1309E-02 0.10475E-02 -0.12221E-02 0.96622E-03 -0.13118E-02 0.15048E-02 -0.12806E-02</td>
</tr>
</tbody>
</table>

**APPENDIX B**
APPENDIX C

LIST OF PUBLISHED PAPERS


1. Introduction:

Digital speech coding has thus far been polarised between the high rate (64,32 kbit/s) level telecommunications network requirements and the military low rate (2.4 kbit/s & below) systems. In the former PCM using A and μ law companding at 64kbit/s and ADPCM at 32 kbit/s have both been standardised by the CCITT. For military systems LPC-10 seems to have been the standard adopted at 2.4kbit/s. The two areas have led respectively to waveform and vocoder solutions. Recently, interest amongst the telecommunications fraternity has been in 16kbit/s for; (i) the Pan-European digital (GSM) cellular mobile radio system and (ii) a digital replacement for compacted f.m. in the INMARSAT Satellite System. It is understood that the former has standardised on RELP and the latter on the UFC coding systems. Both schemes represent the merger between pure waveform & vocoding techniques which will be continued in the search for economic coding schemes at 9.6,4.8,2.4 kbit/s & below. At Surrey we have been especially interested in the range 16-4.8kbit/s and much of the work has been driven by the future requirements of mobile satellite systems (maritime, aeronautical & especially land-mobile). However the results of our work are equally applicable to other mobile systems requiring a coding scheme which is robust to the channel conditions, and yet giving good speech quality for an economically implementable algorithm.

We present a review of the special coding techniques that have been considered in the 9.6-4.8kbit/s region with their inherent advantages and disadvantages for use in mobile systems. In addition we will present new schemes which are showing great promise in terms of speech quality at 4.8kbit/s. Finally we address the extension of some of these schemes down to the 2.4kbit/s region. We will demonstrate a new coded-baseband CELP (CELP-CB) system which is implementable and compares very favourably with LPC-10.

2. Algorithms for 9.6 to 4.8 kb/s speech coding

There are several algorithms which can be used to code speech at rates of 9.6 to 4.8 kb/s. Although they are given different names, the principles on which they are based are very similar. They all estimate and remove the correlation in speech and then quantize and transmit the remaining (residual) signal. At the receiver, the removed correlation is introduced into the received residual with the help of model parameters, which are also transmitted by the encoder.

Estimation of speech parameters both in the time and frequency domain are well developed and currently in use. Therefore, to improve the coded speech quality, most of the recent research work has been concentrated on finding the best possible estimate of the residual signal. It may be possible to divide these coders into two groups in terms of the way in which they operate on the residual signal, i.e. the Analysis and Synthesis Systems and the Analysis by Synthesis Systems.
Analysis and Synthesis Systems include sub-band coders, adaptive transform coders, baseband coders etc. These obtain the residual by an analysis procedure and then directly quantize and transmit this residual. During the quantization process the error between the residual and its quantized value is minimized. Hence the quality of the synthesized speech is very dependent on the accuracy of quantization of the residual signal. These coders can produce high quality speech at bit rates as low as 8 kbit/s with moderate complexity. Their quality deteriorates rapidly as the bit rate approaches 4.8 kbit/s.

Analysis by synthesis systems on the other hand aim to replace the residual signal by a sequence of pulses which minimize the error between the original and the synthetic speech. Analysis by synthesis systems, also known as stochastic coders include Multi-Pulse excited linear prediction (MP-LPC) and code excited linear prediction (CELP), which are capable of producing high quality speech at bit rates as low as 4.8 kbit/s. Although these coders are very attractive because of their high quality, they are very complex and cannot be implemented using current technology without considerable simplification.

In the following sections we briefly explain the basic principles of the most promising coders in the bit rate range 9.6 to 4.8 kbit/s. We then report on a new CELP coded baseband coder (CELP-BB) which can operate at rates below 4.8 kbit/s.

2.1 Sub-band/Transform Coders

The principles of sub-band and adaptive transform coders are based upon the non-flat spectral characteristics of speech and are referred to as frequency domain coders (1). The central feature of the frequency domain coders is the splitting of the speech spectrum into narrow frequency bands using filter banks or block transforms. These frequency bands are then treated separately. In this way the non-flat spectrum of speech is exploited, offering three major advantages.

- Quantization noise can be contained within frequency bands to prevent masking of one frequency range by noise in another.
- Available bits can be distributed according to the energy distribution of the frequency bands.
- Overall quantization noise spectrum can be controlled during the bit allocation process.

To ensure that the receiver is informed about the bit allocation and to adapt its APCM decoder step size, band energies are also quantized and transmitted. Sub-band and transform coders are capable of producing high quality speech at 8 kbit/s with moderate complexities.

2.2 Baseband Coder

Baseband coders can be considered as the second generation of adaptive prediction coders (2). Here, a small portion of the residual signal (after LPC inverse filtering) is quantized and transmitted together with the LPC parameters. At the receiver the remainder of the residual signal is regenerated from the baseband which should contain the fundamental frequency, or at least its two adjacent harmonics. A compromise is made between the quantization noise due to the actual quantization of the baseband and the noise due to high frequency regeneration. The reported speech quality of these decoders is good at bit rates around 9.6 kbit/s. At Surrey we have recently introduced an improved baseband coder which is capable of producing high-quality speech at 9.6 kbit/s. This coder uses Discrete Cosine Transforms (DCT) to obtain pitch harmonics. A simple pitch model is then used to allocate bits and to remove the peaks prior to quantization. This
enables more accurate quantization and hence a larger baseband is transmitted, for a given bit rate. The quantized baseband spectrum and the rest of the spectrum are then cross-correlated to find the best matching point for high frequency regeneration in the frequency domain.

2.3 Stochastic Coders

Stochastic coders aim to overcome the limitation of vocoders by replacing the existing excitation source by a sequence of pulses which are optimized either one by one for MPLPC or as a block for CELP (3). The procedure of finding the best sequence of excitation pulses (replacement for residual signal) is long and computationally very heavy. Therefore, although, these coders can produce high quality speech at rates as low as 4.8 kbit/s, it is essential to simplify them for real time implementation.

The reason that these coders produce relatively high quality is that the memory of the IIR pitch and LPC filters carried over from the previous blocks can produce a good estimate of the current block of samples when clocked through the current speech parameters. Before the search for an optimum excitation sequence, this estimate is subtracted from the original speech and the difference signal is matched with the signal produced by excitation sequences. This allows a coarse vector quantization of the residual signal.

Most of the computation is due to the convolution processes of the synthesis filters during the search for an optimum excitation sequence. Therefore, to reduce complexity search procedures for an optimum sequence must be simplified. Searches can be simplified if done in the frequency domain (4), because convolution processes can be replaced by multiplication. Frequency domain searches can be further simplified by restricting the search to a subset of frequencies. Speech synthesis and memory subtraction should still be performed in the time domain.

3. Speech Coding at rates below 4.8 kbit/s

In a typical CELP coder 2kbit/s of the total 4.8 kbit/s capacity is used for the transmission of the excitation sequences, and the rest of the capacity is used for the transmission of the speech model parameters and in achieving the optimum gain for each block. Therefore, to bring the total bit rate below 3 kbit/s both the capacity for excitation sequences and for speech parameters should be reduced. To achieve both goals and still maintain a reasonable quality we suggest a new approach which we denote as the CELP coded baseband coder (CELP-BB).

The structure of CELP-BB comprises of two major processing stages. The first stage removes the short time correlation using Vector Quantized LPC parameters, and then produces the downsamplied baseband residual. The second processing stage Vector Quantizes the baseband residual using an analysis by synthesis process around a strong pitch filter. In the decoder, pitch is introduced into the baseband residual and then spectral folding is used to recover the output speech.

4. Discussion

From the discussions above it is now clear that future generation coders for low and very low bit rates will be based on the principles of the stochastic coder. In stochastic coders, memory response of the pitch and LPC filters
allows the residual to be coarse quantized. This means that provided the parameters of the speech model are protected, these coders can operate under very severe channel conditions. Along with the development of stochastic coders, digital signal processing (DSP) technology is expanding rapidly making stochastic coders implementable. Therefore we feel confident that speech coding research can now move down below 4.8 kbit/s rates with good possibilities for the production of good-quality coders that can be implemented in available, or predicted, DSP technology.

(1) R. E. Crochiere, J.M. Tribolet
"Frequency Domain Coding of Speech"

(2) J. Makhoul M. Berouti
"Predictive and residual encoding of Speech"
Journal of Acoustic Society of America pp 1633-1641 1979

(3) B.S. Atal
"High quality speech at low bit rates: Multi-pulse and stochastically excited linear predictive coders"

(4) I.M. Trancoso B. S. Atal
"Efficient Procedure for finding the optimum innovation in Stochastic Coders"
ICASSP-86 pp 2375-2378 Tokyo 1986
HYBRID TRANSFORM CODER FOR LOW BIT RATE SPEECH CODING

A.Kondoz*, Prof.B.G.Evans*

ABSTRACT

Frequency domain speech coding techniques such as sub-band coder (SBC) (ref 1) and adaptive transform coder (ATC) (ref 2) can produce high quality digital speech at around 16 Kb/s. However, at bit rates below 16 Kb/s, their rapidly deteriorating speech quality and increasing complexity make them less competitive to the time domain coders such as residual excited linear prediction (RELP) (ref 3). In this study a hybrid of ideas from ATC, RELP, and vector quantization (VQ) are put together in order to improve the quality of a low bit rate transform coder. Informal listening tests have shown that the proposed coder out-performs the speech specific ATC (SSATC) (ref 2) at 9.6 Kb/s as well as SBC (ref 1) and RELP (ref 3).

INTRODUCTION

In recent years frequency domain speech coding has been shown to be efficient at bit rates as low as 16 Kb/s. SSATC has been proposed for lower bit rates than 16 Kb/s. Although SSATC can produce high quality speech at around 9.6 Kb/s, its very high complexity is a big disadvantage. The two main types of distortions seen in the low bit rate transform coders are the band-limitation effect and occasional clicks. These are simply the result of inefficient representation of the residual signal. These distortions are eliminated in the proposed coder by three improvements, e.g. reduced side information, improved quantization, and use of high frequency regeneration (HFR). In the following sections these three topics will be discussed.

SYSTEM DESCRIPTION

A block diagram of the coder under investigation is shown in figure 1. First, speech is analysed to measure the pitch period and to calculate 10 linear prediction coefficients (LPC). LPC parameters are vector quantized using a 10 bit codebook and then used to remove the short time correlation of speech. The remaining residual signal is then frequency transformed by a 256 point discrete cosine transform (DCT). To remove the pitch harmonics of the residual spectrum, original coefficients are divided by their estimated values. Coefficients in the lower frequency part of the spectrum (0 -> 1.5 KHz) are then quantized and transmitted. Quantized coefficients are also cross-correlated with the high frequency coefficients to find the best possible matching points for high frequency regeneration. With this information at the decoder a full-band residual spectrum is obtained by shifting the received coefficients to the calculated frequency points. Full-band coefficients are then inverse DCT transformed and filtered through the LPC synthesis filter to recover the output speech.

1.0 QUANTIZATION OF LPC PARAMETERS

In the proposed coder most of the side information is due to transmission of the LPC parameters. To reduce this by about a factor of 4, a 10 bit full search codebook is used to vector quantize the LPC parameters. The distortion measure employed is a likelihood ratio measure, which is the gain normalized version of the Itakura-Saito measure (ref 4) and is defined as,

\[ d \left( 1/Am : 1/A \right) = \frac{R_x(0)R_a(0)}{E_m} + 2 \sum_{i=1}^{M} \frac{[R_x(i)R_a(i)]}{E_m} - 1 \] (1)

* Dept. of Electronic and Electrical Engineering, University of Surrey, Guildford, Surrey, GU2 5XH
where $Rx(i)$ and $Ra(i)$ are the autocorrelation sequences of the input speech and of the LPC coefficients, respectively. $1/Am$ is the optimal filter for the current block of speech. $Em$ is the minimum residual energy. $M$ is the order of the filter.

The usual procedure to design a codebook is to use the LBG algorithm (ref 6), where an initial vector is split into two and the resulting two vectors into four and so on until the overall distortion is acceptable. After each stage of splitting the resulting codebook is optimized by iterative centroid calculation. Therefore, LBG algorithm requires a lot of computations. During the splitting like a tree, sum branches may move into spaces where there are only a few training vectors. This is a disadvantage, because the contribution of those least used codebook entries to the total signal to distortion ratio will be very small. To both decrease the number of computations required and make sure all the vectors in the codebook are useful, we started with a randomly chosen codebook. The size of the codebook was larger than expected final codebook. This codebook was then reduced in size simply by discarding the least used vectors until the distortion was bigger than desired limit. This final codebook was then optimized by iterative centroid calculation.

2.0 QUANTIZATION OF THE RESIDUAL FREQUENCY COEFFICIENTS

In order to eliminate band limitation effects it was decided to transmit a part of the spectrum and use high frequency regeneration to obtain the remainder. Although this does eliminate the band limitation effects, it also introduces high frequency distortion which increases with decrease in the transmitted bandwidth. To transmit the largest possible bandwidth without noticeable clicks or quantization noise, two quantization algorithms were investigated. They are scalar block quantization with dynamic bit allocation and vector quantization.

2.1 Scalar Block Quantization

Prior to quantization an estimate of the spectrum is used to both remove the pitch harmonics and allocate bits. To create a frequency domain estimate (pitch pattern) the model given in (ref 2) is used:

$$c(p) = \frac{11}{(i - Ge^{-j \omega p})}$$

(2)

where $c_p(\omega)$ represents the magnitude of the estimated frequency coefficients. $p$ is the pitch period and $G$ is the pitch gain.

In our simulations it was found that computing $G$ for every block was not necessary. In fact it causes problems for extreme values of $G$. This was due to the estimated levels between the pitch harmonics being very small or very large, thus, causing the quantized signal to have large errors. After several experiments it was found that a constant pitch gain of 0.65 produced the best results.

In order to maximize the signal to quantization noise ratio and hence transmit the largest possible bandwidth for a given bit rate, bits were allocated according to the variations in the coefficients amplitudes. Using the estimated coefficients, bits were allocated in two steps (ref 1):

i) find the largest amplitude, allocate one bit to it and divide by 2,

ii) check if all bits are allocated, stop else go back to (i).

To avoid cases in which all bits are allocated to the pitch harmonics and no bits to the coefficients in between the pitch harmonics, it was found necessary to limit the maximum and minimum bits per coefficient to 4 and 2, respectively. This produced 13.87 dB signal to quantization noise ratio (SNR) as against 12.30 dB for the fixed bit allocation (measured using voiced and unvoiced data together), thus enabling us to quantize a 1.5 KHz baseband. Although, the difference between the fixed and variable bit allocation case is only 1.57 dB the improvement in subjective quality was much greater.

2.2 Vector Quantization

In addition to vector quantizing the LPC parameters, VQ was also used to quantize the residual frequency coefficients. During the VQ of residual it was assumed that the signal,
after removal of pitch harmonics, had a Gaussian distribution. Therefore, random Gaussian codebooks were used for quantization. During the quantization process, 7 and 8 bit codebooks, with vector dimensions of 4 and 5 were tried. The distortion measure employed was the mean squared error. Two types of search procedures, and optimum scale calculations were tested as follows:

DIRECT ERROR MINIMIZATION (DEM): Here the error between the normalized residual vectors and the unit variance codebook vectors was minimized, and the optimum scale factor was taken to be unity (at the decoder the chosen vector was scaled up by the residual energy).

\[ E_n = \sum_{i=1}^{N} [x(i) - \hat{x}(i)]^2 \]  \hspace{1cm} (3)

where \( E_n \) is the squared error, \( x(i) \) and \( \hat{x}(i) \) are the original and quantized residual vectors. \( N \) is the vector dimension.

SYNTHESIZED ERROR MINIMIZATION (SEM): Letting \( P(i) \) and \( S(i) \) be the estimated and the original coefficients and \( V_n(i) \) be the \( i^{th} \) element of the unit variance \( n^{th} \) vector of the codebook; \( a_n \) be the optimum scale factor for the \( n^{th} \) sequence in the codebook.

\[ E_n = \sum_{i=1}^{N} [S(i) - a_n P(i)V_n(i)]^2 \]  \hspace{1cm} (4)

By setting \( \partial E_n / \partial a_n = 0 \)

\[ a_n = \frac{\sum_{i=1}^{N} [S(i)P(i)V_n(i)]}{\sum_{i=1}^{N} [P(i)V_n(i)]^2} \]  \hspace{1cm} (5)

and,

\[ E_n = \sum_{i=1}^{N} S(i)^2 - \left[ \sum_{i=1}^{N} S(i)P(i)V_n(i) \right]^2 / \sum_{i=1}^{N} [P(i)V_n(i)]^2 \]  \hspace{1cm} (6)

The sequence which maximizes the second term in equation (6) is chosen as the quantized vector. The performance of the two types of error minimization with respect to the size of codebooks and vector dimensions are given in table I.

<table>
<thead>
<tr>
<th>Number of bits</th>
<th>Vector dimension</th>
<th>DEM (dB)</th>
<th>SEM (dB)</th>
</tr>
</thead>
<tbody>
<tr>
<td>8</td>
<td>5</td>
<td>9.09</td>
<td>12.23</td>
</tr>
<tr>
<td>8</td>
<td>4</td>
<td>11.99</td>
<td>15.54</td>
</tr>
<tr>
<td>7</td>
<td>4</td>
<td>10.23</td>
<td>13.73</td>
</tr>
</tbody>
</table>

Table I: VQ performance for residual quantization.

Results tabulated in table I show that SEM has about 3.5 dB better performance than DEM, which was also true in subjective quality comparison. The reasons for this are three fold.

First reason is that the difference between the optimum scale (gain) calculation for each vector. For DEM, decoded vectors are scaled by the energy in the original vector, which means even if the noise power is bigger than the signal power we still transmit. However, in the SEM case, scale calculation takes into account the expected noise power (see equation 5).

The second reason is that in DEM the error is amplified by the pitch filter. However, for SEM this is not so because the error is minimized after the pitch filter response.
Finally, in SEM the scale factor can have either negative or positive sign, which means the size of the codebook is virtually doubled with no extra storage or computation for search. In DEM scale is always positive.

3.0 HIGH FREQUENCY REGENERATION

The principles of the HFR used are explained in (ref 5). At the encoder, quantized coefficients are cross-correlated with the unquantized coefficients in the 1.5 to 3.0 KHz region. In the cross-correlation process three shifts on either side of the 1.5 KHz point are calculated. Location of the maximum in magnitude and its sign bit are transmitted. At the decoder this information is used to shift the received coefficients to optimum locations. The region from 3.0 to 4.0 KHz is simply filled by translating the coefficients from 0.5 to 1.5 KHz.

CONCLUSIONS

In this paper we have described ways of improving the quality of a low bit rate transform coder. Informal listening tests confirmed that the proposed coder had better quality than SSATC, RELP and SBC, whilst maintaining relatively lower complexity. In this paper we have also shown that the synthetic error minimization achieves higher SNR performance than the direct matching process.

REFERENCES


Figure 1. Block diagram of Hybrid Transform Coder.
VECTOR QUANTIZED TRANSFORM CODER FOR SPEECH CODING AT 9.6 KB/S AND BELOW

A.Kondoz*, B.G.Evans*.

ABSTRACT

The transform approach to speech coding has been established for some time and has been shown to be very efficient in controlling the bit allocation and the shape of the noise spectrum[1][2]. Various transform coders have been reported which produce high quality digital speech at around 16 Kb/s [1][2]. Although, these coders can maintain good quality down to about 9.6 Kb/s, they perform poorly at the lower bit rates. Here, we shall discuss how Vector Quantization (VQ) can be used to improve the quality of transform coders. We describe one specific design of Vector Quantized Transform Coder (VQTC) which follows on from the work reported in [3] and is capable of producing good quality speech at as low as 4.8 Kb/s.

INTRODUCTION

In recent years frequency domain speech coding has been shown to be efficient at bit rates as low as 16 Kb/s. Hybrid Transform Coding (HTC) [3] has been proposed for lower bit rates than 16 Kb/s. Although HTC can produce high quality speech at around 9.6 Kb/s, its performance reduces rapidly at lower bit rates. The two main types of distortions seen in the low bit rate transform coders are the band-limitation effect and occasional clicks. These are simply the result of inefficient representation of the residual signal. These distortion are eliminated in the proposed coder by three improvements, e.g. reduced side information, improved quantization, and use of noise shaping. In the following sections these three topics will be discussed.

SYSTEM DESCRIPTION

A block diagram of the coder under investigation is shown in figure 1. First, speech is analysed to measure the pitch period and to calculate 10 linear prediction coefficients (LPC). LPC parameters are vector quantized using a 10 bit code-book and then used to remove the short time correlation of speech, which is then further inverse filtered by a single tap pitch filter. The remaining residual signal is then frequency transformed by a 32 point discrete cosine transform (DCT). Using a suitable filter bank (8 BANDS) the envelope of the current block of speech is obtained. (This is the Sub-Band coder approach, FFT can also be used on the impulse response of the LPC filter to obtain envelope estimate). Each vector of dimension 32 is then coded by minimizing the weighted distance from a unit variance Gaussian code-book. Memory response of the synthesis filters clocked

* Dept. of Electronic and Electrical Engineering, University of Surrey, Guildford, Surrey, GU2 5XH
with zero input is then subtracted from the original signal to produce the difference signal to be matched, as used in CELP [4]. The single vector which was selected earlier is then used to produce the output signal which is compared with the reference in order to calculate the optimum gain. The resultant decoder is fairly simple, chosen sequence IDCT transformed to get the time domain equivalent which is then scaled up and inserted into the synthesis filters to recover the output speech.

1.0 Quantization of LPC parameters

In the proposed coder most of the side information is due to the transmission of the LPC parameters. In order to reduce this by about a factor of 4, a 10 bit full search code-book is used to vector quantize the LPC parameters. The distortion measure employed is a likelihood ratio measure, which is the gain normalized version of the Itakura-Saito measure [5] and is defined as,

\[ d \left( \frac{1}{A_{m}}; \frac{1}{A} \right) = \frac{R_x(0)R_a(0)}{E_m} + 2 \sum_{i=1}^{M} \frac{R_x(i)R_a(i)}{E_m} - 1 \]  

where \( R_x(i) \) and \( R_a(i) \) are the autocorrelation values of the speech signal and filter parameters in the code-book, and \( E_m \) is the minimum residual energy obtained using unquantized LPC parameters.

2.0 Vector quantization and noise shaping of the residual vectors

In recently developed coders for use at 8 Kb/s and below the residual signal is quantized in an analysis by synthesis procedure which is very complex [4]. The reason for the analysis by synthesis coding is to consider the effect of the LPC synthesis filter on the coded residual in terms of its filter gain and spectral shape. This makes it possible to compare the original speech and the coded speech rather than comparing the residual vectors with the code-book entries. In this way the effect of the synthesis filter memory is considered in the error minimization.

Here we have overcome these complexities by considering the spectral shape of the LPC filter whilst error minimizing and considering the memory and the gain of the synthesis filters whilst calculating the amplitude scale factor of the chosen vector. Errors are minimized between the residual vectors and code-book entries as:

\[ ER_{(\text{min})} = \sum_{i=1}^{N} \left[ (X(i) - V(i)) W(i) \right]^2 \]  

where \( X(i) \) and \( V(i) \) are the unit variance residual and code-book vectors respectively, and \( W(i) \) is the noise shaping vector (spectral envelope). As can be seen from equation (2) all convolution processes needed in the search of CELP are turned into multiplications. After this very much simplified search, optimum amplitudes are calculated as follows: First the memory of the synthesis filter is subtracted from the original speech, as this cannot be changed. Then the chosen code-book sequence is used to produce the output response, which is then compared with the memory subtracted original as,

\[ a_n = \sum_{i=1}^{N} \left[ S(i) P(i) \right] \left/ \frac{1}{N} \sum_{i=1}^{N} \left[ P(i) \right]^2 \right. \]
where $a_n$ is the amplitude scale, $S(i)$ and $P(i)$ are the reference vector and the output response vector produced by the selected unit variance sequence.

There are two other advantages of calculating the optimum scale factor as opposed to using the original vector energy as the scale (gain).

The first reason is that if the residual energy is taken as the optimum scale value and if the noise power is larger than the signal power we still transmit. However, in the case of optimum scalar calculation, scale calculation takes into account the expected noise power (see equation 3). If the two vectors do not match the numerator of equation (3) will be zero making the scale zero. Therefore no transmission at all is better than transmitting such a vector.

The second reason is that, in equation (3) the scale factor can have either negative or positive sign, which means that the size of the code-book is virtually doubled with no extra storage or computation for searching. When the residual energy is used as the scalar it is always positive which means that the sign of the vector elements cannot be inverted to form another vector.

One very important point which should be considered is the effect of the pitch filter when the pitch period is smaller than the vector dimensions chosen. When the pitch period is smaller than the vector size, quantization noise affects the prediction of the vector elements which are beyond the pitch period. This causes unpleasant prediction error, which can be eliminated by limiting the minimum pitch period to be equal to the vector size. This has been shown to have no affect on the pitch filter performance.

3.0 Subjective Results

The proposed coder has been tested subjectively to evaluate its performance. In the testing procedure the proposed coder was compared with 5 other coders all operating around 7.2 Kb/s. These coders were Hybrid Transform Coder (HTC) [3], Adaptive Transform Coder (ATC) [1], Vocoder Driven ATC (VDATC) [2], Sub-band Coder (SBC) [6] and the proposed coder without optimum scale calculation (scale is taken to be the energy in the input residual vector) (VQTC2). Processed speech for two male and two female sentences for each of the coders were presented to 29 subjects. Each subject was then asked to rank the coders in a preference order. Results are tabulated in Table 1.

<table>
<thead>
<tr>
<th>Coder type</th>
<th>% Preference</th>
</tr>
</thead>
<tbody>
<tr>
<td>VQTC</td>
<td>65</td>
</tr>
<tr>
<td>HTC</td>
<td>20</td>
</tr>
<tr>
<td>VDATC</td>
<td>7</td>
</tr>
<tr>
<td>ATC</td>
<td>2</td>
</tr>
<tr>
<td>SBC</td>
<td>6</td>
</tr>
<tr>
<td>VQTC2</td>
<td>0</td>
</tr>
</tbody>
</table>

Table 1: Subjective preference results.
Results tabulated in Table 1, shows that VQTC outperforms all of the above listed low-bit rate speech coders. Its quality at around 7 Kb/s is comparable to the original speech quality. VQTC was also simulated at 4.8 Kb/s to compare it with the highly complex Code Excited Linear Prediction (CELP) [4].

4.0 Discussions

Herein we have outlined the design principles of a VQTC to produce good quality speech below 9.6 Kb/s with implementable complexity. VQTC has very high quality at around 7 Kb/s and maintains its good quality at even lower bit rates. In order to compare the speech quality of VQTC and the highly complex CELP it was simulated at 4.8 Kb/s. Although, VQTC is very much simpler than CELP its speech quality was almost comparable to the standard full search CELP. VQTC is flexible in terms of complexity which suits single chip applications. For example a reduced complexity VQTC with fixed $W(i)$ which eliminates the need for filter bank or FFT calculation can still produce good quality. Complexity can be reduced further with a little loss in quality by eliminating the noise shaping completely. This results in a coder where the linear predictive residual is coded directly in the way described above. This solution may be used for bit rates around 9.6 Kb/s to enhance the quality of existing predictive coders.

REFERENCES

Figure 1: Block diagram of vector quantized transform coder.
Recently CELP [1] has proved that it is possible to use vector dimensions as large as 40 or more samples long and still maintain high quality. This is achieved by error minimization between the original and the synthesized vectors, rather than error minimization between the residual vectors and code-book entries used in earlier VQ designs. The CELP design however, is very complex and its quality is affected very much by the quantization error of the LPC parameters [2] making it difficult to produce high quality speech at 4.8 Kb/s.

Here, we present a new vector quantized and easily implementable base-band coder (CELP-BB) which was originally proposed for speech coding below 4.8 Kb/s in [3]. Because most of the computation is performed on the base-band which is decimated, and that during code-book search only the pitch filter response is considered, the complexity of CELP-BB is very much less than CELP and its speech quality is not affected by LPC quantization error to such an extent as in CELP. CELP-BB can produce high quality speech (very similar to the original) down to 4.8 Kbps and although its speech quality starts to deteriorate below 4 Kb/s it is capable of producing natural and intelligible digital speech at 2.4 Kb/s.

1.0 INTRODUCTION

In order to offer the possibility of carrying digital speech over a single analogue voice channel, it is necessary to bring the high quality speech coding rate down to about 4.8 Kb/s. CELP coding of speech seemed to be producing high quality speech at bit rates as low as 6 Kb/s. Below 6 Kb/s however, although it produces intelligible speech the amount of quantization noise and roughness makes it unacceptable for use in any public telephone network. The major causes of this undesirable distortion are the spectral distortions caused by LPC parameter quantization and inefficient representation of the excitation sequences. Accuracy of excitation sequences is reduced by the lower update rate of the pitch filter parameters and larger vector dimensions. The other disadvantage of CELP is its high complexity. The vector Quantized Transform Coder (VQTC) [4] has similar performance to CELP and yet has a possible single chip implementable complexity. Other simplified versions of CELP have been reported in the literature which makes it possible to have single chip implementation [5][6]. Therefore, it seems that the complexity problem of CELP can be solved. However, the quality improvement at around 4.8 Kb/s remains to be solved.

Here, we present a new CELP base-band coding (CELP-BB) to improve the speech quality at around 4.8 Kb/s. As the name suggests, CELP coding is applied to the base-band residual to reduce the bit rate of the base-band coder down to 4.8 Kb/s and below.

In base-band coding the LPC residual is first low-pass filtered and decimated before coding. The decimation factor is chosen to transmit the largest base-band for a given overall
bit rate. At the decoder, received base-band is up sampled by inserting zero-valued samples after each sample and then filtered through the LPC synthesis filter. The LPC synthesis filter interpolates the zero valued samples to produce continuous good quality output speech. There are two major causes for speech quality degradation in base-band coding. First, the base-band quantization noise and second, the high frequency regeneration (HFR) noise. These base-band quantization and HFR noises are dependent on the decimation factor and hence the overall bit rate. Both the base-band quantization noise and HFR noise can be reduced using a pitch filter before low-pass filtering and decimation [7]. Also replacing the low-pass filter with a suitable smoothing filter and selecting the maximum energy sequence in the decimation process [8] approximates the base-band coding to Multi-Pulse LPC coding [9] and eliminates the HFR noise completely.

In the following sections we discuss the coder design of CELP-BB and the results obtained from 9.6 Kb/s to 2.4 Kb/s. In the final section we discuss the results of CELP-BB and compare it with CELP.

2.0 CELP-BB CODER DESCRIPTION.

A block diagram of CELP-BB is shown in Figure 1. Then estimated 10 LPC parameters are scalar quantized using their log area ratios and then used to inverse filter the input speech in order to obtain the LPC residual signal. The LPC residual signal is than divided into sub-blocks, each of which is filtered by the weighting filter (smoothing filter) separately. Filtered sub-blocks are split into a number of sequences equal to the decimation factor. These sequences are compared in terms of their energies and the one with the highest energy is selected for transmission [8]. The position of the selected sequence in each sub-block is also transmitted to the decoder in order to place the sequence in the correct location in HFR. The selected sequences are then stored side by side in a buffer to form a decimated continuous signal. Here, VQ is applied to code the continuous decimated base-band residual. The principle of the VQ is based on CELP which works in the following way. The decimated signal is analysed to obtain its pitch period and pitch coefficient. Using these parameters in a pitch synthesis filter the memory response of the filter is computed and subtracted from the decimated signal to form the reference signal. Gaussian code-book sequences are then searched one by one to match the output response of the pitch synthesis filter with no memory, to the reference signal. The index of the optimum sequence, together with its scale value are transmitted to the decoder. At the decoder, selected code-book sequences are scaled up by their scale factors and passed through the pitch synthesis filter in order to recover the continuous decimated base-band signal. The recovered signal is than sub-segmented and shifted to the correct positions with zeros inserted in between the samples, to form the LPC filter excitation sequence. Using this sequence the LPC synthesis filter is excited to recover the output speech.

2.1 Vector Quantization Of the Base-Band

In order to have an integer number of pulses in each sequence, the LPC residual is divided into a number of samples, which is an integer multiple of the decimation factor. Weighted sub-blocks are split into a number of sequences, equal to the decimation factor \( d \) as:

\[
S_j(i) = W(id + j) \quad i=0,1,...,\frac{N}{d}-1, \quad j=1,2,...,d
\]  

(1)

where \( S_j(i) \) is the \( j^{th} \) sequence, \( W(k) \) is the weighted sub-block and \( N \) is the number of samples in each sub-block. The energies of each sequence are calculated as.
The sequence with maximum \( \sigma_j^2 \) is selected for coding and transmission. In each frame all of the selected sequences are placed one after the other to form a continuous signal \( Y(n) \), which contains \( d \) times less samples than the original LPC residual signal \( X(n) \). This means that the upper and lower limits of the expected pitch period are reduced by a factor of \( d \). The continuous decimated signal is then used as the input to an analysis by synthesis Vector Quantizer, or CELP coder as shown in Figure 2. The input to the CELP coder in this case does not contain short time correlation. It does however possess much stronger long-term correlation. Therefore, in the CELP coder both the LPC synthesis filter and the noise shaping filter are excluded, leaving only the pitch synthesis filter. Analysis by synthesis procedure operating around the pitch synthesis filter vector quantizes the decimated signal. The dimension of the code-book sequences are set to be equal to the number of samples in each decimated sequence. This is not of course a restriction. The error is minimized by maximizing the following [10]:

\[
\sigma_j^2 = \sum_{i=1}^{N/d} S_j^2(i)
\]

(2)

and the optimum scale \( \alpha \) is calculated as,

\[
V_{opt}(i) = \frac{\sum_{i=1}^{N/d} S(i) (V_n(i) \ast f_n(i))^2}{\sum_{i=1}^{N/d} [V_n(i) \ast f_n(i)]^2}
\]

(3)

where \( V_n(i) \) is the code-book sequence, \( f_n(i) \) is the truncated pitch filter response and \( \ast \) denotes the convolution process.

\[
V_n(i) \ast f_n(i) \text{ is the response at the output of the pitch synthesis filter caused by the } n^{th} \text{ vector in the code-book. When the pitch period, or the delay in the pitch filter, is greater than } N/d \text{ the truncated response of the pitch filter has a value of 1 in the first location and zeros anywhere else. This makes,}
\]

\[
V_n(i) = V_n(i) \ast f_n(i)
\]

(5)

Therefore, by limiting the minimum pitch period to be equal to the vector size, the convolution process in equations (3) and (4) can be eliminated, which simplifies equations (3) and (4) as follows.

\[
V_{opt}(i) = \frac{\sum_{i=1}^{N/d} S(i) V_n(i)^2}{\sum_{i=1}^{N/d} [V_n(i)]^2}
\]

(6)

and,

\[
\alpha_n = \frac{\sum_{i=1}^{N/d} S(i) V_n(i)}{\sum_{i=1}^{N/d} [V_n(i)]^2}
\]

(7)

respectively.

If the pitch filter parameters are updated for every residual vector then limiting the minimum pitch period to the base-band residual vector size provides the possibility of
considering the decoded samples in the pitch synthesis filter memory while calculating
the pitch parameters. This increases the prediction gain and produces a stable pitch syn-
thesis filter.

The synthesized decimated signal $\hat{Y}(n)$ is split into sequences which are then put
together to form $Y(n)$. With the help of the corresponding position index $j$, associated
with each sequence, received sequences are shifted to the correct positions with necessary
zeros inserted in between the samples to form the final excitation signal at the decoder.

The overall bit rate of the CELP-BB is simply determined by the vector size of the
decimated signal and the pitch filter parameters update rate. By varying the vector
dimensions it is possible to achieve a range of bit rates from 9.6 Kbps to 2.4 Kbps.

2.2 Results

Although, CELP-BB is a base-band coder, most of the transmission capacity is occupied
by the CELP coder operating in the base-band. It is therefore, necessary to adjust
the parameters of CELP to achieve a given overall bit rate. The flexibility of CELP-BB is its
decimation factor which enables smaller residual vectors and more frequent pitch param-
eter updates. In the following sections we discuss the coder performance for various bit
rates.

2.2.1 9.6 - 4.8 Kbps CELP-BB

A typical 9.6-4.8 Kbps CELP-BB has the following parameters given in Table 1.

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Bits</th>
<th>9.6-7 Kbps</th>
<th>4.8 Kbps</th>
</tr>
</thead>
<tbody>
<tr>
<td>LPC</td>
<td>40</td>
<td>1641</td>
<td>1333.3</td>
</tr>
<tr>
<td>Pitch</td>
<td>10</td>
<td>2052</td>
<td>1333.3</td>
</tr>
<tr>
<td>Vector</td>
<td>8-9</td>
<td>1641</td>
<td>1200</td>
</tr>
<tr>
<td>Gain</td>
<td>6-5</td>
<td>1231</td>
<td>666.7</td>
</tr>
<tr>
<td>Position</td>
<td>2</td>
<td>410</td>
<td>266.7</td>
</tr>
</tbody>
</table>

Table 1. CELP-BB Bit Allocation For 9.6, 7 and 4.8 Kbps.

9.6 to 7 Kbps Coding

Using a decimation factor of 3, CELP-BB at 7 Kbps was tested by calculating its objective
SNRs and conducting informal listening tests. SNR for both the base-band quantization
and the overall coder performance was calculated. The coder was also tested with a three
tap pitch filter. In Table 2 SNR performances of CELP-BB with one and three tap pitch
filters are tabulated.

<table>
<thead>
<tr>
<th>Pitch Tap</th>
<th>1-Tap</th>
<th>3-Tap</th>
</tr>
</thead>
<tbody>
<tr>
<td>(dB)</td>
<td>SegSNR</td>
<td>SNR</td>
</tr>
<tr>
<td>BB quantization</td>
<td>13.05</td>
<td>12.19</td>
</tr>
<tr>
<td></td>
<td>14.24</td>
<td>12.34</td>
</tr>
<tr>
<td>BB prediction</td>
<td>8.40</td>
<td>7.67</td>
</tr>
<tr>
<td></td>
<td>9.77</td>
<td>8.07</td>
</tr>
<tr>
<td>Performance</td>
<td>8.46</td>
<td>7.88</td>
</tr>
<tr>
<td></td>
<td>9.06</td>
<td>8.32</td>
</tr>
</tbody>
</table>

Table 2: CELP-BB performance at 7 and 9.6 Kbps using one
and three tap pitch filters respectively.
Using a 3 tap pitch filter requires an additional 8 bits per sequence hence, the overall bit rate of the coder is about 9.6 Kb/s. The subjective quality of CELP-BB using one and three tap pitch filters was comparable to the original speech quality. This was achieved by efficient coding of the base-band. Coding of the base-band was in both cases better than 3 bits per sample scalar quantization which has about 13 dB SegSNR.

The overall coder SNR performance is in the region of 8.5 dB for 7 Kb/s. As CELP-BB is a base-band coder it would be expected to have a lower SNR value than for the full-band coders. Base-band coders are best tested and compared using subjective listening tests. Therefore, we have compared CELP-BB with CELP [1] and VQTC [4] operating at 7 Kb/s. Informal listening tests showed that all three coders had very similar quality which was comparable to the original speech.

4.8 Kb/s Coding
Here, again a decimation factor of 3 was used. With the bit allocation to the specific parameters given in Table 1, objective and subjective performance of CELP-BB at 4.8 Kb/s was evaluated in exactly the same way as was done for the 7.0 Kb/s. The SNR performance of CELP-BB at 4.8 and 5.8 Kb/s using one and three tap pitch filters respectively is given in Table 3.

<table>
<thead>
<tr>
<th>Pitch Tap</th>
<th>1-Tap</th>
<th>3-Tap</th>
</tr>
</thead>
<tbody>
<tr>
<td>(dB)</td>
<td>SegSNR</td>
<td>SNR</td>
</tr>
<tr>
<td>BB quantization</td>
<td>8.16</td>
<td>7.41</td>
</tr>
<tr>
<td>BB prediction</td>
<td>5.42</td>
<td>4.67</td>
</tr>
<tr>
<td>Performance</td>
<td>6.11</td>
<td>5.78</td>
</tr>
</tbody>
</table>

Table 3: CELP-BB performance at 4.8 and 5.8 Kb/s using one and three tap pitch filters respectively.

Again using three tap pitch filter requires an additional 8 bits per sequence information which takes the total bit rate to 5.8 Kb/s. The difference between the 5.8 and the 4.8 Kb/s cases is not significant since as the vector length becomes larger the performance of one and three tap pitch filters becomes closer.

Results of the informal listening tests showed that CELP-BB maintains its high quality even at 4.8 Kb/s and of course outperforms CELP and VQTC at bit rates below 6 Kb/s. CELP-BB at 4.8 Kb/s was also compared with CELP-BB at 7 Kb/s. It was noticed that although the quality at both bit rates was very close to the original quality, when compared using highly sensitive ear-phones the quality at 7 Kb/s sounded more refined. In the proposed coder no roughness or quantization noise was noticed.

2.2.2 2.4 Kb/s CELP-BB
The results discussed above encouraged us in the possibility of reducing the bit rate of CELP-BB below 4.8 Kb/s. Therefore it was tested at 2.4 Kb/s. Its 10 LPC parameters were vector quantized with a 10 bit full search code-book and the decimation factor was increased to 4. The base-band quantization and prediction performances were 3.194 dB and 1.497 dB respectively. The subjective quality of the coder was natural and intelligible but it contained some quantization error and roughness. The reasons for reduced performance were those of poor pitch prediction and hence poor base-band coding, and the poor performance of the weighting filter and selecting the maximum energy sequence. When the vector size of the base-band residual becomes larger the energy in each sequence tends to be equal which does not require the transmission of the pulse location.
Replacing the weighting filter with a low-pass filter and discarding the grid selection, i.e. implementing an ordinary base-band coder, was tested with the parameters given in Table 4.

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Bits</th>
<th>Bit-Rate</th>
</tr>
</thead>
<tbody>
<tr>
<td>LPC</td>
<td>10</td>
<td>400</td>
</tr>
<tr>
<td>Pitch</td>
<td>10</td>
<td>800</td>
</tr>
<tr>
<td>Vector</td>
<td>10</td>
<td>800</td>
</tr>
<tr>
<td>Gain</td>
<td>5</td>
<td>400</td>
</tr>
</tbody>
</table>

Table 4: CELP-BB Bit Allocation For 2.4 Kb/s.

Using the specifications given in Table 4, the pitch prediction and the base-band quantization performance was increased by 0.48 dB and 0.84 dB respectively. Replacing the weighting filter by a low-pass filter caused slight high frequency regeneration noise, however, the overall quality of the speech was much smoother. Overall speech quality was natural, intelligible and smooth and it did not contain undesirable clicks or annoying energy level variation noise that is present in LPC-10.

3.0 CONCLUSIONS

In this paper, we have discussed a new base-band coder and compared its subjective quality with CELP and VQTC at various bit rates from 9.6 to 4.8 Kb/s. We have also shown that CELP-BB is capable of producing natural, intelligible and smooth speech at 2.4 Kb/s.

The code-book search in CELP-BB requires no convolution. The pitch filter memory is subtracted from the original base-band signal to form the reference signal which is directly matched by the code-book sequences. Another complexity advantage of CELP-BB is that, because it is a base-band coder its complexity is further divided by its decimation factor. At 7 and 4.8 Kb/s using 8 and 9 bit code-books and a decimation factor of 3 the code-book search requires about 0.7 MIPS and 1.4 MIPS respectively. Even if 10 bit code-book was used the code-book search computations would be less than 3 MIPS (using Gaussian code-books). Hence, the overall complexity of CELP-BB at 9.6 Kb/s to 4.8 Kb/s is well within the capabilities of a single AT&T DSP-32 signal processing chip.

The quality of CELP-BB coded speech at 9.6 to 7 Kb/s is comparable to CELP coded speech at the same bit rates. Below 6 Kb/s however, CELP-BB outperforms CELP. CELP-BB coded speech at 4.8 Kb/s is much better than CELP coded speech. High speech quality can be maintained to bit rates as low as 4 Kb/s. In Figure 3, speech waveform plots for the original base-band and their 7 and 4.8 CELP-BB coded versions are shown.

Finally, CELP-BB was tested at 2.4 Kb/s, and although, at this bit rate it required vector quantization of the LPC parameters, its complexity was still relatively low. Speech quality at 2.4 Kb/s was intelligible, natural and smooth which makes it attractive for both military and civilian uses.

4.0 REFERENCES


Figure 1: A block diagram of CELP-BB.

Figure 2: A block diagram of CELP operating on the decimated base-band signal.

Figure 3: Typical waveforms of (a) original base-band and (b) and (c) coded base-band at 7 and 4.8 Kb/s respectively.
APPENDIX D
SOURCE CODE (IN C) OF IMPORTANT ALGORITHMS

CELP coding of SPEECH

******************************************************************************
#include <stdio.h>
#include <math.h>
#define index 256 /* code-book size */
#define lpcsize 10 /* order of LPC filter */
#define blkno 952 /* number of vectors to be processed */
#define vecsize 40 /* excitation vector size */
#define register 150 /* pitch filter memory register */
int check[blkno]; /* To see which vectors in table used */
float blkener[blkno]; /* energi equalizerdue extra energi from lpc*/
float compute(), keepin[vecsize]; /* Size of the data re-transfer */
float inv[vecsize],B1,B2,B3;
float data[vecsize]; /* code-book size /
float lpcoef[lpcsize]; /* pitch FB loop contents from blk to blk */
float pitch[register]; /* buff to store pitch FB loop contents */
float ppp[register]; /* Buffs to transfer */
float pp[register]; /* Buffs to transfer */
float lpc[lpcsize]; /* Buffs to transfer */
float lll[lpcsize]; /* Buffs to transfer */
float ll[lpcsize]; /* Buffs to transfer */
float wwl[lpcsize]; /* Buffs to transfer */
float ww2[lpcsize]; /* Buffs to transfer */
float wwwl[lpcsize]; /* Buffs to transfer */
float www2[lpcsize]; /* Buffs to transfer */
int Pitch; /* pitch period */
float power[lpcsize]; /* error weight */
float Rx,R0,Rp,Sn1,Sn2,Sn3,Sna,Snb,Snc,Snd,Sne,Snf;

main()
{
  int coe,q,w,we,e,r,t,offd=0,offlpc=0,offpic=0,a,s,loop1,loop2,loop,cul;
  float preserror=0.0,pasterror=0.0,en=0.0;
  char *in="book", *fi="input_file", *fo="output_file", *fing="predicted"
  char *fl="lpcoefficient", *ff="scalar", *res="see.indexes"
  char *sec="lpcresidual";
  float delta,del1,del2,del3;

  for(t=1;t<13;t++)
power[t-1]=pow(0.92446525, (float) t);

    t=creat(fo, 0644);
    close(t);
    t=open(fo, 1);
    coe=open(sec, 0);
    cul=creat(fing, 0644);
    close(cul);
    cul=open(fing, 1):

    for(q=0; q < blkno; q++)
    {
        read(coe.datares.sizeof(datares));
        w=open(fl, 0);
        lseek(w.offd, 0);
        read(w.data.sizeof(data));
        close(w);

        w=open(fl, 0);
        lseek(w.offlpc, 0);
        read(w.lpccoef.sizeof(lpccoef));
        close(w);

        offlpc=offlpc-sizeof(lpccoef);
        offd=offd+sizeof(data);

        w=open(in, 0);

        preserror=0.0;
pasterror=(-1.0)*9999999.0;
    r=0;
    for(a=0; a < register; a++)
    pitch[a]=pp[a];

    /* Compute pitch and pitch coefficients */
    Rp=Rx=R0=0.0;
    for(loop=0; loop < register-vecsize; loop++)
    {
        Rx=0.0;
        for(loop1=0; loop1 < vecsize; loop1++)
        Rx=Rx+(datares[loop1]*pitch[vecsize-1-loop1+loop]);
        if(Rx > Rp)
        {
            Rp=Rx;
Pitch=loop+vecsize-1;
        }
    }

Sn1=Sn2=Sn3=Sna=Snb=Snc=Snd=Sne=Snf=0.0;
delta=delt1=del2=del3=0.0;
for(loop=0; loop < vecsize; loop++)
{
    Sn1=Sn1+(pitch[Pitch-1-loop]*datares[loop]);
    Sn2=Sn2+(pitch[Pitch-loop]*datares[loop]);
\[
\begin{align*}
\text{Sn3} &= \text{Sn3} + (\text{pitch}[\text{Pitch}+1 \text{-loop}] \times \text{datares}[\text{loop}]); \\
\text{Sna} &= \text{Sna} + (\text{pitch}[\text{Pitch}-1 \text{-loop}] \times \text{pitch}[\text{Pitch}-1 \text{-loop}]); \\
\text{Snb} &= \text{Snb} + (\text{pitch}[\text{Pitch} \text{-loop}] \times \text{pitch}[\text{Pitch} \text{-loop}-1]); \\
\text{Snc} &= \text{Snc} + (\text{pitch}[\text{Pitch} \text{-loop}+1] \times \text{pitch}[\text{Pitch} \text{-loop}]); \\
\text{Snd} &= \text{Snd} + (\text{pitch}[\text{Pitch} \text{-loop}] \times \text{pitch}[\text{Pitch} \text{-loop}]); \\
\text{Sne} &= \text{Sne} + (\text{pitch}[\text{Pitch} \text{-loop}+1] \times \text{pitch}[\text{Pitch} \text{-loop}]); \\
\text{Snf} &= \text{Snf} + (\text{pitch}[\text{Pitch} \text{-loop}+1] \times \text{pitch}[\text{Pitch} \text{-loop}+1]); \\
\text{delta} &= (\text{Sna} \times \text{Snd} \times \text{Snf}) + (\text{Snb} \times \text{Sne} \times \text{Snc}) + (\text{Snb} \times \text{Sne} \times \text{Snc}); \\
\text{delta} &= \text{delta} - (\text{Snc} \times \text{Snd} \times \text{Snf}) - (\text{Snb} \times \text{Sne} \times \text{Sn3}) - (\text{Sne} \times \text{Sne} \times \text{Sna}); \\
\text{del1} &= (\text{Sn1} \times \text{Snd} \times \text{Snf}) + (\text{Sn1} \times \text{Sne} \times \text{Sn3}) + (\text{Sn2} \times \text{Sne} \times \text{Snc}); \\
\text{del1} &= \text{del1} - (\text{Sn3} \times \text{Snd} \times \text{Snc}) - (\text{Sn2} \times \text{Sn2} \times \text{Snf}) - (\text{Sne} \times \text{Sne} \times \text{Sna}); \\
\text{del2} &= (\text{Sna} \times \text{Sn2} \times \text{Snf}) + (\text{Sn1} \times \text{Sne} \times \text{Snc}) + (\text{Snb} \times \text{Sn3} \times \text{Snf}); \\
\text{del2} &= \text{del2} - (\text{Sn3} \times \text{Sn2} \times \text{Snf}) - (\text{Snb} \times \text{Sn1} \times \text{Snf}) - (\text{Sn3} \times \text{Sne} \times \text{Sna}); \\
\text{del3} &= (\text{Sna} \times \text{Snd} \times \text{Sn3}) + (\text{Snb} \times \text{Sn2} \times \text{Snc}) + (\text{Snb} \times \text{Sne} \times \text{Sn1}); \\
\text{del3} &= \text{del3} - (\text{Sn3} \times \text{Snd} \times \text{Sn1}) - (\text{Snb} \times \text{Sn3} \times \text{Sn3}) - (\text{Sne} \times \text{Sna} \times \text{Sn2}); \\
\text{if} (\text{delta} == 0.0) \\
&\quad \text{B1} = \text{B2} = \text{B3} = 0.0; \\
\text{else} \\
&\quad \text{B1} = \text{del1}/\text{delta}; \\
&\quad \text{B2} = \text{del2}/\text{delta}; \\
&\quad \text{B3} = \text{del3}/\text{delta}; \\
&\quad \text{quantize}(\text{B1}, \text{B2}, \text{B3}); \\
&\quad /* \text{Compute filter memory and subtract from the original} */ \\
&\quad \text{for} (a = 0; a < \text{lpcsize} ; a++) \\
&\quad \quad \text{lpc}[a] = \text{ll}[a]; \\
&\quad \quad \text{pitchinsert}(10, q); \\
&\quad \quad \text{lpcinsert}(); \\
&\quad \quad \text{subtract}(); \\
&\quad \text{write(cul.datap.sizeof(datap));} \\
&\quad /* \text{Search the code-book} */ \\
&\quad \text{for} (e = 0; e < \text{index} ; e++) \\
&\quad \quad \text{read(w, inov.sizeof(inov));} \\
&\quad \text{for} (a = 0; a < \text{register} ; a++) \\
&\quad \quad \text{pitch}[a] = 0.0; \\
&\quad \text{for} (a = 0; a < \text{lpcsize} ; a++) \\
&\quad \quad \text{lpc}[a] = 0.0; \\
&\quad \text{pitchinsert}(0, q); \\
&\quad \text{lpcinsert}(); \\
&\quad /* \text{Noise shaping excluded} */ \\
&\quad \text{for} (a = 0; a < \text{lpcsize} ; a++) \\
&\quad \quad \text{wtlpc1[a]} = 0.0;
\end{align*}
\]
wtlpc2[a]=0.0;
}
weight();

* /
preserror=compute(q);
if(preserror > pasterror)
{
pasterror=preserror;
r=e;
    en=blkener[q];
}

/* restore the synthesis filter memory */
for(a=0;a < register;a++)
pitch[a]=pp[a];
for(a=0;a < lpcsize;a++)
lpc[a]=ll[a];
check[q]=r;
close(w);
w=open(in, 0);
iseek(w,(r*4*vecsize),0);
read(w, keepin.sizeof(keepin));
close(w);

blkener[q]=en;
for(w=0;w < vecsize;w++)
inov[w]=keepin[w]*blkener[q];
pitchinsert(0,q);
lpcinsert();

/* Keep the current filter memory */
for(a=0;a < register;a++)
pp[a]=pitch[a];
for(a=0;a < lpcsize;a++)
ll[a]=lpc[a];
for(a=0;a < vecsize;a++)
data[a]=(short)lperes[a];
write(t.data,sizeof(data));
}
we=creat(res, 0644);
write(we.check.sizeof(check));
close(we);
close(t);
t=creat(tf,0644);
write(t.blkener.sizeof(blkener));
close(t);
close(coe);
}

int pitchinsert(pe,s)
    int s,pe;
{
int i.o.p.M;
float k.l=0.0;
M=Pitch;
if(pe==0)
{
    for(i=0; i<vecsiz;i++)
    {
        i=(B2*pitch[M])+(B1*pitch[M-1])+(B3*pitch[M+1]);
        picres[i]=inov[i]+1;
        for(o=register-1:o>0:o--)
            pitch[o]=pitch[o-1];
        pitch[0]=inov[0]+1;
        l=0.0;
    }
}
else
{
    for(i=0;i<vecsiz;i++)
    {
        i=(B2*pitch[M])+(B1*pitch[M-1])+(B3*pitch[M+1]);
        picres[i]=1;
        for(o=register-1:o>0:o--)
            pitch[o]=pitch[o-1];
        pitch[0]=1;
        l=0.0;
    }
}

return;
}

int 1pcinsert()
{
    int i.o.p;
    float k.l=0.0;
    for(i=0;i<vecsiz;i++)
    {
        k=0.0;
        for(p=0;p<1pcsize;p++)
            k=k+(1pc[p]*1pcoef[p]);
        lpcres[i]=k+picres[i];
        for(o=1pcsize-1:o>0:o--)
            lpc[o]=lpc[o-1];
        lpc[0]=picres[0]+k;
    }
    return;
}

subtract()
{
    int i;
    for(i=0;i<vecsiz;i++)
float compute(lk)
{
    int lk;
    
    int i.o.p;
    float k,l;
    k=1=0.0;
    for(i=0;i<vecsize;i++)
    {
        k=k+(data[i]*lpres[i]);
        l=l+(lpres[i]*lpres[i]);
    }
    blkener[lk]=k/l;
    k=(k*k)/l;
    return(k);
}

int weight()
{
    int i.o.p;
    float k,l,m=0.0;
    float sec[vecsize];

    for(i=0;i<vecsize;i++)
    {
        k=0.0;
        for(o=0;o<lpsize;o++)
            k=k+(lpcoef[o]*wlpc1[o]);
        sec[i]=lpres[i]-k;
        for(o=lpsize-1;0>o--)
            wtlp1[o]=wtlp1[o-1];
            wtlp1[0]=lpres[i];
    }
    for(i=0;i<vecsize;i++)
    {
        l=0.0;
        for(o=0;o<lpsize;o++)
            l=l+(lpcoef[o]*wlpc2[o]*power[o]);
            weighted[i]=sec[i]+l;
            for(o=lpsize-1;0>o--)
                wtlp2[o]=wtlp2[o-1];
                wtlp2[0]=sec[i];
    }
    return;
}
#include <stdio.h>
#include <math.h>
#include <sys/types.h>
#include <sys/dir.h>
#include <sys/stat.h>
#define MAX 64
#define MAX2 200
#define MAX3 1024
#define pi 3.141592654

main(argc, argv)
int argc;
char *argv[];
{
    int r, i, j, k, p, fd, pitch, offset, s_offset, no_blks, filesize;
    float sum, Rp, Ro, value, mod_sum, gain, x;
    char *f_res, *f_dct, *f_op, *f_lpc="feedbk", *f_in="Res332.m";
    char *f_pit="pit128.10", *f_hel="hellop2";
    float DCT(), INDCT(), kernel(), VQ(), sort_book(), weight();
    int fsize();
    int l_offset, p_os;
    float sn1,sn2,sn3,sn4,snb,snc,snd,sne,pnf;
    float dela,del1,del2,del3;
    float B1, B2, B3;

    /* check input parameters */
    if (argc!=8) {
        printf("Usage: p_dct ve_size pi_len book_size Res_f DCT_f code_f OP_f 0):
        exit(-1);
    }
    /* assign input parameters */
tsize = atoi(argv[1]);
p_len = atoi(argv[2]);
booksize = atoi(argv[3]);
f_res = argv[4];
f_dct = argv[5];
f_book = argv[6];
f_op = argv[7];
/* creat O/P file */
i = creat(f_op, 0644);
close(i);

i = creat(f_pit, 0644);
close(i);

i = creat(f_in, 0644);
close(i);

error = fopen("Me3.32.err", "w");
nor_err = fopen("Me3.32.norm", "w");
/* find size of residual file */
filesize = fsize(f_res);
if (filesize < 0)
    exit(0);
else
    no_blks = (int)(filesize/4)/tsize;

/* initialisation of the buffers */
for (i=0; i<tsize; i++)
{
    buf1[i]=buf2[i]=buf3[i]=0.0;
    buf4[i]=buf5[i]=buf7[i]=buf12[i]=buf16[i]=0.0;
    buf9[i]=buf8[i]=buf11[i]=buf13[i]=buf14[i]=buf15[i]=0.0;
}
for (i=0; i<p_len; i++)
    buf6[i]=0.0;

kernel();  /* cal. fmat and imat kernel of DCT-IDCT */
sort_book();  /* sort codebook into the array "book" */
s_offset = 0.0;
offset = 0.0;
l_offset = 0.0;
p_os = 0.0;
r = 4;
/* block by block processing */
for (n=0; n<no_blks; n++)
{
    fd = open(f_hel, 0);  /* read in block[tsize] from input original file */
lseek(fd,s_offset,0);
read(fd,inbuf,(2*tsize));
close(fd);

fd = open(f_res,0); /* read in block[tsize] from input residual file */
lseek(fd.offset,0);
read(fd,buf1,(4*tsize));
close(fd);

/* read in from weighting and LPC coeff. file every (length/tsize=4) loop */
if (r = 4)
{
    r = 0;
p = open(f_dct,0); /* read in block[tsize] from the weighting file */
lseek(p.p_offset,0);
read(p.buf10,(16*tsize));
close(p);

    p = open(f_lpc,0); /* read in block[order] from LPC coeff. file */
lseek(p.p_os,0);
read(p.aji,(4*H));
close(p);
p_os = p_os + (4*H);
_l_offset = l_offset + (16 * tsize);
}

r = r + 1;

for (i=0; i<tsize; i++)
    buf16[i] = (float)(inbuf[i]);

/* calculate pitch */

gain = 0.0;
pitch = 31;
Rp = 0.0;
value = 0.0;
mod_sum = 0.0;
for (i=0; i<(p_len-tsize); i++)
{
    sum = 0.0;
    for (p=0; p< tsize; p++)
    {
        sum = sum + (buf6[p+i] * buf1[tsize-1-p]);
    }
    if (sum >= value)
    {
        pitch = tsize + i - 1;
        Rp = sum;
        value = sum;
    }
/* calculate gain */

sni = sn2 = sn3 = sn4 = sn5 = sn6 = sn7 = sn8 = 0.0;
dela = del1 = del2 = del3 = 0.0;

for (p = 0; p < tsize; p++)
{
    sn1 = sn1 + (buf6[pitch-1-p] * buf1[p]);
    sn2 = sn2 + (buf6[pitch-p] * buf1[p]);
    sn3 = sn3 + (buf6[pitch+1-p] * buf1[p]);
    sna = sna + (buf6[pitch-1-p] * buf6[pitch-1-p]);
    snb = snb + (buf6[pitch-p] * buf6[pitch-p-1]);
    snc = snc + (buf6[pitch-p+1] * buf6[pitch-p-1]);
    snd = snd + (buf6[pitch-p] * buf6[pitch-p]);
    sne = sne + (buf6[pitch-p+1] * buf6[pitch-p]);
    snf = snf + (buf6[pitch-p+1] * buf6[pitch-p+1]);
}

dela = (sna*snd*snf) + (snb*sne*snc) + (snb*sne*snc);
dela = dela - (sna*snd*snf) - (snb*sne*snc) - (sne*sne*sna);

del1 = (sn1*snd*snf) + (snb*sne*sn3) + (sn2*sne*snc);
del1 = del1 - (sn1*snd*snf) - (snb*sne*sn3) - (sne*sne*sn1);

del2 = (sna*sn2*snf) + (snb*sn2*sn3) + (snb*sn3*snc);
del2 = del2 - (sna*sn2*snf) - (snb*sn2*sn3) - (sne*sne*sn1);

del3 = (sna*sn3*snf) + (snb*sn3*sn3) + (snb*sn3*sn1);
del3 = del3 - (sna*sn3*snf) - (snb*sn3*sn3) - (sne*sne*sn2);

if (dela == 0.0)
    B1 = B2 = B3 = 0.0;
else
    {
        B1 = del1/dela;
        B2 = del2/dela;
        B3 = del3/dela;
    }

/* apply pitch and gain to calculate the feedback buffer */

for (i = 0; i < tsize; i++)
{
    buf7[i] = (B2*buf6[pitch-i]) + (B1*buf6[pitch-1-i]) + (B3*buf6[pitch+1-i]);
}

for (i = 0; i < tsize; i++)
{
    for (p = (p_len-1); p > 0; p--)
        buf6[p] = buf6[p-1];  /* shift register by one place */
}

/* inverse LPC filtering */
for (i = 0; i < tsize; i++)
{
buf13[i] = buf7[i];
}
for (i=0; i<(tsize+H); i++)
    buf14[i] = buf11[i];
for (k=0; k<H; k++)
    buf14[k] = buf14[k+tsize];
/* flush out memory */
for (k=H; k<(tsize+H); k++)
{
    x = 0.0;
    for (j=0; j<H; j++)
    {
        x = x + (ajl[j] * buf14[k-1-j]);
    }
    buf14[k] = x + buf13[k-H];
    buf15[k-H] = buf14[k];
}
/* apply feedback buffer to input and output */
for (i=0; i<tsize; i++)
{
    buf2[i] = buf1[i] - buf7[i];
}
/* DCT then VQ then INDCT residual signal */
DCT();
VQ();
INDCT();

/* insert quantised residual into pitch buffer */
for (i=0; i<tsize; i++)
{
    buf6[tsize-i-1] = buf5[i] + buf7[i];
    buf9[i] = buf5[i] + buf7[i];
}
/* synthesis LPC filtering */
for (k=0; k<H; k++)
    buf11[k] = buf11[k+tsize];
for (k=H; k<(tsize+H); k++)
{
    x = 0.0;
    for (j=0; j<H; j++)
    {
        x = x + (ajl[j] * buf11[k-1-j]);
    }
    buf11[k] = x + buf9[k-H];
    buf12[k-H] = buf11[k];
    opbuf[k-H] = (short)(buf11[k]);
}
/* store output and other intermediate files */
fd = open(f_op.1);
lseek(fd,s_offset,0);
write(fd,opbuf,(2*tsize));
close(fd);

i=open(f__pit,1);
lseek(i,offset,0);
write(i,buf9,(4*tsize));
close(i);

i = open(f__in.1);
lseek(i,offset,0);
write(i,buf5,(4*tsize));
close(i); 

offset = offset + (4*tsize);
s__offset = s__offset + (2*tsize);

} 
} 

/* fsize: function to determine size of file in characters */

fsize(ipname)
char ipname[];
{
    struct stat stbuf;
    int p;
    if (stat(ipname,&stbuf) == -1)
    {
        printf("fsize: cannot find %s0, ipname); 
        p = -1;
        return(p);
    }
    else
    {
        p = stbuf.st_size;
        return(p);
    }
}

/* kernel: function to calculate the transform kernel of DCT and IDCT */ 
/* for a given transform size. */

float kernel()
{
    int i, p;
    float x, y;
    for (i=0; i<tsize; i++)
    {

x = 0.0;
y = 0.0;
for (p=0; p < tsize; p++)
{
    y = ((2*i)+1)*p*pi;
    x = ((2*p)+1)*i*pi;
    y = y/(2*tsize);
    x = x/(2*tsize);
    fmat[i][p] = cos(x);
    imat[i][p] = cos(y);
    x = 0.0;
    y = 0.0;
}

/* DCT: Discrete cosine transform function. Uses the "fmat"-matrix */
/* of the function "kernel". */

float DCT()
{
    int i, p;
    float x;

    for (i=0; i < tsize; i++)
    {
        x = 0.0;
        for (p=0; p < tsize; p++)
            x = x + (buf2[p]*fmat[i][p]);
        if (i == 0)
            buf3[i] = (sqrt(2.0/tsize)) * x * (sqrt(0.50));
        else
            buf3[i] = (sqrt(2.0/tsize)) * x;
    }

    /* INDCT: Inverse DCT. Uses "imat"-matrix of "kernel" */

    float INDCT()
    {
        int i, p;
        float x;

        for (i=0; i < tsize; i++)
        {
            x = 0.0;
            for (p=0; p < tsize; p++)
            {
                if (p==0)
                    x = x + (buf4[p] * imat[i][p] * (sqrt(0.50)));
                else
                    x = x + (buf4[p] * imat[i][p]);
            }
            buf5[i] = sqrt(2.0/tsize) * x;
float sort_book()
{
    int fd, p, i, j;
    float sum, q[32768];

    fd = open(f_book, 0);
    lseek(fd, 0, 0);
    read(fd, q, sizeof(q));
    close(fd);

    /* get codebook into array */

    p = 0.0;
    for (i=0; i<booksize; i++)
    {
        for (j=0; j<tsize; j++)
        {
            book[i][j] = q[p];
            p = p + 1;
        }
    }

    /* normalise the codebook to unit variance */

    for (i=0; i<booksize; i++)
    {
        sum = 0.0;
        for (j=0; j<tsize; j++)
        {
            sum = sum + (book[i][j] * book[i][j]);
        }
        sum = sum / tsize;
        sum = sqrt(sum);
        for (j=0; j<tsize; j++)
        {
        }
    }

    /* VQ: vector quantised the residual signal */

    float VQ()
    {
        int i, j, index, k;
        float opt, sign, Com_gain, x, value, sum, vec_sq, dist, G, min, max;
        float opt_sign, Com_sign;
        float Error, Nor_err, res_sq, ori_enr, Com_cor;
index = 0;
dist = 9999999999.0;
opt = 0.0;
max = 0.0;

Com_gain = 0.0;

sum = 0.0;

/* search for best codebook entry */
for (i=0; i<booksize; i++)
{
    sum = 0.0;
    value = 0.0;
    vec_sq = 0.0;
    G = 0.0;

    for (j=0; j<tsize; j++)
    {
        vec_sq = vec_sq + (book[i][j] * book[i][j]);
        sum = sum + (book[i][j] * buf3[j]);
    }
    G = sum/vec_sq;

    min = 0.0;
    for (j=0; j<tsize; j++)
    {
        sum = (G*book[i][j]) - buf3[j];
        value = (sum * buf8[j]);
        min = min + (value * value);
    }

    if (min <= dist)
    {
        index = i;
        dist = min;
    }
}

/* find gain using the index codeword */
sum = 0.0;
vec_sq = 0.0;

/* synthesis using codeword without gain applied */
for (i=0; i<tsize; i++)
    buf4[i] = book[index][i];

INDCT();

for (j=0; j<(tsize+H); j++)
{
    lpc_zero[j] = 0.0;
for (k=H; k<(tsize+H); k++)
{
    x = 0.0;
    for (j=0; j<H; j++)
    {
        x = x + (aji[j] * lpc_zero[k-1-j]);
    }
    lpc_zero[k] = x + buf5[k-H];
    empty[k-H] = lpc_zero[k];
}
/* find codeword gain */

res_sq = 0.0;
ori_enr = 0.0;
for (i=0; i<tsize; i++)
{
    Res[i] = buf16[i] - buf15[i];
    res_sq = res_sq + (Res[i] * Res[i]);
    ori_enr = ori_enr + (buf16[i] * buf16[i]);
}

sum = 0.0;
vec_sq = 0.0;
for (i=0; i<tsize; i++)
{
    sum = sum + (Res[i] * empty[i]);
    vec_sq = vec_sq + (empty[i] * empty[i]);
}

Com_cor = (sum*sum)/vec_sq;
Com_gain = sum/vec_sq;

Error = res_sq - Com_cor;
Nor_err = Error/ori_enr;

fprintf(error, ": %f Error);
fprintf(nor_err, ": %f Nor_err);

/* final VQ output of residual */
for (i=0; i<tsize; i++)
    buf4[i] = Com_gain * book[index][i];

/* "weight": function to reduce the dynamics of the weighting buffer */

float weight()
{
    int i, j;
    float sum;
    for (j=0; j<tsize; j++)
{ 
  i = j * 4;
  sum = 0.0;
  sum = sum + (buf10[i] * buf10[i]) + (buf10[i+1] * buf10[i+1]);
  sum = sum + (buf10[i+2] * buf10[i+2]) + (buf10[i+3] * buf10[i+3]);
  sum = sum/4;
  sum = sqrt(sum);
  buf8[j] = sum;
}
```c
#include <stdio.h>
#include <math.h>
#define index 1024 /* code-book size */
#define vcsesize 27 /* decimated base-band residual vector size */
#define register 70 /* pitch filter memory register */
#define length 108 /* LPC-residual vector size before decimation */
#define number 650 /* number of vectors to be processed */
#define deci 4 /* decimation factor */

float computeO;
float memory[11].weigted[length+10];
float trans[length];
int check[number]; /* To see which vectors in table used */
float blkener[number]; /* energi equalizer due extra energi from lpc*/
int shift.period;
float R0RpR0pRx.pgain;
float keepin[vcsesize]; /* Size of the data re-transfer */
float inov[vcsesize];
float picres[vcsesize]; /*buff for pitch filter output */
float error[vcsesize]; /* buff to hold original-synthetic speech */
float data[length]; /* input data transfer buff */
float data1[vcsesize]; /* input data transfer buff */
float data2[vcsesize]; /* input data transfer buff */
float ener[number]; /* block energy */
float pitch[register]; /* buff to store pitch feedback-loop */
float pp[register]; /* ppp[] and pp[] are buffs to transfer */
float pp[register]; /* pitch FB loop contents from blk to blk */
float power[12]; /* error weight */

main()
{
    int coe,q.w,we.e.r.t.offd=0,offp=0,offpic=0,a.s;
    float preserror=0.0,pasterror=0.0;
    char *in="code-book", *fi="lpresidual", *fo="output_file":
    float en=0.0;
    char *eng="energi";
    char *ff="scalar";
    char *res="see.vectors";

    t=creat(fo,0644);
    close(t);
    t=open(fo.1);

    for(q=0;q<number;q++)
    {
        w=open(fi.0);
        Iseek(w,offd.0);
        read(w,data,sizeof(data));
        close(w);
}
```
offd = offd + sizeof(data);

/* Compute the weighted base-band */
weight();
    w = open(in, 0);
    preserror = 0.0;
    pasterror = (-1.0) * 9999999.0;
    r = 0;
    for(a = 0; a < register; a++)
        pitch[a] = pp[a];
    for(a = 0; a < vecsize; a++)
        data2[a] = weighted[5 + shift + (a * deci)];

/* Compute pitch and pitch coefficient */
Rp = R0 = 0.0;
    for(a = 0; a < register - vecsize; a++)
        {Rx = 0.0;
         for(coe = 0; coe < vecsize; coe++)
             Rx = Rx + (data2[coe] * pitch[vecsize - 1 - coe + a]);
            if(Rx > Rp)
                {Rp = Rx;
                 period = a + vecsize - 1;
                }
        }
    for(a = 0; a < vecsize; a++)
        R0 = R0 + (pitch[period - a] * pitch[period - a]);
    if(R0 == 0.0)
        pgain = 0.0;
    else
        pgain = Rp / R0;
    quantize(pgain);
    pitchinsert(10, q);
/* Subtract the pitch filter memory from the LPCres */
subtract();
    for(e = 0; e < index; e++)
        {read(w, inov, sizeof(inov));

for(a = 0; a < register; a++)
    pitch[a] = 0.0;
    pitchinsert(0, q);
    preserror = compute(q);
    if(preserror > pasterror)
        {pasterror = preserror;
         r = e;
         en = blkener[q];
        }
/* restore the pitch filter memory */

for(a=0;a < register;a++)
pitch[a]=pp[a];
check[q]=r;
close(w);
w=open(in, 0);
llseek(w,(r*4*vecsizw).0);
read(w,keepin,sizeof(keepin));
close(w);

blkener[q]=en;
for(w=0;w < veceize:w++)
inov[w]=keepin[w]*blkener[q];
pitchinsert(0,q);

/* Keep the current pitch filter memory */

for(a=0;a < register;a++)
pp[a]=pitch[a];
for(a=0;a < length;a++)
trans[a]=0.0;

/* Form the upsampled LPCexcitation sequence */

for(a=0;a < vecsize;a++)
trans[shift+(a*deci)]=picres[a];
write(t,trans,sizeof(trans));

we=creat(res, 0644);
write(we,check,sizeof(check));
close(we);
close(t);
t=creat(ff, 0644);
write(t,blkener,sizeof(blkener));
close(t);
close(coe);

int pitchinsert(pe,s)
int s.pe:
{
    int i.o,p,M;
    float k,l=0.0;
    M=period;
    if(pe==0)
    {
        for(i=0; i < vecsize; i++)
        {
            l=(pgain*pitch[M]);
            picres[i]=inov[i]+l;
            for(o=register-1;o > 0;o--)
        }
    }
}
pitch[0]=pitch[0-1];
pitch[0]=inov[i]+1;
1=0.0;
}
}
else
{
  for(i=0;i<vecsize;i++)
  {
    l=(pgain*pitch[i]);
picres[i]=l;
    for(o=register-1;o>0;o--)
pitch[o]=pitch[o-1];
pitch[0]=l;
l=0.0;
  }
}
return;

subtract()
{
  int l;
  for(i=0;i<vecsize;i++)
data1[i]=(float)data2[i]-picres[i];
return;
}

float compute(lk)
int lk:
{
  int i.o.p;
float k,l;
l=1=0.0;
  for(i=0;i<vecsize;i++)
  {
    k=k+(data1[i]*picres[i]);
    l=l+(picres[i]*picres[i]);
  }
blkener[lk]=k/l;
k=(k*k)/l;
return(k);
}

weight()
{
  int i.p.s;
float k,l;
float filter[11]:

  filter[0]=filter[10]=(-0.016356);
  filter[1]=filter[9]=(-0.045650);
  filter[2]=filter[8]=(-0.000000);
filter[3]=filter[7]=(0.2507930);
filter[4]=filter[6]=(0.7007900);
filter[5]=(1.0000000);
k=0.0;
for(i=0;i < length+10;i++)
  l=0.0;
  for(s=10; s > 0; s--)
    memory[s]=memory[s-1];
  if(i < length)
    memory[0]=data[i];
  else
    memory[0]=0.0;
  for(s=0; s < 11; s++)
    l=1+(filter[s]*memory[s]);
  weighted[i]=l;
  l=0.0;

k=0.0;
shift=0;
for(i=0; i < deci; i++)
  l=0.0;
  for(s=0; s < vecsize; s++)
    l=1+(weighted[5+i+(s*deci)]*weighted[5+i+(s*deci)]);
  if(1 > k)
    k=1;
    shift=i;