# Neural Code Excited Linear Prediction for Low Power Adaptive Speech Coding

by

SriGouri Kamarsu

98

A Thesis
Submitted to the Faculty of Graduate Studies
in Partial Fulfilment of the Requirements
for the Degree of

**Master of Science**

Department of Electrical and Computer Engineering
University of Manitoba
Winnipeg, Manitoba

ISBN 0-612-13232-3

Name ______________________________

*Dissertation Abstracts International* is arranged by broad, general subject categories. Please select the one subject which most nearly describes the content of your dissertation. Enter the corresponding four-digit code in the spaces provided.

Electronics & Electrical | 0 5 4 4 | U·M·I

SUBJECT TERM | SUBJECT CODE

## Subject Categories

## THE HUMANITIES AND SOCIAL SCIENCES

### COMMUNICATIONS AND THE ARTS

| | |
|---|---|
| Architecture | 0729 |
| Art History | 0377 |
| Cinema | 0900 |
| Dance | 0378 |
| Fine Arts | 0357 |
| Information Science | 0723 |
| Journalism | 0391 |
| Library Science | 0399 |
| Mass Communications | 0708 |
| Music | 0413 |
| Speech Communication | 0459 |
| Theater | 0465 |

### EDUCATION

| | |
|---|---|
| General | 0515 |
| Administration | 0514 |
| Adult and Continuing | 0516 |
| Agricultural | 0517 |
| Art | 0273 |
| Bilingual and Multicultural | 0282 |
| Business | 0688 |
| Community College | 0275 |
| Curriculum and Instruction | 0727 |
| Early Childhood | 0518 |
| Elementary | 0524 |
| Finance | 0277 |
| Guidance and Counseling | 0519 |
| Health | 0680 |
| Higher | 0745 |
| History of | 0520 |
| Home Economics | 0278 |
| Industrial | 0521 |
| Language and Literature | 0279 |
| Mathematics | 0280 |
| Music | 0522 |
| Philosophy of | 0998 |
| Physical | 0523 |
| Psychology | 0525 |
| Reading | 0535 |
| Religious | 0527 |
| Sciences | 0714 |
| Secondary | 0533 |
| Social Sciences | 0534 |
| Sociology of | 0340 |
| Special | 0529 |
| Teacher Training | 0530 |
| Technology | 0710 |
| Tests and Measurements | 0288 |
| Vocational | 0747 |

### LANGUAGE, LITERATURE AND LINGUISTICS

| | |
|---|---|
| Language | |
| General | 0679 |
| Ancient | 0289 |
| Linguistics | 0290 |
| Modern | 0291 |
| Literature | |
| General | 0401 |
| Classical | 0294 |
| Comparative | 0295 |
| Medieval | 0297 |
| Modern | 0298 |
| African | 0316 |
| American | 0591 |
| Asian | 0305 |
| Canadian (English) | 0352 |
| Canadian (French) | 0355 |
| English | 0593 |
| Germanic | 0311 |
| Latin American | 0312 |
| Middle Eastern | 0315 |
| Romance | 0313 |
| Slavic and East European | 0314 |

### PHILOSOPHY, RELIGION AND THEOLOGY

| | |
|---|---|
| Philosophy | 0422 |
| Religion | |
| General | 0318 |
| Biblical Studies | 0321 |
| Clergy | 0319 |
| History of | 0320 |
| Philosophy of | 0322 |
| Theology | 0469 |

### SOCIAL SCIENCES

| | |
|---|---|
| American Studies | 0323 |
| Anthropology | |
| Archaeology | 0324 |
| Cultural | 0326 |
| Physical | 0327 |
| Business Administration | |
| General | 0310 |
| Accounting | 0272 |
| Banking | 0770 |
| Management | 0454 |
| Marketing | 0338 |
| Canadian Studies | 0385 |
| Economics | |
| General | 0501 |
| Agricultural | 0503 |
| Commerce-Business | 0505 |
| Finance | 0508 |
| History | 0509 |
| Labor | 0510 |
| Theory | 0511 |
| Folklore | 0358 |
| Geography | 0366 |
| Gerontology | 0351 |
| History | |
| General | 0578 |
| Ancient | 0579 |
| Medieval | 0581 |
| Modern | 0582 |
| Black | 0328 |
| African | 0331 |
| Asia, Australia and Oceania | 0332 |
| Canadian | 0334 |
| European | 0335 |
| Latin American | 0336 |
| Middle Eastern | 0333 |
| United States | 0337 |
| History of Science | 0585 |
| Law | 0398 |
| Political Science | |
| General | 0615 |
| International Law and Relations | 0616 |
| Public Administration | 0617 |
| Recreation | 0814 |
| Social Work | 0452 |
| Sociology | |
| General | 0626 |
| Criminology and Penology | 0627 |
| Demography | 0938 |
| Ethnic and Racial Studies | 0631 |
| Individual and Family Studies | 0628 |
| Industrial and Labor Relations | 0629 |
| Public and Social Welfare | 0630 |
| Social Structure and Development | 0700 |
| Theory and Methods | 0344 |
| Transportation | 0709 |
| Urban and Regional Planning | 0999 |
| Women's Studies | 0453 |

## THE SCIENCES AND ENGINEERING

### BIOLOGICAL SCIENCES

| | |
|---|---|
| Agriculture | |
| General | 0473 |
| Agronomy | 0285 |
| Animal Culture and Nutrition | 0475 |
| Animal Pathology | 0476 |
| Food Science and Technology | 0359 |
| Forestry and Wildlife | 0478 |
| Plant Culture | 0479 |
| Plant Pathology | 0480 |
| Plant Physiology | 0817 |
| Range Management | 0777 |
| Wood Technology | 0746 |
| Biology | |
| General | 0306 |
| Anatomy | 0287 |
| Biostatistics | 0308 |
| Botany | 0309 |
| Cell | 0379 |
| Ecology | 0329 |
| Entomology | 0353 |
| Genetics | 0369 |
| Limnology | 0793 |
| Microbiology | 0410 |
| Molecular | 0307 |
| Neuroscience | 0317 |
| Oceanography | 0416 |
| Physiology | 0433 |
| Radiation | 0821 |
| Veterinary Science | 0778 |
| Zoology | 0472 |
| Biophysics | |
| General | 0786 |
| Medical | 0760 |

### EARTH SCIENCES

| | |
|---|---|
| Biogeochemistry | 0425 |
| Geochemistry | 0996 |
| Geodesy | 0370 |
| Geology | 0372 |
| Geophysics | 0373 |
| Hydrology | 0388 |
| Mineralogy | 0411 |
| Paleobotany | 0345 |
| Paleoecology | 0426 |
| Paleontology | 0418 |
| Paleozoology | 0985 |
| Palynology | 0427 |
| Physical Geography | 0368 |
| Physical Oceanography | 0415 |

### HEALTH AND ENVIRONMENTAL SCIENCES

| | |
|---|---|
| Environmental Sciences | 0768 |
| Health Sciences | |
| General | 0566 |
| Audiology | 0300 |
| Chemotherapy | 0992 |
| Dentistry | 0567 |
| Education | 0350 |
| Hospital Management | 0769 |
| Human Development | 0758 |
| Immunology | 0982 |
| Medicine and Surgery | 0564 |
| Mental Health | 0347 |
| Nursing | 0569 |
| Nutrition | 0570 |
| Obstetrics and Gynecology | 0380 |
| Occupational Health and Therapy | 0354 |
| Ophthalmology | 0381 |
| Pathology | 0571 |
| Pharmacology | 0419 |
| Pharmacy | 0572 |
| Physical Therapy | 0382 |
| Public Health | 0573 |
| Radiology | 0574 |
| Recreation | 0575 |
| Speech Pathology | 0460 |
| Toxicology | 0383 |
| Home Economics | 0386 |

### PHYSICAL SCIENCES

#### Pure Sciences

| | |
|---|---|
| Chemistry | |
| General | 0485 |
| Agricultural | 0749 |
| Analytical | 0486 |
| Biochemistry | 0487 |
| Inorganic | 0488 |
| Nuclear | 0738 |
| Organic | 0490 |
| Pharmaceutical | 0491 |
| Physical | 0494 |
| Polymer | 0495 |
| Radiation | 0754 |
| Mathematics | 0405 |
| Physics | |
| General | 0605 |
| Acoustics | 0986 |
| Astronomy and Astrophysics | 0606 |
| Atmospheric Science | 0608 |
| Atomic | 0748 |
| Electronics and Electricity | 0607 |
| Elementary Particles and High Energy | 0798 |
| Fluid and Plasma | 0759 |
| Molecular | 0609 |
| Nuclear | 0610 |
| Optics | 0752 |
| Radiation | 0756 |
| Solid State | 0611 |
| Statistics | 0463 |

#### Applied Sciences

| | |
|---|---|
| Applied Mechanics | 0346 |
| Computer Science | 0984 |
| Engineering | |
| General | 0537 |
| Aerospace | 0538 |
| Agricultural | 0539 |
| Automotive | 0540 |
| Biomedical | 0541 |
| Chemical | 0542 |
| Civil | 0543 |
| Electronics and Electrical | 0544 |
| Heat and Thermodynamics | 0348 |
| Hydraulic | 0545 |
| Industrial | 0546 |
| Marine | 0547 |
| Materials Science | 0794 |
| Mechanical | 0548 |
| Metallurgy | 0743 |
| Mining | 0551 |
| Nuclear | 0552 |
| Packaging | 0549 |
| Petroleum | 0765 |
| Sanitary and Municipal | 0554 |
| System Science | 0790 |
| Geotechnology | 0428 |
| Operations Research | 0796 |
| Plastics Technology | 0795 |
| Textile Technology | 0994 |

### PSYCHOLOGY

| | |
|---|---|
| General | 0621 |
| Behavioral | 0384 |
| Clinical | 0622 |
| Developmental | 0620 |
| Experimental | 0623 |
| Industrial | 0624 |
| Personality | 0625 |
| Physiological | 0989 |
| Psychobiology | 0349 |
| Psychometrics | 0632 |
| Social | 0451 |

# NEURAL CODE EXCITED LINEAR PREDICTION FOR LOW POWER ADAPTIVE SPEECH CODING

BY

SRIGOURI KAMARSU

A Thesis submitted to the Faculty of Graduate Studies of the University of Manitoba in partial fulfillment of the requirements of the degree of

MASTER OF SCIENCE

# Abstract

Telephone bandwidth speech compression has been an ongoing area of research for several years. Numerous applications in telecommunications and storage have emerged in the past two decades. The ease of real-time implementation using single-chip digital signal processors has led to widespread implementation of speech coding algorithms in personal communication systems, for both wired and wireless communications.

Artificial neural networks have demonstrated their usefulness for clustering and pattern classification problems. The use of artificial neural learning algorithms for high quality speech compression at low bit-rates and low computation rates, potentially in non-stationary environments, are examined in this study. By using a class of artificial neural learning algorithms for determining the codebook in an adaptive vector quantizer, moderate sized codebooks can be generated that can be searched in parallel and are able to adapt to non-stationary environments. Unsupervised learning algorithms including frequency-sensitive competitive learning and Kohonen's self-organizing feature maps have been investigated for learning the codebook vectors. In contrast with earlier work, these learning rules have been employed in vector quantization of the residual signal after linear predictive coding and pitch prediction in a neural analogy to the code-excited linear prediction (CELP) approach. The performance of these algorithms for speaker-dependent and speaker-independent speech compression are presented. The results obtained by the present neural CELP method compare favorably with those of the CELP method, requiring reduced computational power and comparable bit-rates with a tolerable reduction in speech quality. The effects of limited precision on classification and learning in competitive learning algorithms for low power VLSI implementations are also explored in this thesis.

## Acknowledgements

I would like to thank my advisor Professor Howard Card, for his valuable guidance, continual support and encouragement which were invaluable in the completion of this thesis. I would also like to thank my colleagues who have been a valuable source of information and assistance throughout my graduate program. I would like to thank everyone who listened to my results for speech quality tests.

Financial support for this work was received from NSERC and from Micronet, a Network of Centres of Excellence.

I would also like to thank my parents Smt. Jyothi and Sri RamaRajeswara Rao, and Girish K. Bhat for their patience and moral support.

# Table of Contents

# List of Figures

# Chapter 1

# Introduction

Compression of telephone bandwidth speech has been an ongoing area of research for several years. In the past two decades, there has been an overwhelming interest in this field which has resulted in numerous applications in telecommunications and storage. Some applications of speech compression include wired and wireless telecommunication networks, consumer products for personal communication, and digital audio systems [1]. The ease of real-time implementation of speech-coding algorithms using single-chip digital signal processors has led to widespread implementation of speech algorithms in personal communication systems. Another new area of application is multimedia personal computing where voice storage is becoming a standard feature. Wideband audio coding for high-fidelity reproduction of voice and audio has also emerged as an important activity in the past decade. Applications of wideband audio coding lie largely with the broadcasting industry, motion picture industry, music industry and multimedia computing.

## 1.1. Speech Coding

Speech coding may be defined as a digital representation of speech that provides for efficient storage, transmission, recovery and perceptually faithful reconstruction of original speech. In other words, coding compresses speech for digital storage and expands or decompresses the stored data to reconstruct the original speech without significant loss in quality. Much of the research in speech and audio compression involves lossy compression techniques, where the original representation of the signal samples is never recovered exactly after decoding (decompression). At high compression ratios, there is a significant degradation in the reconstructed speech quality due to the lossy techniques used for compression. Hence, in digital processing of speech signals, there are two conflicting requirements viz. first, we need to achieve the lowest possible bit-rates (high compression ratios); second, we want to achieve this with minimum loss of speech quality. A third requirement for low-power (e.g. mobile) implementations is to reduce computational requirements of these algorithms. Satisfying these requirements is the purpose of the ongoing research in the speech coding community. Figure 1.1 shows the different bit-rates and an approximate designation of the speech quality that can be achieved at these bit-rates [2].

Speech coding algorithms can be divided into two main categories *waveform coders* and *vocoders or parametric coders.* In waveform coders, the data transmitted from encoder to the decoder specify a representation of the original speech as a waveform of amplitude versus time, so that the reproduced signal approximates the original waveform and, consequently, provides an approximate recreation of the original sound. In contrast, vocoders do not reproduce an approximation to the original waveform; instead, parameters that characterize individual sound segments are specified and transmitted to the decoder, which then reconstructs a new and different waveform that will have a similar sound. Often these parameters characterize the short-term spectrum of a sound or the parameters specify a mathematical model of human speech production for a particular sound. Vocoders operate at lower bit-rates than waveform coders but the reproduced speech quality, while intelligible, usually suffers from a loss of naturalness and some of the unique characteristics of an individual speaker are often lost.

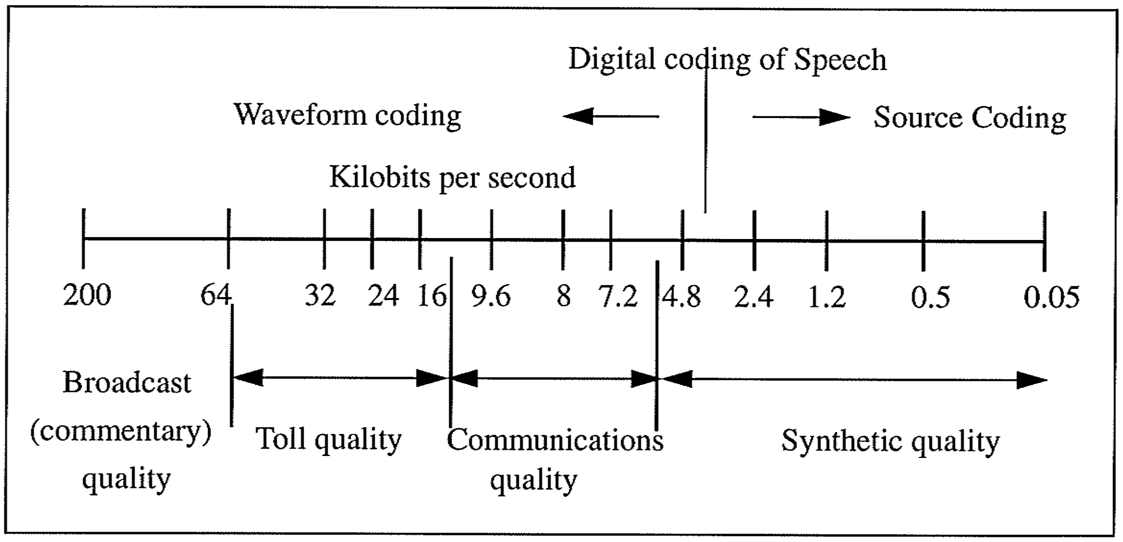

*Figure 1.1. Spectrum of speech coding transmission rates in nonlinear scale and associated quality.*

One of the most popular and notable waveform coding algorithms is code-excited linear prediction (CELP). Other algorithms in commercial use today are adaptive delta modulation (ADM), adaptive differential pulse code modulation (ADPCM), adaptive predictive coding (APC), multipulse linear predictive coding (MP-LPC) and regular pulse excitation (RPE) [3]. MP-LPC, RPE and CELP are sometimes viewed as "hybrid" algorithms because they borrow some features of vocoders, but they are usually classified as waveform coders. Various versions of CELP are available, but all algorithms in this family are based on linear prediction, analysis-by-synthesis, methods with a stochastically generated codebook for excitation. CELP produces bit-rates of 4800 bits per second (bps) with very good intelligibility and high quality, but the computational rates are usually very high.

Although vocoders were studied several decades ago, the most important vocoder is the linear predictive coding (LPC) vocoder method which is still extensively used in low bit-rate voice telephony. A version of the LPC vocoder has been used for many years as a U.S. Government standard, for secure voice communication. The bit-rate that can be obtained for LPC is 2400 bps with very low computational rates, but often, the reproduced speech sounds artificial or "unnatural" with a buzzy character and the identity of the speaker is hard to recognize.

Vector quantization (VQ) techniques can be used to obtain low bit-rate coding below 2400 bps with speech quality similar to that produced by CELP. In VQ techniques, a *k*-dimensional data vector is encoded using one of a finite set of *M* symbols. Each symbol is called a codeword and the set of all the codewords is known as the codebook. The VQ techniques usually require computationally extensive codebook learning and search. VQ problems can be mitigated by using various artificial neural network (ANN) algorithms. The use of ANN techniques for VQ of telephone bandwidth speech is the main focus of this thesis.

## 1.2. Artificial Neural Networks (ANN)

The human brain is superior to that of a digital computer at tasks like recognizing speech and faces. Other important features of the human brain are that its operation is inherently parallel and that it can adapt to a new environment by learning. Computers were built to perform complex mathematical computations at high speeds. Since they are designed to perform precise mathematical computations, they are not well suited to pattern classification problems. Since the brain handles the task of pattern classification well, one would hope that computers with architectures that attempt to mimic its operation, would be able to perform well on problems such as human perception. This area of study is called artificial neural networks (ANN), and has become an important field of research in the past decade. ANNs have been applied to a wide range of problems in data clustering and pattern recognition [4]. ANNs make use of parallelism and learning algorithms to solve complex tasks like pattern classification.

The basic form of an ANN architecture is shown in Figure 1.2. Each processing element in the ANN is called a neuron and the connections between the neurons are known as synapses. Each synapse has an associated weight $w_{ij}$ which represents the strength of the connection from neuron $j$ to neuron $i$. Typically, each neuron computes the output as a nonlinear transformation of the weighted sum of its inputs. ANNs learn to solve desired tasks by example. All the information necessary to solve the task must be provided by the examples, and this information is processed and stored in the weights. The network uses the example set to modify these weights so that it can learn to perform the desired task. The method by which the weights are modified is known as the learning algorithm. There are numerous available learning algorithms which can be broadly classified as either *supervised* or *unsupervised* learning algorithms.

In supervised learning algorithms, learning is performed on the basis of direct comparison of the output of the network to known correct answers. Since the correct answers are provided, this kind of learning is also called learning with a teacher. In unsupervised learning, there is no teacher or the correct answers for specific inputs are not available. The only available information is in the correlations of the input data or signals. The network must discover categories from these correlations and produce output signals corresponding to the input category.

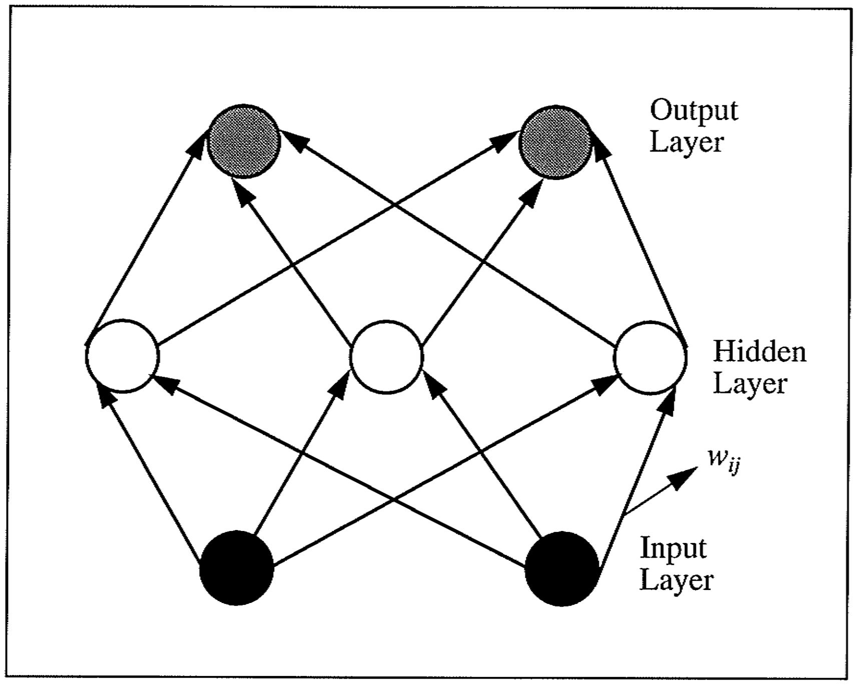

*Figure 1.2. Basic Neural Network Architecture*

Unsupervised learning can be used for data clustering and learning a codebook in VQ applications. An unsupervised learning algorithm known as competitive learning (CL) is particularly well suited for VQ applications and hence can be used in speech compression applications [5]. Various CL techniques are available which include soft competitive learning (SCL) and frequency-sensitive competitive learning (FSCL). Another technique called the Kohonen self-organizing feature map (KSFM) is used to enforce a topological relationship among the units in a network. Since ANNs are a highly parallel computer architecture, they offer the potential for real-time VQ applications as the codebooks can be searched in parallel. Most ANN training algorithms are adaptive and these ANN based

VQ design algorithms can be used to build adaptive vector quantizers which are crucial in applications where the source statistics are changing. The computational requirements of the learning algorithms can also be reduced if the learning algorithms can tolerate a lower bit precision. In that case, these algorithms can be implemented using low precision (and low power) digital signal processing (DSP) in real-time applications.

Normally in neural VQ, the speech signal is preprocessed to obtain LPC parameters which are then vector quantized. It has been shown in many previous studies of speech coding by non-neural techniques that in order to obtain high quality speech, it is necessary to transmit the prediction residual along with the LPC parameters. Transmission of the residual signal requires a significantly larger number of bits than those required to send the LPC parameters. Hence, it seems useful to quantize the residual signal rather than the LPC parameters. In contrast with earlier work, ANN learning rules are employed in VQ of the residual signal in this thesis to obtain low-bit rates and low computation rates for speech while retaining acceptable perceived speech quality. The effects of limited precision on learning and classification are included in this study. Preliminary results of this work have been published in [6], [7] and [8].

## 1.3. Summary

This chapter provided an overview of speech compression and the potential role of artificial neural networks. The importance of speech compression in communications and a summary of LPC, CELP and VQ compression schemes were presented. This was followed by an introduction to ANNs and the classification of ANN architectures based on the learning mechanisms. Finally, the use of ANNs in speech compression applications was introduced.

The remainder of this thesis is organized as follows: speech compression techniques, LPC, CELP and VQ will be discussed in detail in Chapter 2. In Chapter 3, algorithms for VQ of speech using ANN techniques will be discussed. These algorithms will be applied to an experimental speech compression task and the results obtained will be presented in Chapter 4. The effects of limited precision on learning and classification will also be discussed in Chapter 4. In Chapter 5 conclusions will be drawn based on simulations obtained in Chapter 4 and proposals for future work in this area will be presented.

# Chapter 2

# Speech Coding Techniques

Speech compression techniques have made rapid progress in the last decade. Linear predictive coding (LPC), code-excited linear prediction (CELP) and vector quantization (VQ) techniques have been used in numerous telecommunications applications. Artificial neural network (ANN) techniques can be used for VQ to potentially obtain low bit-rates, low computation rates and high quality speech compression. Before we can design such ANN algorithms for speech coding, an understanding of the non-neural speech compression techniques is necessary. Although speech compression techniques like adaptive differential pulse code modulation (ADPCM), subband coding (SBC) and adaptive transform coding (ATC) techniques have been primarily used in telecommunications applications, LPC, CELP and VQ techniques are of great importance in low bit-rate speech coding as lossy compression techniques. These techniques are designed to remove the redundant information from the speech signal. In this chapter, the LPC, CELP and VQ approaches will be discussed; the other techniques have been described in [1], [2], [3], [9] and [10].

## 2.1. Linear Predictive Coding (LPC)

LPC methods are among the most popular and powerful analysis techniques for processing speech. This approach has become the predominant technique for estimating speech parameters, e.g., pitch, formants, spectra, vocal tract area functions and for representing speech for low bit-rate transmission or storage. The importance of this method lies in its ability to provide accurate estimates of the speech parameters and in its relative speed of computation.

The underlying assumption in most speech processing schemes is that the properties of the speech signal change relatively slowly with time. This assumption leads to a variety of short-time processing methods in which short segments of speech signals are isolated and processed as if they were short segments from a sustained sound. The LPC method is accurate when it is applied to stationary signals i.e., signals whose behavior does not change with time. To perform LPC analysis, the speech signal is segmented into analysis frames which are quasi-stationary.

The basic idea of linear predictive analysis is that a speech sample can be approximated as a linear combination of past speech samples, thereby removing redundancies in the speech signal. The LPC method is based on the speech production model shown in Figure 2.1 This model removes the near sample correlations to a large extent. The system is excited by an impulse train or random noise based on whether the speech signal is voiced or unvoiced. If the signal is voiced then the pitch period is estimated. In LPC, the vocal tract is modelled as an all-pole recursive filter which is known as the formant filter. Incorporating a gain G, the filter can be expressed as

$$H(z) = \frac{G}{1 + a_1 z^{-1} + \ldots + a_{N_f} z^{-N_f}} = \frac{S(z)}{U(z)} \tag{2.1}$$

where $N_f$ is the order of the model and $a_k$ are the filter coefficients. If *s(n)* is the speech output of the model, and *u(n)* is the excitation input, the equation above can be written in the time domain as

$$s(n) = Gu(n) + \sum_{k=1}^{N_f} a_k s(n-k). \tag{2.2}$$

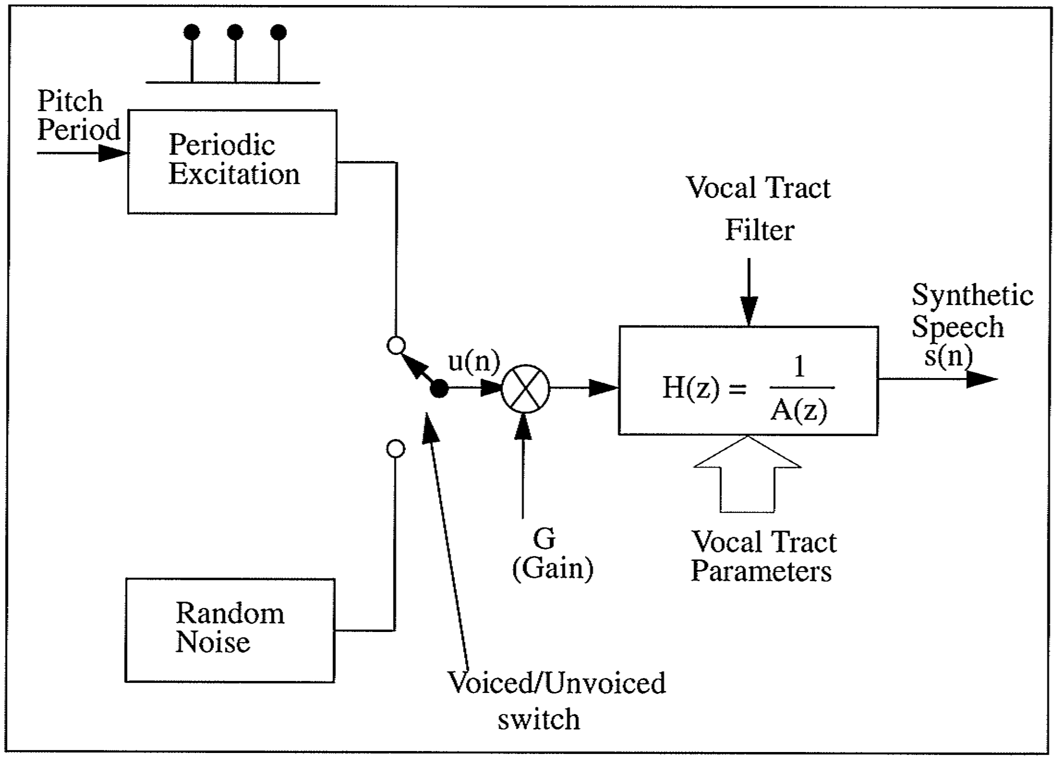

*Figure 2.1. Speech Production Model for LPC*

In other words, every speech sample is computed as a linear combination of the previous speech samples together with a contribution from the excitation. This formulation is also the reason for calling the method linear predictive coding. From (2.2), the formant filter or short-term filter has a transfer function

$$\frac{1}{H(z)} = A(z) = \sum_{k=1}^{N_f} a_k z^{-k}. \tag{2.3}$$

The LPC predictor coefficients $a_k$ are determined directly from the speech signal in such a manner that the mean-squared error of the short segment of speech is minimized. The speech signal is segmented into analysis frames by multiplying the speech signal *s(n)* by a window signal, *w(n),* which is zero outside the analysis frame. The most popular windowing function used is a Hamming window given by

$$\begin{aligned} w(n) &= 0.54 - 0.46 cos\left(\frac{2\pi n}{N}\right),\ 0 \le n \le N-1 \\ w(n) &= 0,\ otherwise. \end{aligned} \tag{2.4}$$

Here $N$ is the window length in samples and is generally taken in the range of 20-40 ms, with 30 ms being a typical value. Usually, the successive windows are chosen to overlap and the distance between the successive windows is called the frame period. Typical values of the frame period are 10-30 ms. The LPC parameters are estimated using the method given in Appendix A.1. One possibility for the selection of the excitation for voiced signals is given in [2]. The main features of the government standard LPC-10 algorithm are as follows: the frame length is 22.5 ms (180 samples for a sampling rate of 8 kHz) and the total bits per frame is 54, which gives a total data rate of 2400 bps. LPC analysis is performed pitch synchronously and a 10th order LPC model is used for voiced signals. For unvoiced signals, only a 4th order model is used since this is sufficient to determine the speech spectrum. The remaining bits are used for error protection. The speech quality of the synthesized speech has an undesirable mechanical quality and often the identity of the speaker is lost; but this method is used for applications like secure telecommunications where a low bit-rate is more important than the quality of the speech produced. The LPC method has the advantage of low computation rates requiring about 1.7 million instructions per second (MIPS)[1].

### 2.1.1. Pitch Prediction Algorithm

The determination of periodicity in a speech segment is important in many speech coding algorithms. This periodicity determines if the segment is voiced or unvoiced, and if voiced, it determines the fundamental period. For the LPC method, pitch prediction is necessary in order to generate the excitation as shown in Figure 2.1. For voiced signals (such as vowels) the speech signal demonstrates a fine structure arising from the quasi-periodic nature of the vocal tract, in addition to short-term correlations. The quasi-periodic nature of vibrations in the voiced speech remains to a large extent in the residual signal obtained after the short-term prediction (non-recursive inverse $A(z)$ of (2.3)) as shown in Figure 2.2. This long-term periodicity can be removed further by pitch prediction [11]. The pitch predictor has a smaller order than the formant filter and removes these distant sample correlations; this filter is also known as a long-term filter. The delays associated with these taps are grouped around the pitch lag value. The transfer function for a pitch predictor or the long-term predictor with 3 taps is

$$P(z) = \beta_1 z^{-M} + \beta_2 z^{-(M+1)} + \beta_3 z^{-(M+2)} \tag{2.5}$$

where $M$ is the pitch period.

1.Multiply, add, multiply-accumulate and compare instructions

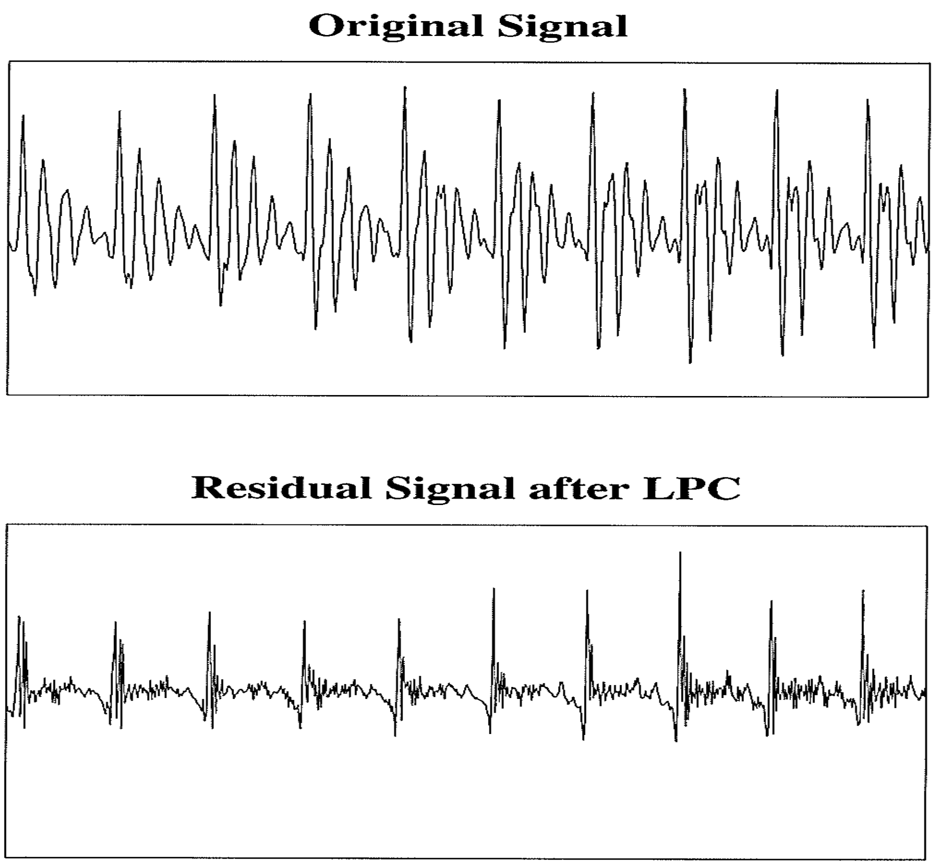

*Figure 2.2. Original Speech Signal and the Residual Signal Obtained after LPC*

In order to obtain the pitch predictor coefficients $\beta_i$, the pitch period or pitch lag value $M$ has to be determined. When the formant or LPC analysis is performed first, the near sample based redundancies have been removed to a large extent before pitch analysis. The conventional predictor configuration uses a cascade of a formant predictor and a pitch predictor. The value of the pitch period is chosen so as to maximize the correlation of the residual after the formant prediction. The normalized correlation array is given by

$$\sum_{m=M}^{M-1+N_p} \frac{\phi^2(0,m)}{\phi(m,m)} \tag{2.6}$$

where

$$\phi(i,j) = \sum_{n=0}^{N-1} d(n-i)\, d(n-j) \tag{2.7}$$

and *d(n)* is the residual signal after formant prediction which is input to the pitch predictor. The value of *M* that maximizes the quantity in (2.6) is chosen as the pitch period. The pitch prediction parameters can be estimated using the method in Appendix A.2.

## 2.2. Vector Quantization

Shannon's rate distortion theory states that better performance can be achieved by coding vectors instead of scalars. Vector quantization is used to compress speech and image signals. VQ is often used for speech compression with the LPC model to achieve low bit-rate coding. Recently, with the emergence of new and efficient methods of encoding high-dimensionality data vectors, VQ became associated with high-quality speech coding at low rates.

A vector quantizer maps each input vector in a $k$-dimensional Euclidean space $\boldsymbol{R}^k$, into one of the finite number of representative vectors in $\boldsymbol{R}^k$. The set of representative vectors is called a codebook, and each representative vector is called a code vector or a reference vector. A general VQ scheme is shown in Figure 2.3. An optimal vector quantizer minimizes the average distortion over all the representative vectors. The lower the distortion caused by reproducing an input vector with the corresponding representative vector, the higher is the performance of the VQ. A complete review of VQ is presented in [12] and [13].

A number of different performance criteria can be used to determine an optimal codebook. In speech applications, the objective is to minimize the overall distortion due to VQ in the reconstructed signal. Thus the design criterion used to design an optimal codebook of a given size is the minimization of the average distortion in encoding vectors using the codebook. Another possible criterion is to maximize the entropy of the codebook, i.e., ensure that each of the codewords is used equally or frequently in encoding the data. The idea here is to ensure that all the codewords are doing their fair share in representing the entire input data. For the case when $N$ is fixed and $k$ is very large, it was shown that the codebook that maximizes the entropy also minimizes the expected distortion [14].

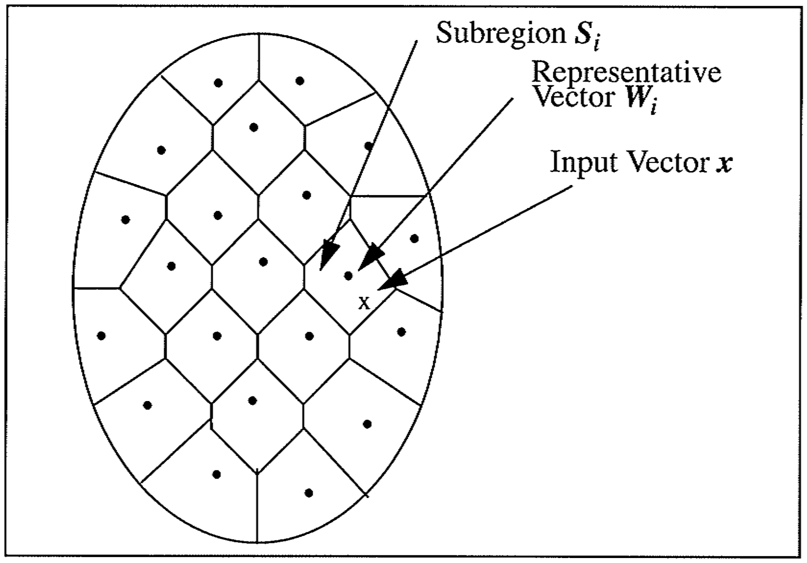

*Figure 2.3. Vector Quantization Scheme*

Given the performance criterion, the VQ design process involves the determination of the codebook that is optimal with respect to the criterion. In general, this requires knowing the probability distribution of the input data. Typically, this distribution is not known and the codebook is constructed through a process called *training*. During training, a set of data vectors that is representative of the data that will be encountered in practice is used to learn an optimal codebook. During the training process, a distortion measure *d(x,w)* is typically used to determine which data points are considered as being in the same region as the representative vector. The distortion measure can be viewed as the cost of representing input $\boldsymbol{x}$ by the codevector $\boldsymbol{w}$. By determining which training data vectors lie in the same region, the $k$ dimensional data space is partitioned into cells $S_i$ (See Figure 2.3). All of the input vectors that fall into a particular cell are mapped into a single, common reproductive vector $\boldsymbol{w}_i$. If the cells are partitioned according to a minimum distortion rule, then the partition is referred to as a Voronoi or Dirichlet partition. The most common distortion measure is the Euclidean distance given by

$$d(\boldsymbol{x}, \boldsymbol{w}) = \|\boldsymbol{x} - \boldsymbol{w}\|^2 = \sum_{j=1}^{k} |x_j - w_j|^2. \quad (2.8)$$

Another distortion measure called the Itakura-Saito distortion measure is frequently used in speech coding applications, and is a spectral distortion measure.

The training process which is used to build the codebook can be summarized as follows: Each of the data vectors is compared to each of the codewords and the corresponding distortion is calculated. The codeword that most closely matches the data vector, i.e., the reproduction vector which represents the input vector with minimum distortion is selected and the codeword is updated to reflect the inclusion of this new data vector in its partition. Various VQ algorithms are available and have been summarized in [12], [13] and [14].

In [15] a large reduction in coding rate was achieved through short-term temporal compression of the speech followed by VQ. The VQ is applied to the temporal decomposition output to implement coders operating in a range of 450-600 bps with natural sounding speech output. Hence VQ can be used effectively for very low rate speech coding by parameterizing the speech signal by quantizing these parameters effectively. By employing VQ for LPC parameters, the bit-rate can be reduced to 800 bps. However, VQ methods require high computational power for training and searching the codebooks.

## 2.3. Code Excited Linear Prediction (CELP)

Predictive coders reduce bit-rate by removing redundancies in the speech signals by linear prediction and then transmitting the quantized parameters of the predictor as well as the quantized residual. It is however very difficult to quantize the prediction residual accurately at rates less than 2-3 bits/sample. For low-rate, high-quality speech coding, a more efficient representation of the excitation sequence is required. This problem was addressed in [16] in which it was suggested that high-quality speech at low bit-rates may be produced by coding of Gaussian excitation sequences in conjunction with analysis-by-synthesis linear prediction and perceptual weighting. A novel excitation scheme for analysis-by-synthesis CELP was proposed in [17]. This analysis-by-synthesis CELP coder is shown in Figure 2.4.

The CELP coder contains two time-varying linear recursive filters each with a predictor in its feedback loop. The first feedback loop includes the LPC analysis filter with a transfer function given in (2.3) and the second feedback loop includes the pitch predictor with a transfer function given in (2.5). The excitation is encoded using a codebook of Gaussian sequences. The codebook contains 1024 vectors and each vector is 5 ms long. A gain factor $g$ scales the excitation vector and the excitation samples are filtered by the long-term (pitch) and short-term (LPC) synthesis filters.The optimum vector is selected such that the perceptually weighted mean-squared-error is minimized. The perceptual weighting filter $W(z)$ is applied directly to the input speech signal $s(n)$ and the synthetic output $\hat{s}(n)$ and the resulting error is minimized to obtain the excitation vector and the gain.

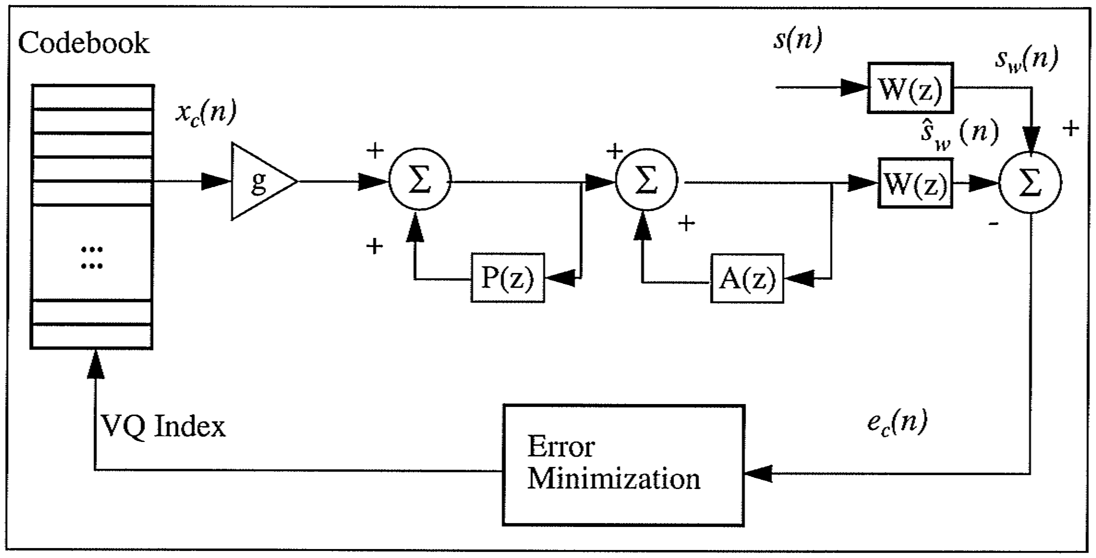

*Figure 2.4. Analysis-by-Synthesis CELP*

A 4800 bps CELP algorithm has been adopted by the Department of Defense for possible use in a third-generation secure telephone unit [18]. The synthesis configuration for Federal Standard 1016 (FS1016) is shown in Figure 2.5. The speech signal is sampled at 8 kHz and segmented into frames of 30 ms. Each frame is segmented in subframes of 7.5 ms. The excitation in this CELP is formed by combining vectors from an adaptive and a stochastic codebook with gains $g_a$ and $g_s$ respectively. The excitations are selected in every subframe by minimizing the perceptually weighted error measure.

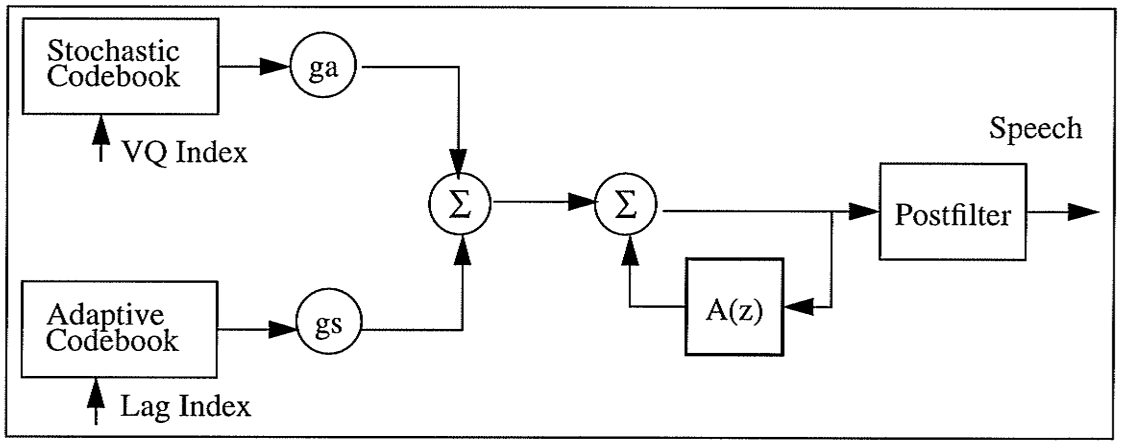

*Figure 2.5. FS1016 CELP Synthesis [9].*

The codebooks are searched sequentially starting with the adaptive codebook. The term adaptive codebook is used because the long-term predictor (LTP) lag search can be viewed as an adaptive codebook search where the codebook is defined by previous excitation sequences and the lag determines the specific vector. The adaptive codebooks contains the history of past excitation signals and the LTP lag search is carried over 128 integer (20-147) and 128 noninteger delays. A subset of lags is searched in even subframes to reduce the computational complexity. The stochastic codebook contains 512 sparse and overlapping code vectors. Each code vector consists of 60 samples and each sample is ternary valued (1, 0, -1) to allow for fast convolution. Ten short-term predictor coefficients $A(z)$ are encoded as line spectrum pairs (LSP) [2] on a frame-by-frame basis. Subframe LSP's are obtained by applying linear interpolation of frame LSP's. A short-term pole-zero postfilter is also part of the standard.

The characteristics of the CELP coder are given in Table 2.1. One bit per frame is used for synchronization, 4 bits per frame for forward error correction and 1 bit per frame for future expansion giving a total bit-rate of 4800 bps. The computational requirements for CELP are 12.6 million instructions per second (MIPS). CELPs major computational requirements are dominated by the transmitter's codebook searches. CELP coders do not exhibit the usual vocoder problems in background noise because they use a more sophisticated excitation model than the classical vocoder's pitch and voicing. Background noise, including multiple speakers, is faithfully reproduced. Informal listening tests indicate that the 4800 bps CELP coder's speech intelligibility and quality are comparable to 32,000 bps continuously variable slope delta (CVSD) coding.

| | Linear Predictor | Adaptive Codebook | Stochastic Codebook |
|---|---|---|---|
| Update | 30 ms | 7.5 ms | 7.5 ms |
| Parameters | 10 LSP's | 1 gain, 1 delay, 256 codewords | 1 gain, 1 index, 512 codewords |
| Bits per frame | 34 | index: 8+6+8+6<br>±gain: 5x4 | index: 9x4<br>gain: 5x4 |
| Rate | 1133.33 bps | 1600 bps | 1866.67 bps |

*Table 2.1. CELP characteristics [18]*

## 2.4. Summary

This chapter provided an overview of LPC, VQ and CELP speech coding techniques. LPC provided an efficient method of estimating the speech signal as a linear combination of past speech samples. It removed short-term correlations in the speech signal to a large extent. The LPC filter was excited by a impulse train or random noise based on whether the speech signal was voiced or unvoiced. The pitch period was estimated if the speech signal is voiced. The pitch filter was used to remove the long-term correlations in the residual signal after LPC. LPC produced bit-rates of 2400 bps with the reproduced speech quality having an artificial or buzzy character. However, the computational requirements were very low and at 1.7 MIPS.

The vector quantization method was then introduced and the advantages of the VQ method were presented. By using a suitable learning algorithm, VQ produced a codebook which is a finite set of representative vectors for a set of input vectors. The index of the best matching representative vector for any input vector is transmitted instead of the whole vector, thereby reducing the bit-rates. VQ can be potentially used to obtain bit-rates as low as 450 bps with good speech quality. However, the VQ technique suffers from increased computational requirements for the codebook learning and search.

CELP uses the linear-predictive-analysis-by-synthesis method along with two codebooks for LTP lag and excitation. The Federal Standard CELP 1016 was then described. CELP produces very high speech quality even in noisy environments and requires a bit-rate of 4800 bps. The computational requirement is 12.6 MIPS requiring most of this relatively large computational power for the codebook search.

In the next chapter, artificial neural learning algorithms for VQ will be discussed. These algorithms can be potentially used to solve the VQ problems to produce simultaneously low bit-rate and low computation rate speech compression.

# Chapter 3

# Artificial Neural Learning for Vector Quantization (VQ)

## 3.1. Introduction

A number of studies have reported the use of artificial neural network (ANN) algorithms in VQ encoding and codebook design [5], [19], [20] and [21]. ANNs are highly parallel computing structures consisting of a number of simple processing units called neural units, each with a set of interconnections (weights) from the other units and from the inputs to the network. ANNs are designed to perform well on tasks such as human perception, where they adapt to a new environment by learning. ANNs learn to perform any desired task by training on a set of examples. The set of examples consists of sample inputs that are representative of the task the network must perform. The network learns from these examples and stores the information in the weights. The values of the weights are modified by a suitable learning algorithm. The network also has the ability to generalize upon the training cases. The ANN learning algorithms most suitable for VQ of speech are the unsupervised learning methods including competitive learning and feature mapping.

An unsupervised learning network has a set of inputs and outputs with no external feedback to say what the outputs should be. The network must discover on its own patterns, features, regularities, correlations or categories in the input data and output an appropriate code. The network must derive the error and the necessary weight modifications directly from the learning rule, and from the statistics of the training data. Some of the features that might be represented by the outputs of such networks are familiarity, principal components, geometric clustering, prototyping, encoding and feature mapping. Principal component analysis can be used for dimensionality reduction of the data. The encoding problem can also be performed using clustering or feature mapping which is often called vector quantization (VQ). One of the major advantages of formulating the VQ problem in terms of ANN algorithms is that a large number of these algorithms can be applied to the VQ task. Most ANN algorithms are adaptive and allow for the possibility of training the VQ on-line even in a non-stationary environment.

Most unsupervised learning networks consist of only a single layer with inputs and outputs. The number of output units is usually smaller than the number of inputs except in the case of feature mapping. Learning techniques are either based on connections that learn using a modified Hebb rule (e.g. principal component analysis) [4] or competitive learning. In this chapter, competitive learning algorithms used for VQ of speech will be discussed. Another unsupervised learning algorithm known as the Kohonen self-organizing feature maps (KSFM) is also discussed. A relevant measure of the quality of the codebooks obtained by these algorithms is the codebook entropy. The codebook entropy $E$ is given by

$$E = -\sum_{i=1}^{N} p_i \, log_2(p_i) \tag{3.1}$$

where $N$ is the number of codewords in the codebook and $p_i$ is the relative frequency with which codeword $i$ is used in encoding the data set. In the ideal case, if all the codeword entries are equally utilized, the value of the entropy would be $log_2(N)$. The closer the entropy to this ideal value, the more uniform is the codeword usage implying a better codebook performance.

## 3.2. Competitive Learning (CL)

The aim of CL networks is to cluster or categorize the input data. The clusters must be found by the network itself from the correlations of the input data. Inputs which are similar should be classified as being in the same category and so should most strongly activate the same output unit. Categorization or clustering can be used for data encoding and compression through VQ, which has applications in speech coding. (Chapter 2.2). In CL, only one output unit is on at any time and the output units compete for being the active output. This network is therefore called a *winner-take-all* (WTA) network.

A simple CL network is shown in Figure 3.1. The connections shown with open arrows are inhibitory and the remainder are excitatory. There is a single layer of output units $O_i$, each fully connected to a set of inputs $x_i$ via excitatory connections $w_{ij}$. The connection of the unit to itself is excitatory, which will help the neuron to reinforce its output. The connections to other output units are inhibitory and attempt to suppress the output of other neurons. The inputs and outputs are usually binary 0/1. The output which is most active is known as the winner. The winner is normally the unit with the largest net input

$$h_i = \sum_j w_{ij} x_j = \boldsymbol{w}_i \cdot \boldsymbol{x} \tag{3.2}$$

for the current input vector $\boldsymbol{x}$. Thus,

$$\boldsymbol{w}_{i*} \cdot \boldsymbol{x} \geq \boldsymbol{w}_i \cdot \boldsymbol{x} \quad \textit{(for all i)} \tag{3.3}$$

defines the winning unit $i^*$ with the output of the corresponding winner $O_{i*} = 1$ and all other outputs equal to 0. If the inputs and the weights for each unit are normalized, so that $|w_i| = 1$ for all $i$, then (3.3) is equivalent to

$$|\boldsymbol{w}_{i*} - \boldsymbol{x}| \leq |\boldsymbol{w}_i - \boldsymbol{x}| \quad \textit{(for all i).} \tag{3.4}$$

This states that the winner is the unit with normalized weight vector $\boldsymbol{w}$ closest to the input vector $\boldsymbol{x}$.

The algorithm starts with small random values for the weights. Then a set of input patterns $\boldsymbol{x}^{\mu}$ is applied to the network in random order and the distortion $d(\boldsymbol{x},\boldsymbol{w}_i)$ (Euclidean distortion (2.8) or any other suitable measure) is calculated. The corresponding winner $i^*$ is selected as the unit with the lowest distortion for each input, and the weight vector corresponding to the winning unit is moved closer (during learning) to the input pattern $\boldsymbol{x}^{\mu}$. Figure 3.2 shows a competitive learning network with the input vectors represented by dots and the weight vectors represented by crosses. The figure shows that the output units have discovered a cluster of inputs and have moved to its center of gravity after learning.

There are several ways of updating the winning unit so that it moves closer to the input vector and is more likely to win on that input in the future. In simple competitive learning, the weight of the winning unit is updated using

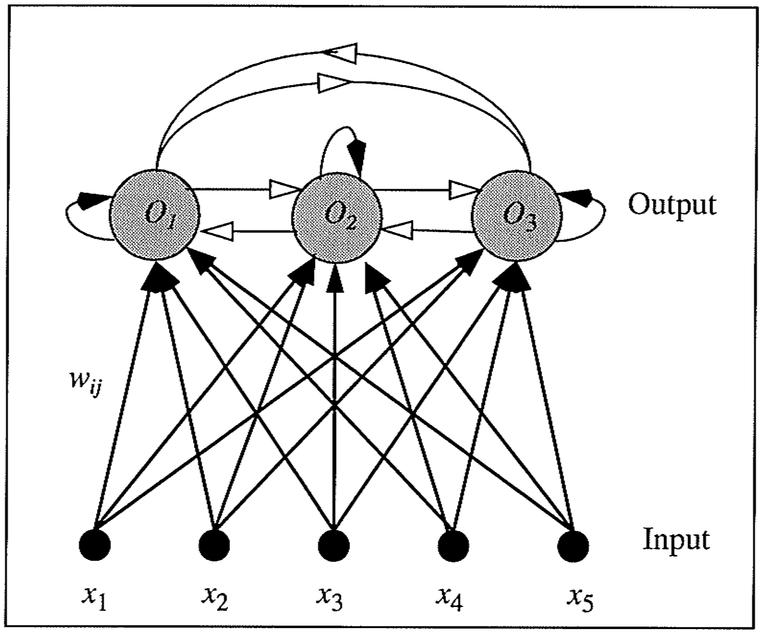

*Figure 3.1. A Simple Competitive Learning Network*

$$w_{i^*j}(n+1) = w_{i^*j}(n) + \varepsilon\left(x_j^{\mu} - w_{i^*j}(n)\right) \tag{3.5}$$

where $\boldsymbol{x}^{\mu}$ is the set of input patterns, ε is the learning rate, $n$ is the training iteration and the weight of the winning unit $\boldsymbol{w}_{i*}$ moves directly towards the input pattern $\boldsymbol{x}^{\mu}$. This kind of CL is known as hard competitive learning since only the winning unit is updated.

There is a problem with this learning method. If the weight vector $\boldsymbol{w}_i$ of a particular unit is initially far removed from any input vector, then it may never win and therefore never learn. These units are termed as dead units. The consequence of having dead units is that it reduces the codebook entropy value defined in (3.1). There are several ways of preventing dead units and increasing the value of the entropy. One of the methods is called frequency-sensitive competitive learning (FSCL) which will be discussed in the next section. There are also soft competitive learning algorithms (SCL) in which all units are updated in proportion to their current responsibility for the input pattern [22]. The hard CL network requires low computational requirements since only the winner is updated.

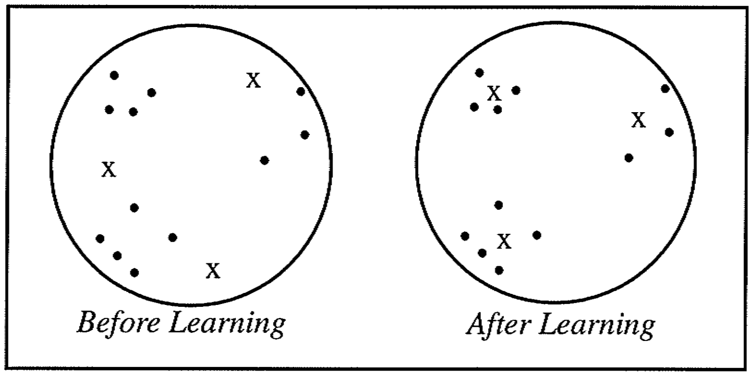

*Figure 3.2. Competitive Learning*

## 3.3. Frequency-Sensitive Competitive Learning (FSCL)

FSCL for VQ of speech was proposed by Krishnamurthy et al [5] and [19]. The motivation for the FSCL network is to overcome the limitations of the CL network while retaining its computational advantages. Since the main problem of hard CL is dead units or underutilized units, FSCL keeps count of how frequently each unit wins. This information is used when the winner is updated. This mechanism is sometimes called a conscience; frequent winners penalize themselves by increasing their distortion measure, thereby enhance the opportunities for infrequent winning units to win in the competition.

A similar conscience mechanism approach was first described in [23] which suggested the use of a threshold to be subtracted from the net input and an adjustment to the thresholds to encourage frequently losing units to win. The FSCL method applies these models to the VQ problem. In the FSCL network, each unit incorporates a count of the number of times a unit wins. The distortion measure used to determine the winner is modified to include this count. If $u_i$ denotes the number of times a unit wins the competition and $d(\boldsymbol{x},\boldsymbol{w}_i)$ is the distortion used to obtain the winner, the modified distortion measure for the training process is defined as:

$$d^* (\boldsymbol{x}, \boldsymbol{w}_i) = d(\boldsymbol{x}, \boldsymbol{w}_i(n)) \times \mathcal{F}(u_i) \tag{3.6}$$

where $\mathcal{F}$ is a nondecreasing function called the *fairness* function. The fairness function introduces a count-dependent weighting to the distortion measure. This function provides a way to control the behavior of the FSCL training procedure. For example, choosing $\mathcal{F}(u_i) = 1$ reduces the FSCL to the standard CL algorithm. Two possible choices for the fairness function are $\mathcal{F}(u_i) = u_i$ or $\mathcal{F}(u_i) = u_i^k$ where $k = \beta exp(-n/T)$ where $\beta$ and $T$ are constants and $n$ is the training iteration number. If a given neural unit wins the competition frequently during the learning process, its count $u_i$ increases and so does its distortion in (3.6), since $\mathcal{F}$ is nondecreasing. The increased distortion value reduces the likelihood that this unit will be the winner in the future, giving other units with a lower count value a chance to win the competition. The winning unit at each iteration is chosen as the unit with minimum distortion $d^*$. After the winner is obtained, its weight is updated using (3.5) and the count $u_i$ of the winner is incremented. This method encourages all neural units to participate equally in the competition encouraging uniform codeword usage and hence increased codebook entropy. The computational requirements for FSCL are slightly greater than those of CL algorithm due to the use of the modified distortion measure (3.6). The memory requirements during the training process are also increased to store the winning frequency of each neural unit.

## 3.4. Kohonen Self-Organizing Feature Maps (KSFM)

The CL and FSCL networks pay little attention to the geometrical arrangement of the output units. A few network architectures convey some information based on the location of the winning output unit, with nearby output units corresponding to nearby input patterns. A network that performs such a mapping is called a feature map. The feature map preserves the topology and the neighborhood relations from the space of possible inputs to the line or plane of the output units. There are a number of ways of designing unsupervised learning networks that self-organize into a feature map. The KSFM algorithm introduced by Kohonen [24] uses the CL algorithm to obtain the winner but the weight update rule is modified to preserve the topology and neighborhood relations.

KSFM uses the architecture shown in Figure 3.3 which is a fully connected network. There are $N$ continuous valued inputs $x_1$ to $x_N$. The output units $O_i$ are arranged in a one or two-dimensional array and are fully connected via $w_{ij}$ to the inputs. A CL rule is used to choose the winning unit $i^*$ with the weight vector $\boldsymbol{w}_{i*}$ closest to the input vector $\boldsymbol{x}$. The winning unit can also be found as the unit with the lowest distortion $d(\boldsymbol{x},\boldsymbol{w}_i)$.

In order to incorporate a topological relationship among the output units, the weight update equation is modified as

$$w_{ij}(n+1) = w_{ij}(n) + \varepsilon(n)\,\Lambda(i, i^*)\left(x_j^{\mu} - w_{ij}\right) \tag{3.7}$$

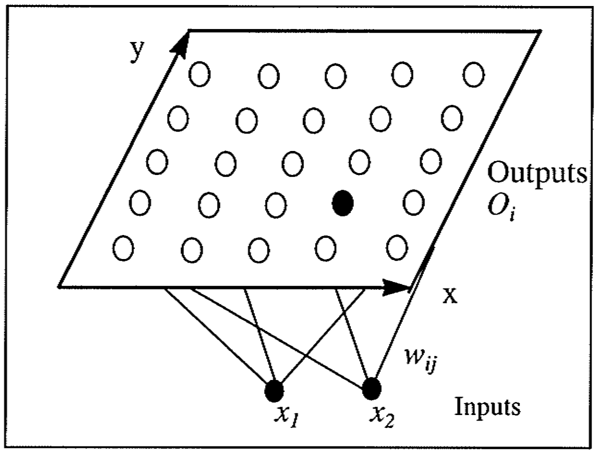

*Figure 3.3. Feature Mapping Architecture*

for all $i$ and $j$. The neighborhood function $\Lambda(i, i^*)$ is 1 for $i = i^*$ and $\varepsilon(n)$ is a time-varying learning rate. The neighborhood function has to be changed dynamically during learning for the algorithm to be useful. In earlier stages of training, a wide neighborhood $\Lambda(i, i^*)$ and large learning rate $\varepsilon$ are used. As the training progress both these values should be gradually reduced. This allows the network to organize its elastic net rapidly during the period when a large neighborhood and learning rate are used and then refine it slowly with respect to the input pattern as the neighborhood and the learning rate are decreased. A typical choice for the neighborhood function is

$$\Lambda(i, i^*) = exp\left(-|r_i - r_{i^*}|/2\sigma^2\right) \tag{3.8}$$

where $\sigma$ is a width parameter that is gradually reduced and $|r - r_{i^*}|$ is the distance between the units $i$ and $i^*$ in the output array.

By the use of neighborhoods, the KSFM network overcomes the problem of underutilized units giving a larger entropy than a hard CL network. One drawback of KSFM compared to the CL and FSCL networks is the additional computation involved during training. This additional computation arises from both the calculation of the neighborhood of the winning unit as well as from updating all the units in the neighborhood.

## 3.5. Summary

In this chapter, several artificial neural learning methods for the adaptive VQ of speech were discussed. Artificial neural learning methods can be potentially used to learn the codebook in a VQ. ANNs are adaptive and hence allow for the possibility of training the VQ on-line, and for their employment in a nonstationary environment. Three unsupervised learning methods to train the codebook in a VQ were discussed in this chapter. Firstly, the competitive learning algorithm was introduced which is a winner-take-all network. An input vector was applied to the network and the output closest to the input unit was chosen as the winner. The weight of the winner was updated so as to move it closer to the input. The CL network introduced dead-units or underutilized units. The FSCL network which has a conscience mechanism was then introduced to avoid the underutilization of codewords and to increase the codebook entropy. The conscience mechanism allowed the units that win often to increase their distortion accordingly to permit other infrequently winning units to participate in the competition. The third algorithm that was described was the KSFM in which the network incorporated neighborhood and topological relationships among the output units. This was done by choosing a neighborhood function and by updating a neighborhood of the winning unit. Initially a large neighborhood was chosen so as allow the network to organize itself rapidly and as the neighborhood decreased the network would fine tune its weight values.

Among the three learning algorithms, CL has the lowest computational requirements and KSFM has the highest. In the next chapter, the learning methods discussed in this chapter will be used to vector quantize the excitation signal for speech compression.

# Chapter 4

# Neural VQ for Speech Compression

In the earlier chapters, LPC, VQ and CELP speech coding techniques were discussed. LPC requires low bit-rates and computational power, but the reproduced speech sounds artificial or unnatural with a buzzy character. CELP uses a computationally expensive algorithm to obtain good intelligibility and excellent speech quality with bit-rates higher than LPC. VQ can be used potentially to reduce the bit-rate below LPC with speech quality similar to CELP but usually requires computationally expensive codebook search. By employing ANN algorithms for VQ, one can achieve moderate sized codebooks that can be searched in parallel and are able to adapt to non-stationary environments with bit-rates lower than CELP. This means that ANN algorithms can obtain a compromise solution with low bit-rate, moderate speech quality and moderate computation-rate. Computational requirements may be further reducible through the use of restricted bit precision. In this chapter, the performance of the unsupervised learning algorithms discussed in Chapter 3 for speaker-dependent and speaker-independent speech compression will be examined and the results will be compared to those obtained by using CELP. The results obtained by using limited precision in both the classification and learning computations of these algorithms are also examined.

For all the simulations presented in the remainder of this thesis, software was written in C and C++ programming languages. Various modules were developed to implement the analysis and synthesis of the speech signal and for generating the training data. A public domain C software package known as *som_pak* [25] which implements the KSFM algorithm was obtained from Finland. This software was extended to implement the FSCL algorithm. The distortion measure was the Euclidean function (2.8) and the fairness function for the FSCL algorithm was $\mathcal{F}(u_i) = u_i$ for all the simulations. Another public domain C program which implements the FS1016 CELP [26] was obtained from the Department of US Defense to compare the results of this study.

## 4.1. Neural VQ of LPC Parameters

In earlier studies [5] and [19], the speech signal was preprocessed to obtain the LPC parameters (Section 2.1) which were then vector quantized using artificial neural learning algorithms. These experiments were repeated and included in this study. The training data was obtained from the TIMIT database [27] for both speaker-dependent and speaker-independent speech compression. The database consists of speech segments from 630 different speakers from 8 major dialects of American English, each speaking 10 phonetically rich sentences. The training data consisted of 6000 data vectors of LPC autocorrelation coefficients of 10 different sentences from a single male speaker. The data was downsampled to 8 kHz and the LPC autocorrelation coefficients were extracted from a window size of 40 ms (320 samples). The autocorrelation coefficients are vector quantized using the FSCL and KSFM learning algorithms described in Chapter 3.

The experiments were conducted for a codebook size of 128 codewords. Figure 4.1 and Figure 4.2 show the codeword utilization for the codebooks learned using FSCL and KSFM. The neighborhood for the KSFM was chosen to be a $16 \times 8$ rectangular grid for a codebook of size 128.

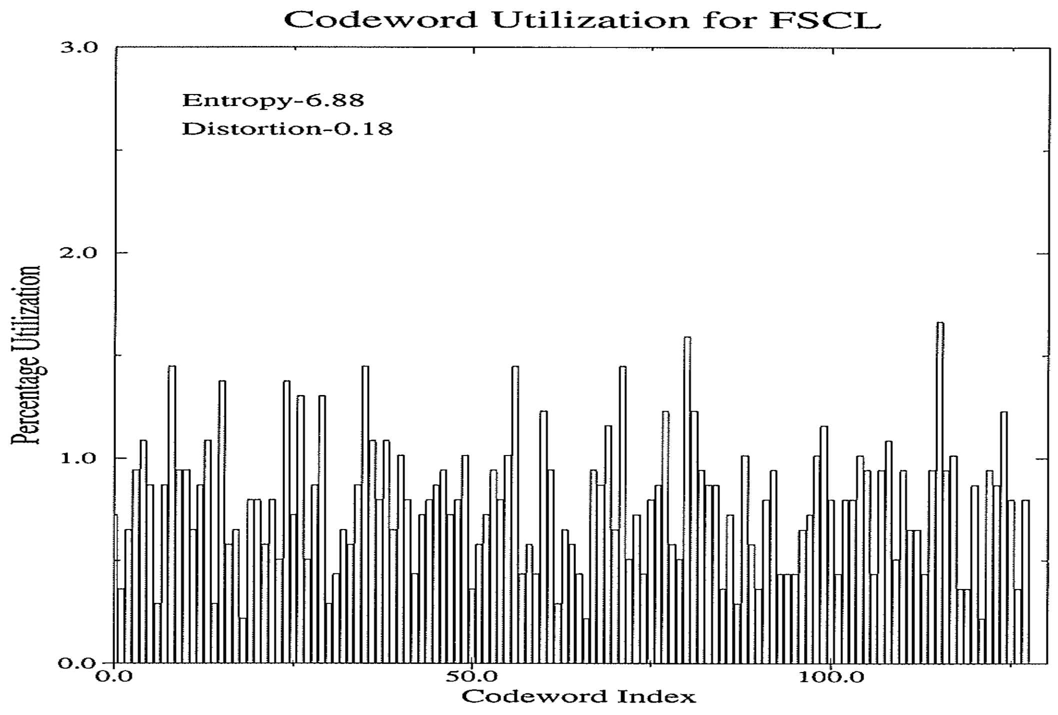

*Figure 4.1. Codeword Utilization Using FSCL Algorithm*

A comparison of these figures shows that the codeword utilization is more uniform using the FSCL technique as compared to the KSFM method. The entropy value for FSCL was 6.89 and for KSFM was 6.79, the ideal value being 7 which shows that the fairness function in FSCL encourages a uniform codeword utilization. A comparison of the distortion values shows that the FSCL also gives a lower distortion than KSFM method. Informal listening tests revealed that the neural net vector quantizer preserves the shape of the LPC filter and the LPC synthesized speech was identical to the original speech when the true residual signal was used to excite the inverse LPC filter. In [19] speaker-independent speech coding experiments were also performed. For a codebook size of 128, the speaker-independent experiments introduces some additional distortion in VQ and a larger codebook has to be used.

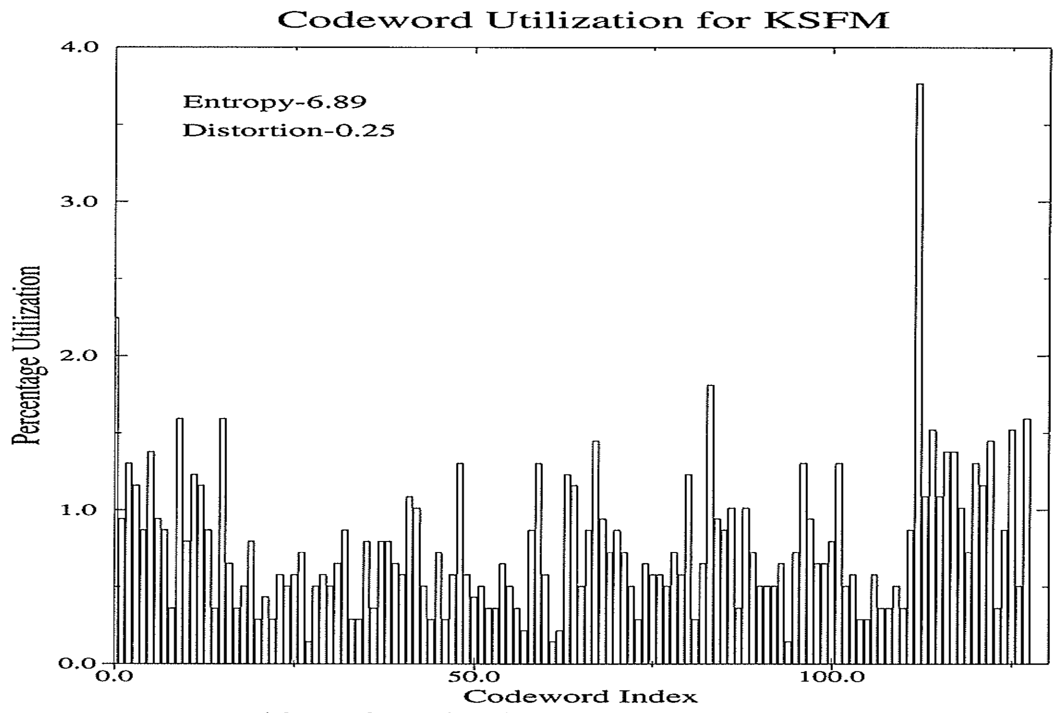

*Figure 4.2. Codeword Utilization Using KSFM Algorithm*

## 4.2. Neural CELP

It has been shown in many previous studies of speech compression by non-neural techniques that, in order to obtain high speech quality, it is necessary to transmit the prediction residual signal along with the LPC parameters. It is, however, very difficult to quantize the prediction residual accurately at rates less than 2-3 bits/sample. Below these rates, the quantization error starts showing significant correlation with the speech signal.

As a result, the predictor is no longer optimal and the coding gain drops. CELP coding gets around this problem by using an analysis-by-synthesis loop to minimize the frequency-weighted mean-squared error between the coder input and the decoder output. In neural VQ of speech, the prediction residual has to be transmitted and the transmission of the prediction residual requires a significantly larger number of bits than those required to transmit the LPC parameters. Hence, it seems useful to vector quantize the residual signal rather than vector quantizing the LPC parameters as in the previous section. The amount of information in the LPC prediction residual can be reduced by performing pitch prediction on the LPC prediction residual which removes redundant pitch information. The residual signal obtained after LPC and pitch analysis can be vector quantized in a neural analogy to the CELP approach.

In the neural CELP approach, the speech signal is preprocessed to obtain 10 LPC parameters. The residual signal obtained after inverse LPC analysis is used to extract the pitch period and 3 pitch prediction coefficients. The final residual after the removal of the pitch information is then vector quantized using the artificial neural learning methods discussed in Chapter 3. The codebook can be learned off-line, and is available both at the transmitter and the receiver. At the transmitter, the original speech signal is filtered through the inverse LPC and pitch prediction filters and the residual signal is obtained. The closest matching codebook vector for the residual vector is searched for in the codebook and the corresponding index of the codebook vector is obtained. The analysis stage of neural CELP is shown in Figure 4.3(a). The codeword index along with the LPC and pitch parameters are quantized and transmitted to the receiver. The codebook vector corresponding to the codeword index is used at the receiver in the reconstruction of the speech signal. The analysis and synthesis stages of the neural CELP coder are shown in Figure 4.3(b).

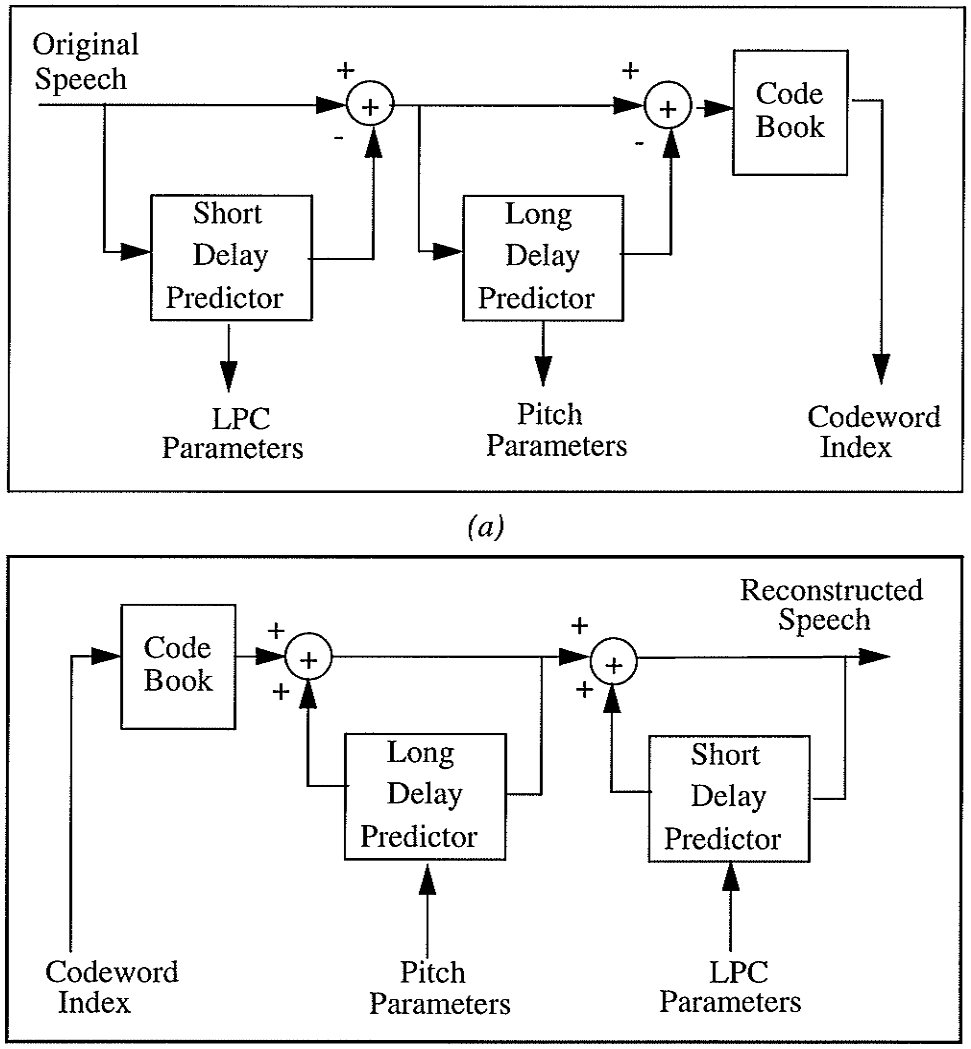

*Figure 4.3. (a) Speech Analysis Model for Neural CELP. (b) Speech Synthesis Model for Neural CELP.*

### 4.2.1. Speaker-Dependent Case (40 ms LPC Frames)

For speaker-dependent speech compression, the speech data was obtained from 10 sentences in the TIMIT database spoken by a single male speaker. The speech signal is preprocessed to obtain ten LPC parameters and the prediction residual is used to estimate the pitch period and three pitch prediction coefficients. A frame size of 40 ms is used for LPC analysis and 20 ms for pitch prediction. The final residual i.e. the excitation signal obtained after LPC and pitch prediction, is vector quantized using the FSCL and KSFM algorithms. The training data set consisted of about 7000 vectors with each vector being a 5 ms (40 dimensional vector) excitation signal. The FSCL and KSFM learning algorithms are employed for training the codebook and the codebook entropy is calculated for different codebook sizes. Figure 4.4 shows the codeword utilization for a codebook size of 512 when the residual excitation is vector quantized using FSCL. Figure 4.5 shows a similar plot when KSFM is employed for VQ. The graphs show that the FSCL gives a larger value of entropy than the KSFM. The distortion is calculated as the average sum of the squares of the error between the training set and the codebook. The graphs show that the FSCL results in a lower distortion than KSFM.

The error during training is shown in Figure 4.6. It can be seen that the average error in FSCL is less than that of KSFM. The FSCL case also reaches a minimum much faster than the KSFM case. The learning in KSFM is performed in two phases as suggested in the software documentation. In the first phase, a large neighborhood and learning rate are used. The codebook in the initial phase is trained for 50000 training iterations. In the second phase, both the neighborhood and the learning rate are reduced and the training is performed for another 150000 training iterations. From Figure 4.6, the error for KSFM decreases when the learning rate is reduced in the second phase of learning. The optimal learning rate was obtained by experimentation. For the FSCL a learning rate of 0.05 was

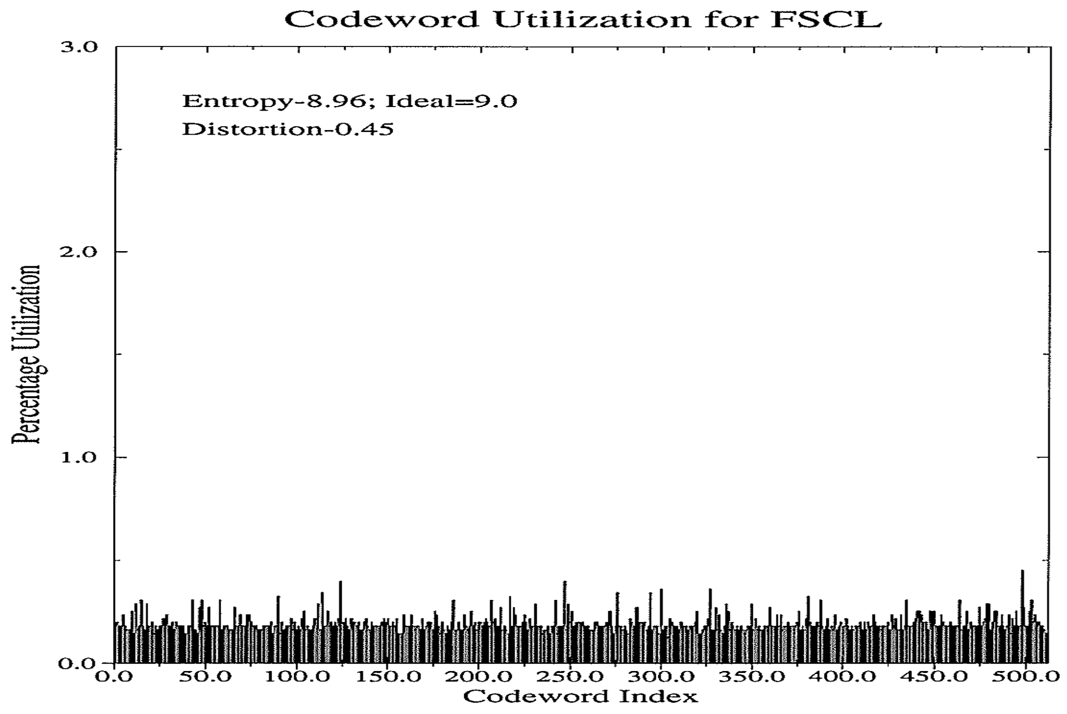

*Figure 4.4. Codeword Utilization for VQ of Excitation Signal using FSCL*

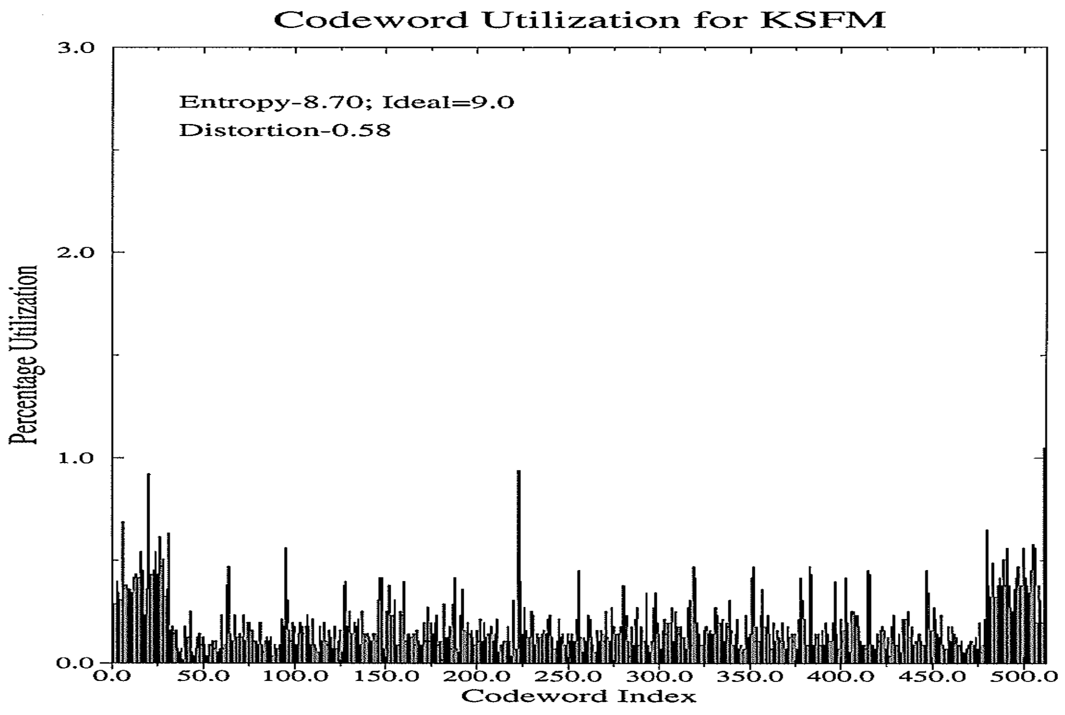

*Figure 4.5. Codeword Utilization for VQ of Excitation Signal using KSFM*

used. For the KSFM, the learning rate in the first phase was 0.05 and in the second phase the learning rate was reduced to 0.02.

Table 4.1. shows the relative codebook entropies for different codebook sizes obtained using the FSCL and KSFM methods. It can be seen from the table that the relative entropy as a percentage of the ideal value reduces slightly as the codebook size increases. FSCL results in a higher entropy than KSFM. This results from the fairness function which encourages uniform utilization of the codewords. The entropy values cannot be compared directly with codebook entropy for the CELP method since the excitation codebook in CELP is generated stochastically.

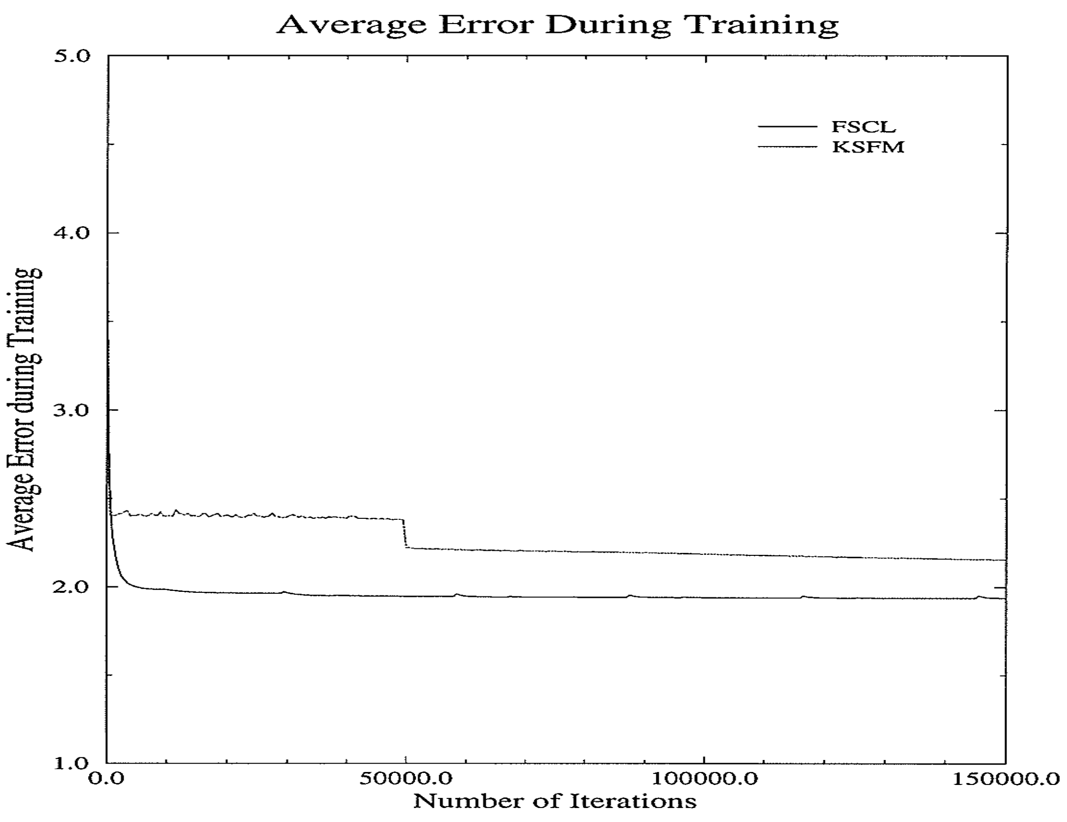

*Figure 4.6. Average Error during Training for FSCL and KSFM Learning*

| Codebook Size | Entropy as% of Ideal (FSCL) | Entropy as% of Ideal (KSFM) |
|---|---|---|
| 512 | 99.5 | 96.6 |
| 1024 | 99.4 | 96.1 |
| 2048 | 98.5 | 95.4 |

*Table 4.1. Entropy for Speaker-Dependent Data.*

Another relevant measure for comparison of the reconstructed speech is the segmental signal-to-noise (SNR). The SNR is calculated as

$$SNR = 10log_{10}\left(\frac{\sum_{n} s^{2}(n)}{\sum_{n} (s(n) - \hat{s}(n))^{2}}\right) \tag{4.1}$$

where *s(n)* is the original speech signal and $\hat{s}(n)$ is the reconstructed speech signal. The SNR for each frame is calculated using (4.1) and the segmental SNR of the entire speech segment is obtained as an average of all the frame SNR values whose value is between -10 dB and 64 dB [26]. The segmental signal-to-noise ratio (SNR) obtained for a speech segment by regular CELP coding is 4.08 dB for an adaptive codebook of size 256 and a stochastic codebook of size 512. The LPC analysis frame size was 40 ms, the adaptive codebook was searched every 20 ms and the stochastic codebook was searched every 5 ms for the regular CELP method. Table 4.2 shows a comparison of the segmental SNR obtained for the reconstructed speech segment using the excitation codebooks learned via FSCL and KSFM learning methods for encoding and decoding the excitation signal. For the preprocessing phase, the LPC analysis frame size was 40 ms, the pitch prediction analysis frame size was 20 ms and excitation codebook was searched every 5 ms. The results show that the present VQ method introduces some distortion in the reconstructed speech for codebooks of size lower than 2048 codewords. The distortion introduced by the codebook results in the distortion in the inverse pitch predictor, thereby reducing the speech quality of the reconstructed speech signal.

| Codebook Size | FSCL SNR dB | KSFM SNR dB |
|---|---|---|
| 512 | 3.10 | 2.33 |
| 1024 | 3.39 | 2.99 |
| 2048 | 4.17 | 3.12 |

*Table 4.2. Comparison of the Segmental SNR for Speaker-Dependent VQ*

The theoretical computation rate required by CELP is 12.6 MIPS [18]. CELP's computations, excluding the codebook searches and including the receiver require approximately 2 MIPS. The adaptive codebook of size 256 requires about 2.3 MIPS and the stochastic codebook of 512 codewords requires 8.3 MIPS which gives the total computation requirement of 12.6 MIPS. The neural CELP method however requires less computational power compared to CELP. The LPC and pitch prediction requires about 1.5 MIPS and the codebook search for a size of 2048 codewords is about 4.7 MIPS giving a total computation rate of 6.2 MIPS. The computational requirements of the neural CELP method may be reduced still further by restricting precision, as discussed in (Section 4.4).

The bit-rate required by CELP is about 4300 bps with a 40 ms LPC frame size, 20 ms frame size for an adaptive codebook of 256 codewords and 5 ms frame size for the stochastic codebook with 512 codewords. The bit-rate obtained by neural CELP with 40 ms LPC frames, 20 ms pitch prediction frames, and 5 ms excitation frame size is about 4800 bps for an excitation codebook of 2048 codewords. The bit-rate for neural CELP is therefore marginally higher than CELP.

For speaker-dependent speech coding, the speech quality using neural CELP is lower than that of CELP for a codebook size below 2048. However, informal subjective tests indicated that the perceived speech quality is very intelligible. The bit-rate using neural CELP is also slightly larger than CELP. The most important advantages of neural CELP coding are the reduced computational requirements and the adaptability to nonstationary environments.

### 4.2.2. Speaker-Independent Speech Compression (40 ms) LPC Frames

For speaker-independent speech compression, the speech data was obtained from 32 speakers from the TIMIT database (26 male, 6 female) having different dialects each speaking one sentence. The training set consisted of about 20000 samples of the residual signal, initially of dimension 40 samples of the excitation signal (5 ms). The FSCL and KSFM learning algorithms were applied to learn the codebooks of different sizes and the entropy was calculated for each codebook. As in the speaker-dependent case, Table 4.5. shows the comparison of relative entropies for the KSFM and FSCL algorithms. The results indicate that the FSCL gives a higher value for entropy than the KSFM method and encourages uniform codeword utilization.

Table 4.5 shows the segmental SNR of the reconstructed signal obtained by using the FSCL and KSFM learning methods for the codebook. The SNR for the CELP method for the same speech sample was 5.6 dB indicating that the CELP method gives better performance than the neural methods in terms of speech quality. The codebook size for the speaker-independent case is larger than in the speaker-dependent case. Once again, the FSCL gives a higher SNR than the KSFM algorithm. In order to obtain a SNR equivalent to that of CELP, the neural method requires a very large codebook and a large number of training samples. Though the SNR is otherwise lower than that of CELP, the perceived speech quality remained very intelligible, and faithful to the speaker, when informal subjective tests were conducted.

| Codebook Size | Entropy as% of Ideal (FSCL) | Entropy as% of Ideal (KSFM) |
|---|---|---|
| 8192 | 98.8 | 95.7 |
| 4096 | 99.0 | 96.0 |
| 2048 | 99.3 | 96.5 |
| 1024 | 99.4 | 96.6 |

*Table 4.3. Entropy for Speaker-Independent Data (40 ms Frames)*

| Codebook Size | FSCL SNR dB | KSFM SNR dB |
|---|---|---|
| 8192 | 5.51 | 4.66 |
| 4096 | 5.03 | 4.19 |
| 2048 | 4.38 | 3.86 |
| 1024 | 4.18 | 3.65 |

*Table 4.4. Segmental SNR for Speaker-Independent Data (40 ms Frames)*

The bit rate obtained by using the neural VQ with a codebook size of 2048 is about 4800 bps. This bit rate can be reduced if we use a VQ for the LPC parameters in addition to the residual VQ. An LPC codebook of size 1024 and a residual codebook of size 2048 will give a bit rate of 4275 bps and this can be reduced further by using a smaller codebook for the LPC parameters. The LPC codebook further degrades the SNR but preserves the perceived speech quality.

The computation rate required by neural CELP when both LPC and residual codebooks of size 1024 are used increases from 6.2 MIPS to 6.35 MIPS because of the additional codebook search involved. This value is still less than the computation rate required by CELP which is 12.6 MIPS for a comparable codebook size.

### 4.2.3. Speaker-Independent Speech Compression (30 ms) LPC Frames

In order to improve the pitch predictor in the neural CELP method, the LPC analysis frame size was reduced from 40 ms to 30 ms (240 samples at 8 kHz sampling rate). The pitch prediction coefficients were extracted using the LPC residual signal on a frame size of 7.5 ms. The excitation obtained after the LPC and pitch prediction was then employed in VQ. Each training vector consisted of 7.5 ms (60 samples) of the excitation signal. The vector quantizer was trained using the FSCL and KSFM algorithms. Table 4.5 shows the comparison of the new SNR values for the FSCL and KSFM algorithms. The segmental SNR obtained using the CELP method was 7.55 dB using the same analysis frame sizes.

The results in Table 4.5 show that the SNR obtained by FSCL for a codebook size of 8192 is higher than the SNR obtained by CELP. Informal subjective tests indicated that the perceived speech quality is similar to that of CELP even with a codebook size of less than 8192 codewords. The bit-rate obtained by using the neural VQ with a codebook size of 2048 and CELP are both about 4800 bps. The computation rates are further reduced in the neural CELP method to 5.4 MIPS for a codebook of size 2048 codewords. This reduction in the computation rates is due to the reduction in the LPC and pitch analysis frame size and to the increase in the codeword dimension.

| Codebook Size | FSCL SNR dB | KSFM SNR dB |
|---|---|---|
| 8192 | 8.62 | 5.96 |
| 4096 | 6.04 | 4.96 |
| 2048 | 5.39 | 4.46 |
| 1024 | 4.96 | 4.10 |

*Table 4.5. Segmental SNR for Speaker-Independent Data (30 ms LPC Frames)*

## 4.3. Subjective Speech Quality Tests

In order to test the speech quality obtained by using neural CELP, subjective speech quality tests were conducted. A new data set which consisted of a vowel database [28] was used in the subjective tests. The data set consisted of isolated words and pseudowords (had, hid, head, hod, hood, hud) spoken by 7 female and 13 male speakers, each saying all 6 of the words. The codebooks were trained using the FSCL learning algorithm with a codebook of size 4096 codewords. The words were then reconstructed using the codebook and tested for their subjective quality. The SNR for the test samples using neural CELP was less when compared to the SNR obtained by regular CELP. However, the reconstructed speech samples were tested for intelligibility. The original speech signal, the reconstructed signal using CELP, and the reconstructed signal using neural CELP were played to a successive set of listeners who were asked if the words could be recognized. The audience was also asked to rate the quality of the speech samples on a scale of 1 to 5 with 5 being excellent and 1 being poor. The results of the word recognition tests are shown in Table 4.6. The results show that all the vowels could be distinguished in 85% of the trials. The vowels that were wrongly identified using the neural CELP method couldn't be recognized using either the original speech or the CELP coder. In the word recognition tests, most of the cases that resulted in misclassification of the vowels was caused due to an apparent "t" sound in the recordings at the end of the words for certain speakers, rather than the appropriate "d" sound. In the second test, the quality of the speech was tested. The results of the mean-opinion-score (MOS) tests are shown in Table 4.7. The results show the speech quality the CELP output is very close to that of the original speech. The listeners indicated that the words were very intelligible and the quality of neural CELP was close to that of the original except for a slight raspy or hoarse sound in the output. This was responsible for the lower MOS for the neural CELP method.

| Compression Method | Accuracy |
|---|---|
| Original | 85% |
| CELP | 85% |
| Neural CELP | 85% |

*Table 4.6. Word Recognition Tests*

| Compression Method | MOS |
|---|---|
| Original | 4.5 |
| CELP | 4.4 |
| Neural CELP | 4.0 |

*Table 4.7. Speech Quality Tests (Mean-Opinion Score)*

## 4.4. Computing with Reduced Power Requirements

Most neural network simulations on conventional computers have been performed using a 32-bit floating-point representation for the network's weights, distortion measures and weight-update calculations. Using lower precision numbers, it is possible to place more weights and processing elements in the same chip area, with a corresponding increase in the performance and storage capacity to hardware cost. If the computations are performed in lower precision arithmetic the computational requirements and with them the power consumption in the device can be reduced. By reducing the power consumption in the device, the system can be used for portable communications and computing devices.

Experiments have been conducted to examine if the artificial neural learning algorithms can tolerate reduced precision during encoding and, and more importantly, during learning. During encoding, the 7.5 ms excitation signal is used as the input to the codebook and the corresponding closet vector in the codebook is identified. The output is the index of the winning vector i.e. the vector whose distortion is the lowest. Experiments were conducted when the precision was reduced during distortion calculations. The test speech signal was encoded using reduced precision distortion calculation, and then decoded using the method of Figure 4.3(b). The segmental SNR of the reconstructed signal was calculated and is shown in Table 4.8. The SNR was only reduced to 4.9 from 4.96, for example, for a 1024 word codebook size when 12 bit precision was used.

Limited precision arithmetic was also applied during learning. The precision was reduced during both distortion and weight update calculations during learning. The weights are also stored using reduced precision. Table 4.8 shows the results obtained when limited precision arithmetic was performed during learning.

| Precision in bits | FSCL SNR dB |
|---|---|
| 16 | 4.95 |
| 12 | 4.9 |
| 8 | 4.88 |
| 6 | 3.47 |

*Table 4.8. Segmental SNR for Limited Precision Classification*

The codebook size was 1024 and the FSCL method was used for learning. From Table 4.5, the SNR using 32 bit precision for a codebook size of 1024 is 4.96 dB. These results indicate that the ANN algorithms are tolerant of low precision arithmetic which can result in low power consumption for portable computing with little degradation of speech quality. However, the precision cannot be reduced below approximately 8 bits without a significant degradation in the speech quality. Subjective tests have shown that the perceived speech quality remains intelligible for precision greater than 8 bits. This system can therefore be used in real-time applications using low power DSP chips or custom VLSI hardware.

| Precision in bits | FSCL SNR dB |
|---|---|
| 16 | 4.91 |
| 12 | 4.88 |
| 8 | 4.37 |
| 6 | 3.17 |

*Table 4.9. Segmental SNR for Limited Precision Learning*

## 4.5. Summary

The work presented in this chapter examined the neural CELP algorithm and its performance for speech coding. In the neural CELP algorithm, the excitation signal is vector quantized using FSCL and KSFM algorithms and the codebook is generated using training samples. The LPC and pitch prediction coefficients of the input speech signal are calculated and the excitation signal is identified from the codebook for the closet matching codebook vector. The index of the closet matching codebook vector along with the pitch and LPC coefficients are quantized and transmitted to the receiver. At the receiver, the codebook vector corresponding to the codeword index is used in the reproduction of the speech. Speaker-dependent and speaker-independent speech compression experiments were conducted. For both the experiments, the results showed that the codebooks generated by FSCL provide improved codeword utilization of the than those using KSFM. The distortion obtain by FSCL is also lower than that of KSFM. The comparison of the results obtained by the neural CELP method with the regular CELP method showed that neural CELP is capable of producing comparable bit-rates and lower computation rates than CELP, as well as being suitable for a nonstationary environment. CELP has a slightly higher SNR compared to that of neural CELP for small excitation codebook sizes. Subjective speech quality tests showed that the speech output produced by neural CELP is very intelligible except for a slight raspy or hoarse sound.

# Chapter 5

# Conclusions and Future Work

## 5.1. Summary and Conclusions

This thesis began with a brief overview of conventional speech coding techniques including linear predictive coding, code-excited linear prediction and vector quantization. The potential role of artificial neural networks for vector quantization (VQ) of speech was then described. Several artificial neural learning methods for adaptive VQ of speech were discussed. Artificial neural learning methods are adaptive and hence allow for the possibility of learning the codebooks on-line in a non-stationary environment such as the corner automated teller machine or in a portable computer. Three unsupervised learning methods: competitive learning, frequency-sensitive competitive learning and Kohonen self-organizing feature maps were discussed, for learning an adaptive codebook using VQ. These learning methods were adopted in a new speech compression technique known as neural CELP. In our neural CELP method, the input speech was preprocessed to obtain 10 LPC coefficients. The output of the inverse LPC filter was used to extract the pitch period and 3 pitch prediction coefficients. The final residual after LPC and pitch prediction was used to generate the codebook. Experiments were conducted for speaker-dependent and speaker-independent speech compression. The codebook was generated using 5 ms and 7.5 ms signals of the excitation signal. The codebook entropies, segmental SNR, bit-rates, and computational rates were calculated for codebooks obtained using FSCL and KSFM, and these were compared to the results of CELP coding. Experiments were also conducted by employing low precision arithmetic for classification and learning.

The results of this thesis showed that artificial neural learning methods are of potential importance for speech coding. The techniques used in this study differ from earlier neural VQ studies and conventional CELP coding methods. The excitation signal was vector quantized using the neural learning methods rather than quantizing the LPC parameters as in the earlier studies. There was only one codebook for the excitation which was generated by training on speech samples as opposed to the stochastic codebook in the standard CELP approach. The results obtained demonstrate that the neural CELP approach is capable of producing high quality speech, low computation rates and low bit-rates. The subjective speech quality tests indicated that the quality of speech with smaller codebook sizes using the neural CELP approach was below that of CELP but remained very intelligible. The listeners indicated a slight raspy or hoarse sound for a few words in the test set using neural CELP. The artificial neural learning methods are also tolerant of low precision arithmetic. The results of low precision computing indicated that the ANN learning algorithms perform well even if the precision is reduced from 32 bits to 8 bits with only a small reduction in the SNR ratio. However, if the precision is reduced below 8 bits, there is significant degradation of speech quality. These results indicate that the system can be implemented in real-time by using a DSP chip for preprocessing of speech and by replacing the neurally-learned vector quantizer by a lookup table. However, ultimate real-time performance at low power in non-stationary environments will be best obtained using integer arithmetic of restricted precision in dedicated ANN hardware.

## 5.2. Future Work

In this work, only software simulations of the experiments were performed. In order to make the system more practical to use, it should be implemented as custom hardware or as low-power DSP chips. More research should be performed regarding speech quality of the neural CELP method which may include obtaining better pitch detection algorithms, or alternatively, post-processing of the reconstructed speech. The performance of the algorithms for speech signals for speakers with high pitch (e.g. female speakers) was lowest. An adaptive pitch prediction algorithm may have to be used for such speakers. An artificial neural network can be used for pitch prediction which may result in an overall improvement of the speech quality. The learning time of the algorithms for a large codebook size is significant even on a high performance computer. More effort is warranted in developing fast learning algorithms so that codebooks can be generated on-line in real-time applications. These faster learning algorithms can be employed for fine-tuning the codebooks on-line in non-stationary environments.

# References

[1] L. R. Rabiner, “Applications of Voice Processing to Telecommunications”, *Proceedings of the IEEE,* Vol. 82, No. 2, February 1994, pp. 197-228.

[2] P. E. Papamichalis, “Practical Approaches to Speech Coding”, *Prentice-Hall Inc.,* 1987.

[3] S. Singhal, D. L. Gall and C. Chen, “Source Coding of Speech and Video Signals”, *Proceedings of the IEEE*, Vol. 78, No. 7, July 1990, pp. 1233-1249.

[4] J. Hertz, A. Krough and R. G. Palmer, “Introduction to the Theory of Neural Computation, *Addison Wesley Publishing Company,* 1991.

[5] S. Ahalt, A. K. Krishnamurthy, P. Chen and D. E. Melton, “Competitive Learning Algorithms for Vector Quantization”, *Neural Networks*, Vol. 3. No. 3, 1990, pp. 277-290.

[6] S. Kamarsu and H. C. Card, “Artificial Neural Learning for Speech Compression”, *17th Biennial Symposium on Communications,* Kingston, May 1994, pp. 111-114.

[7] S. Kamarsu and H. C. Card, “Vector Quantization of Speech using Artificial Neural Learning”, to appear in the *IEEE Pacific Rim Conference*, Victoria, May 1995.

[8] ,S. Kamarsu and H. C. Card, “Neural Code-Excited Linear Prediction for Low Power Speech Compression”, to appear in the *IEEE WESCANEX 95*, Winnipeg, May 1995.

[9] A. S. Spanias, “Speech Coding: A Tutorial Review”, *Proceedings of the IEEE,* Vol. 82, No. 10, October 1994, pp. 1539-1582.

[10] L. R. Rabiner and R. W. Shafer, “Digital Processing of Speech Signals”, *Prentice-Hall Inc.,* 1978.

[11] R. P. Ramachandran and P. Kabal, “Pitch Prediction Filters in Speech Coding”, *IEEE Transactions on ASSP,* Vol. 37, No. 4, April 1989, pp. 467-478.

[12] Y. Linde, A. Buzo, R. M. Gray, “An Algorithm for Vector Quantizer Design”, *IEEE Transactions on Communications,* Vol. Com-28, No. 1 January 1980, pp. 84-95.

[13] R. M. Gray, “Vector Quantization”, *IEEE ASSP Magazine,* April 1984, pp. 4-28.

[14] J. Makhoul, S. Roucos and H. Gish, “Vector Quantization in Speech Coding”, *Proceedings of the IEEE,* Vol, 73, No. 11, 1985, pp. 1551-1588.

[15] Y. Cheng and D O’Shaughnessy, “On 450-600 b/s Natural Sounding Speech Coding”, *IEEE Transactions on Speech and Audio Processing*, Vol. 1, No. 2, April 1993, pp. 207-220.

[16] B. S. Atal, “Predictive coding of Speech at Low Bit Rates”, *IEEE Transactions on Communications,* Vol. Com-30, No. 4, April 1982, pp. 600-614.

[17] M. F. Schroeder, B. S. Atal, “Code-Excited Linear Prediction (CELP): High Quality Speech at Very Low Bit Rates”, *Proceedings of ICASSP,* April 1985, pp. 937-940.

[18] J. P. Campbell, Jr., T. E. Tremain, V. C. Welch, “The DOD 4.8 KBPS Standard (Proposed Federal Standard 1016)”, *Advances in Speech Coding,* Kluwer Academic Publishers, 1990, pp. 121-133.

[19] A. K. Krishnamurthy, S. C. Ahalt, D. E. Melton and P. Chen, “Neural Networks for Vector Quantization of Speech and Images”, *IEEE Journal of Selected Areas in Communication*, Vol. 8. No. 8, October 1990, pp. 1449-1457.

[20] T. Lee and A. M. Peterson, "Adaptive Vector Quantization using a Self-Development Neural Network", *IEEE Journal of Selected Areas in Communication,* Vol. 8. No. 8, October 1990, pp. 1458-1471.

[21] L. V. Veleva and R. K. Kunchev, "Adaptive Speech Coding with DCT and Neural Net Vector Quantization", *Electronics Letters,* Vol. 29, No. 8, 15th April 1993, pp. 704-705.

[22] G. E. Hinton and S. J. Nowlan, "The Bootstrap Widrow-Hoff Rule as a Cluster-Formation Algorithm", *Neural Computation,* Vol. 2, 1990, pp. 355-362.

[23] S. Grossberg, "Adaptive Pattern Classification and Universal Recording: II. Feedback, Expectation. Olfaction, Illusions", *Biological Cybernetics*, 23, 1976, pp. 187-202.

[24] T. Kohonen, "The Self-Organizing Map", *Proceedings of the IEEE*, Vol. 78, No. 9, 1990, pp. 1464-1480.

[25] T. Kohonen, J. Kangas, J. Laaksonen, "SOM_PAK, The Self-Organizing Map Program Package", *Helsinki University of Technology*, Laboratory of Computer and Information Science, 1992. ftp: ftp.cochlea.hut.fi;/pub/som_pak.

[26] J. Campbell, "FS 1016 CELP Code, Version 3.2", *Department of Defense,* August 1993. ftp: super.org/pub/celp_3.2a.tar.Z.

[27] "TIMIT Acoustic-Phonetic Continuous Speech Corpus", *DARPA*, NIST Speech Disc 1-1.1, October 1990.

[28] S. E. Fahlman, "CMU Benchmark Collection for Neural Net Learning Algorithms", *Carnegie Mellon University*, School of Computer Science, 1993.

# Appendix A

# Estimation of LPC and Pitch Parameters

## A.1. LPC Parameter Estimation

In order to obtain the LPC parameters, the vocal tract information and the gain must be determined. The speech signal is windowed using the windowing function defined in (2.4) and the windowed speech signal is given by

$$s_n(m) = s(m+n)\,w(m) \ . \tag{A.1}$$

From (2.2), the estimate $\hat{s}_n(m)$ is given by

$$\hat{s}_n(m) = \sum_{k=1}^{N_f} a_k s_n(m-k) \ . \tag{A.2}$$

In order to obtain $a_k$, we have to minimize the error over all available samples.

$$\sum_m (s_n(m) - \hat{s}_n(m))^2 \tag{A.3}$$

Minimization leads us to the following set of linear equations which can be expressed in a matrix form as

$$\begin{bmatrix} R_n(0) & R_n(1) & R_n(2) & \dots & R_n(N_f-1) \\ R_n(1) & R_n(0) & R_n(1) & \dots & R_n(N_f-2) \\ R_n(2) & R_n(1) & R_n(0) & \dots & R_n(N_f-3) \\ \dots & \dots & \dots & \dots & \dots \\ R_n(N_f-1) & R_n(N_f-2) & R_n(N_f-3) & \dots & R_n(0) \end{bmatrix} \begin{bmatrix} a_1 \\ a_2 \\ a_3 \\ \dots \\ a_{N_f} \end{bmatrix} = \begin{bmatrix} R_n(1) \\ R_n(2) \\ R_n(3) \\ \dots \\ R_n(N_f) \end{bmatrix} . \tag{A.4}$$

Here, $R_n(k)$ is known as the autocorrelation function defined as

$$R_n(k) = \sum_{m=0}^{N-1-k} s_n(m)\, s_n(m+k) . \tag{A.5}$$

This $N_f \times N_f$ matrix of autocorrelation values is a Toeplitz matrix i.e., symmetric and all elements along a given diagonal are equal. Such a matrix is nonsingular and it can always be inverted. Hence, a solution can always be found, and the stability of the resulting solution can be guaranteed. This property can be exploited to obtain an efficient algorithm for the solution. The Durbin's recursive algorithm is used to solve the matrix equation above for the LPC coefficients $a_k$.

## A.2. Estimation of Pitch Parameters

The pitch predictor removes the distant sample correlations. The value of the pitch period is chosen so as to maximize the correlation of the residual after the formant prediction. A normalized correlation array given by

$$\sum_{m=M}^{M-1+N_p} \frac{\phi^2(0,m)}{\phi(m,m)} \tag{A.6}$$

where

$$\phi(i,j) = \sum_{n=0}^{N-1} d(n-i)\,d(n-j) \tag{A.7}$$

and *d(n)* is the residual signal after formant prediction which acts as the input to the pitch predictor. The value of $M$ that maximizes the quantity in (A.6) is chosen as the pitch period. The pitch filter is given by

$$P(z) = \beta_1 z^{-M} + \beta_2 z^{-(M+1)} + \beta_3 z^{-(M+2)} \tag{A.8}$$

The parameters $\beta_i$ are estimated by solving the following matrix equation:

$$\begin{bmatrix} \phi(M,M) & \phi(M,M+1) & \phi(M,M+2) \\ \phi(M+1,M) & \phi(M+1,M+1) & \phi(M+1,M+2) \\ \phi(M+2,M) & \phi(M+2,M+1) & \phi(M+2,M+2) \end{bmatrix} \begin{bmatrix} \beta_1 \\ \beta_2 \\ \beta_3 \end{bmatrix} = \begin{bmatrix} \phi(0,M) \\ \phi(0,M+1) \\ \phi(0,M+2) \end{bmatrix}. \tag{A.9}$$

In many cases, the off-diagonal elements in the covariance matrix (A.9) are negligible. These terms are correlation terms with small lags. This approximation is justified by the observation that when LPC analysis is done before the pitch prediction, the LPC filter removes the short-term correlations. So, (A.9) can be simplified as

$$\begin{bmatrix} \beta_1 \\ \beta_2 \\ \beta_3 \end{bmatrix} = \begin{bmatrix} \phi(0,M)/\phi(M,M) \\ \phi(0,M+1)/\phi(M+1,M+1) \\ \phi(0,M+2)/\phi(M+2,M+2) \end{bmatrix}. \tag{A.10}$$

Using (A.10), the pitch prediction coefficients $\beta_i$ can be estimated.