Robust Spectral Parameter Coding in Speech Processing

Nadim Batri
B. Eng.

Department of Electrical Engineering
McGill University
Montreal, Canada

May 1998

A thesis submitted to the Faculty of Graduate Studies and Research in partial fulfillment of the requirements for the degree of Master of Engineering.

Abstract

Linear predictive coding (LPC) is employed in many low bit rate speech coders. LPC models the short-term spectral information for a block of speech using an all-pole response. Line spectral frequencies (LSF) have been found to be an effective parametric representation for the all-pole response.

Vector quantization (VQ) is often used to code the coefficients of the response. VQ performs poorly whenever it is coding coefficient vectors which are not well matched to the distribution of its codebooks. A shift in the distribution can be caused by filtering (microphones, filters in communication equipment, etc.), speaker or environmental variability (male, female, background noise, etc.). In this thesis, we explore a method for matching the distribution of the vectors representing the incoming speech signal to the distribution of the codebooks. A novel mapping model based on the transformation of codebooks using the mean and the standard deviation of the distributions is used to increase the robustness of vector quantization. The mapping model is optimized in two ways — choosing the most suitable spectral parameter representation and seeking the best way to obtain the form of the mapping model. The effectiveness and limitations of this method are investigated through simulation of a split vector quantizer (SVQ) of the LPC coefficients.

Sommaire

Le codage à prédiction linéaire (LPC) est employé dans plusieurs algorithmes de compression de paroles au très bas débit binaire. LPC représente le spectre court-terme pour un block de parole avec une réponse tous pôles. Les fréquences de raies spectrales (LSF) ont été jugées une représentation paramétrique efficace pour la réponse tous pôles.

La quantification vectorielle (VQ) est utilisée fréquement pour coder les coefficients de la réponse. La performance du VQ est pauvre quand des vecteurs qui ne sont pas representés dans ses livres de code sont codés. Un déplacement dans la distribution peut être causê par un filtrage (microphones, filtres dans des équipements de communication, etc.), par une variabilité d'orateur ou d'environnement (homme, femme, bruit embiant, etc.). Dans cette thèse, on explore une méthode pour faire correspondre la distribution des vecteurs représentant le signal de parole reçue à la distribution des livres de code. Un nouveau modèle de correspondance, basé sur la transformation des livres de code en utilisant la moyenne et l'écart-type des distributions, est utilisé pour améliorer la robustesse de la quantification vectorielle. Le modèle de correspondance est optimisé en deux façons — choisir la meilleur représentation paramétrique spectrale et trouver la bonne façon pour obtenir la forme du model de correspondance. L'efficacité et les limitations de cette méthode sont étudiées par le biais de simulation d'une quantification vectorielle divisée (SVQ) des coefficients LPC.

Acknowledgments

I would like to express my deepest gratitude to my supervisor Prof. Peter Kabal for his knowledge and guidance throughout my graduate studies at McGill. His patience and his good humor had made my stay very much enjoyable.

I would like to thank the Natural Sciences and Engineering Research Council of Canada (NSERC) who financially supported the project.

I would like to thank all my friends in the Telecommunication and Signal Processing Laboratory, past and present, for their fruitful ideas and interesting discussions. Special thanks to Jacek Stachurski, not only for his very helpful technical advice but for his friendship as well. I am particularly thankful for Eddie Choy for his support and help throughout my two years in masters.

Finally I would like to thank my parents for their support and encouragement during my whole life.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Wireless communication has become an important technology in the modern era. The transmission of voice, audio, video and data must be efficient in use of bandwidth and reliable in the face of varying channel conditions. Speech processing plays a key role in enabling wireless transmission of voice and audio. In particular, digital speech coding is important to increase system capacity while maintaining high voice quality. Specifically, speech coding is the process of representing a speech signal digitally. Such a digital signal can then be transmitted or stored efficiently. Once digitized, the digital representation of a speech signal incurs little or no further distortion due to transmission and storage. This is in contrast to analog representations which suffer from distortion due to transmission (channel noise) or storage (tape hiss).

The primary goal of a speech coding system is to achieve acceptable reconstruction of the speech signal with certain constraints, such as bit rate, complexity and especially robustness. More recently, different kinds of robustness have been studied: robustness to channel noise [1], robustness to speaker variability [2], and robustness to filtering conditions [3]. The latter is the subject of this thesis.

The speech is often coded in the form of parameters that represent the signal economically while still allowing speech reconstruction with minimal quality loss. A source filter model that is based on the physiology of human speech production is often used to parameterize certain features of the speech frequency magnitude spectrum associated with each frame of speech signal. The model is based on the output speech being a linear combination of past speech in addition to an input excitation.

This all-pole filter models the resonant frequencies, or formants, of the speech spectrum. The filter coefficients (called LP for linear predictive) can be converted through mathematical transformations into other parametric representations such as reflection coefficients or line spectral frequencies. These spectral parameters need to be quantized with sufficient accuracy to maintain speech intelligibility and quality. Scalar quantization, vector quantization and matrix quantization are some of the numerous quantization methods.

1.1 Speech Properties

Before elaborating on speech coding, some speech properties have to be discussed. Speech is highly redundant. For example, multiple peak clipping of the speech signal (i.e. reducing it to binary waveform) eliminates virtually all amplitude information, yet listeners easily understand speech distorted in such a way. Speech signals are non-stationary and at best can be considered as quasi-stationary over short segments, typically 5–20 ms. The statistical and spectral properties of speech are thus defined over short segments. Speech can generally be classified as *voiced* (e.g., /a/, /i/, etc), *unvoiced* (e.g., /sh/), or mixed. Time- and frequency-domain plots for voiced and unvoiced segments are shown in Fig. 1.1. Voiced speech is quasi-periodic in the time domain and harmonically structured in the frequency domain while unvoiced speech is noise-like and broadband. In addition, the energy of voiced segments is generally higher than the energy of unvoiced segments. An important feature in the spectrum are the resonances (peaks) known as formants (see the spectrum of the voiced speech in Fig. 1.1). These formants are known to be perceptually important in the recognition of speech sounds, particularly for vowels. In this thesis, we will examine coding schemes which attempt to reproduce the spectral features accurately.

Two major types of correlations are present in a speech signal. These are known as near-sample redundancies and distant-sample redundancies. Near-sample redundancies are those which are present among speech samples that are close together. Distant-sample redundancies are due to the inherent periodicity of voice speech.

These distinctive properties of speech are the foundation of speech coding or speech compression.

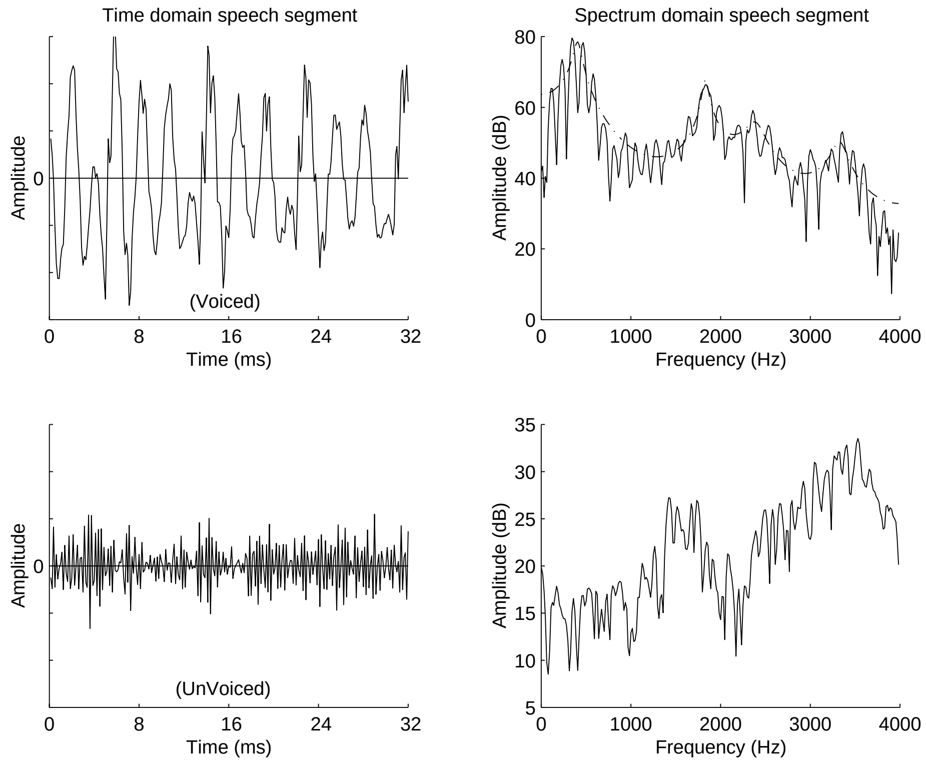


Fig. 1.1 Voiced and unvoiced segments and their short-time spectra. The power spectra were calculated over segments 32 ms long smoothed with a Hamming window. The dotted line represent the spectral envelope or the formant structure for the voiced segment.

1.2 Overview of Speech Coding

Speech coding is the field concerned with obtaining compact digital representations of voice signals for the purpose of efficient transmission or storage. It involves sampling and amplitude quantization. An analog speech waveform $s_a(t)$ is sampled at a rate $f_s \geq 2B$, where B is the frequency bandwidth of $s_a(t)$, yielding the discrete-time speech signal $s[n]$. At this stage, the quantization or binary representation of $s[n]$ can be direct or parametric. Direct quantization implies binary representation of the speech sample themselves while parametric quantization involves binary representation of a speech model and/or spectral parameters.

In the simplest type of coder, *pulse code modulation* (PCM), the amplitude of the speech signal $s[n]$ is quantized to one of 2^R magnitude levels, where R is the number of bits used to encode each sample. Different types of quantization are used to encode the speech signal. We can classify them as uniform, nonuniform and adaptive quantization. This method does not have any mechanism for redundancy removal. Nonuniform quantizers are more useful since they encode the speech signal with fewer bits than that used in linear (or uniform) quantization because human hearing sensitivity tends to be logarithmic, and because speech signals have an amplitude distribution biased toward lower amplitudes [4]. By quantizing a prediction error signal rather than the original signal sample, *differential pulse code modulation* takes advantage of the existing correlation among neighboring samples to reduce the bit rate. This prediction error signal is the difference between the current sample and its predicted value. Similar methods such as *delta modulation* (DM), *adaptive delta modulation* (ADM) and *adaptive differential pulse code modulation* (ADPCM) use the redundancy of speech to reduce the bit rate or alternately improve the quality of speech [5, 6].

Compressing speech to further reduce bit rates involves using an *analysis-synthesis* process. Figure 1.2 shows the basic components of such a system. In the analysis stage, speech is represented by a compact set of parameters which are then encoded efficiently. In the synthesis stage, these parameters are decoded and used in conjunction with a reconstructive mechanism to form speech.

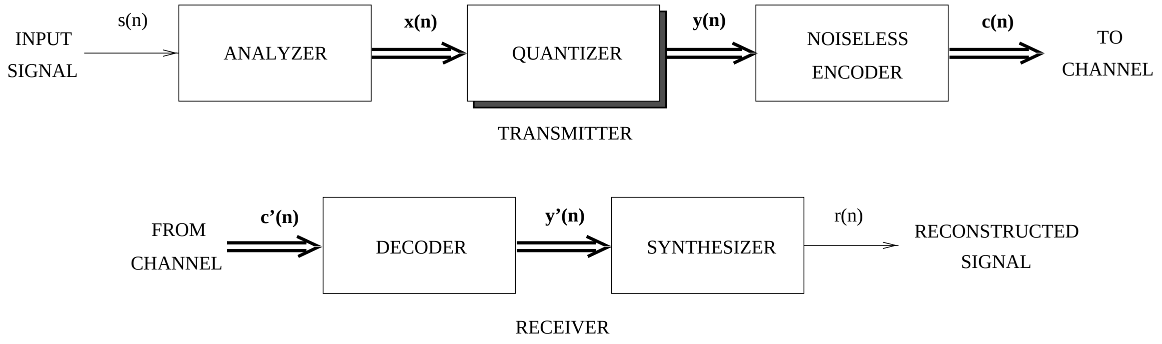


Fig. 1.2 Basic components of an analysis-by-synthesis compression system.

The quantization block is one of the main stages of a speech coder. Vector quantization is the process of jointly quantizing the set of parameters as a single vector to another real-valued, discrete-amplitude vector. While scalar quantization can take advantage of the correlation of consecutive speech samples and their probability density function, vector quantization can take advantage of four interrelated properties of vector parameters: *linear dependency, nonlinear dependency, shape of the probability density function (pdf), and vector dimensionality* [7].

1.3 Motivation for Our Research

A principal use of Vector Quantization[†] in speech coding has been to reduce the transmission rate to 1200-bit/s and below for the spectral parameter while maintaining acceptable speech intelligibility and quality. More recently, VQ has been studied at higher data rates [8, 9]. VQ has been also used regularly and effectively in pattern-recognition type of speech applications, such as in speech and speaker recognition [10].

VQ is a process where a collection of representative vectors is stored in a codebook and then an input vector is matched to the “closest” vector in the codebook. An index will then represent the quantized version of the vector. The drawback of VQ can be its large cost of memory and computational complexity. In general, it increases exponentially with the number of dimensions and the number of bits per dimension.

One measure of the efficiency of VQ is its robustness. Robustness can refer to the ability of a VQ to perform adequately in presence of channel errors. Knagenhjelm [11] introduced the idea of reordering the indexes of the codebook to mitigate the effect of channel errors. Robustness can also refer to the resistance to degraded performance when tested on data whose distribution is different of that of the training data used to implement the codebook for VQ. Due to the diverse distribution that we can get from speech signals, such as speaker variability, different filtering (microphones and filters of communications equipments) or environmental variability (background noise), robustness becomes very difficult to achieve. Building a codebook for each probability density function and alternating between them when appropriate is a possible solution for this problem. It is however tedious and impractical to implement since it demands large storage capabilities. The storage problem worsens if the dimension of the vector or the dimension of the codebook increases.

Mapping one probability density function (pdf) to another is the subject of this thesis. In doing so, we will be using only one set of data and transformations rather

[†]In this thesis, Vector Quantization is abbreviated as VQ.

than storing all possible codebooks. This technique saves a lot of memory in return for a small additional complexity. Our study targets one type of robustness, which involves filtered speech. When filtering speech signals, the pdfs of the respective prediction coefficients vary significantly and in turn degrade the performance of VQ. One solution, proposed by Ramachandran and Sondhi [3], was to switch to a scalar quantizer when the performance of the VQ is poor. In this case, one bit has to be added to the transmission block to determine which quantization method is used. The resulting improvement is modest. In our study, a mapping function is used to convert one codebook to satisfy the pdfs of different types of filtered speech. Additional computational complexity is added to determine the type of filtering done on the speech and to select the proper mapping function. The complexity of the proposed scheme is small since we are applying a simple mapping once per analysis frame to the single vector of LP parameters derived from the speech.

1.4 Organization of Thesis

The intent of this thesis is to provide a method to improve the robustness of VQ for different type of filtered speech signals. Chapter 2 reviews the method of linear predictive coding that is used in most speech coders to model the short-term spectral parameter. We will introduce several alternative parametric representations of LP filter coefficients as well as some objective distortion measures used to evaluate quantizer performance. Chapter 3 provides an overview of VQ and its basic components and then explores different methods that will increase its robustness. In particular, individual mapping of the LP coefficients and difference mapping are explored and theoretical limitation for the stability of the LP filter are presented. Chapter 4 describes the implementation of the mapped VQ and includes simulation results and performance evaluations. Chapter 5 concludes the thesis with a summary of our work and suggestions for future investigation.

Chapter 2

Linear Predictive Speech Coding

The purpose of this chapter is to give an overview of linear predictive speech coding. The quality and reliability of coded speech rely on the accurate reconstruction of the envelope of the short-time power spectrum. Linear predictive coding (LPC) is widely used to encode the spectral envelope because of its relative computational simplicity. This method models the speech signal as a linear combination of past speech samples and an excitation signal source. In speech coding applications, the linear predictor (LP) parameters are extracted frame-by-frame from the speech signal, typically at a rate of 50 frames/sec. For telephone speech sampled at 8 kHz, typically a 10'th order LP analysis is performed. Different representation of the LP parameters and distortion measures are introduced in this chapter. The speech database and filters used in this thesis are described as well.

2.1 LPC Model

The most general predictor form in linear prediction is the *autoregressive moving average* (ARMA) model where a speech sample $s[n]$ is predicted from p past predicted speech samples $s[n-1], \ldots, s[n-p]$ with the addition of an excitation signal $u[n]$ [12] according to:

$$s[n] = \sum_{k=1}^{p} a_k s[n-k] + G \sum_{l=0}^{q} b_l u[n-l], \qquad b_0 = 1, \tag{2.1}$$

where G is a gain factor for the input speech and a_k and b_l are sets of filter coefficients. Equivalently, in the frequency domain, the transfer function of the linear prediction speech model is

$$H(z) = \frac{B(z)}{A(z)} = G \frac{1 + \sum_{l=1}^{q} b_l z^{-l}}{1 - \sum_{k=1}^{p} a_k z^{-k}}. \tag{2.2}$$

$H(z)$ is referred to as a pole-zero model in which the polynomial roots of the denominator and the numerator are, respectively, the poles and zeros of the system. When $a_k = 0$ for $1 \le k \le p$, $H(z)$ becomes an all-zero or *moving average* (MA) model since the output is a weighted average of the q prior inputs. Conversely, when $b_l = 0$ for $1 \le l \le q$, $H(z)$ reduces to an all-pole or *autoregressive* (AR) model in which case the prediction operation is written as:

$$s[n] = \sum_{k=1}^{p} a_k s[n-k], \tag{2.3}$$

and its frequency domain transfer function simply as:

$$H(z) = \frac{1}{1 - \sum_{k=1}^{p} a_k z^{-k}} = \frac{1}{A(z)}. \tag{2.4}$$

The spectral pattern can be well modeled by $1/A(z)$. A shortcoming of this model is the absence of representation of the spectral zeros due to the glottal source and the vocal tract response in the nasal portion. Usually such zeros contribute nothing to the spectral magnitude and add only linear phase, since they result from simple time delay. With a reasonable number of coefficients p, say 10, the spectral match is quite acceptable. Another shortcoming is the poor prediction of unvoiced sounds.

The output $e(n)$ is called *prediction error* or *residual*:

$$e[n] = s[n] - \sum_{k=1}^{p} a_k s[n-k]. \tag{2.5}$$

The LP coefficients can be computed from the input speech signal. In that case these coefficients need to be transmitted to the decoder, and this technique is referred to as *forward adaptive* (see Fig. 2.1a). On the other hand when linear prediction is performed using previously reconstructed speech samples, this procedure is referred to as *backward adaptive* and has the additional advantage that no explicit transmission of the LP coefficients is needed (see Fig. 2.1b). Since we are concerned with the coding of the LP coefficients for transmission, the forward adaptive technique is used.

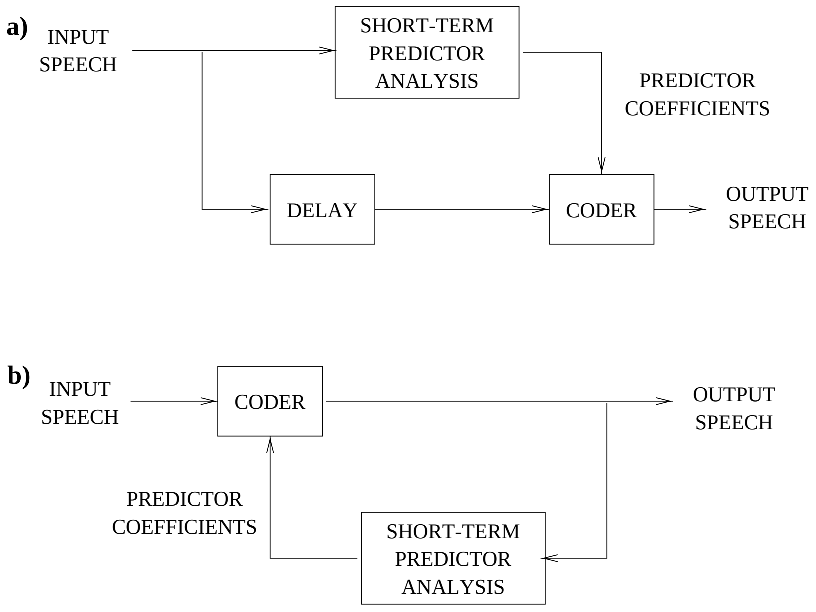


Fig. 2.1 Block diagram of (a) a forward-adaptive coder, and (b) a backward-adaptive coder.

There are two widely used approaches for the estimation of the LP coefficients which are presented in the next section.

2.2 Estimation of the Linear Prediction Coefficients

A speech signal is not stationary and its statistics are not explicitly known. The predictor must therefore be adapted to the changing signal characteristics in LP coding applications. It is of common practice to consider the speech signal as stationary over short time intervals (about 20 ms). The predictor coefficients can thus be estimated from a sequence of speech samples obtained from an interval over which the signal is considered to be stationary. Windowing the sampled signal is therefore the first step in linear prediction parameter estimation and choosing the right window is an important issue [6]. Depending on the linear predictor form to be employed, the estimated parameters differ. The *autocorrelation method* procedure is employed if windowing is performed on the speech signal whereas the *covariance method* results when windowing is applied on the residual (error) signal.

2.2.1 Autocorrelation Method

In the autocorrelation method, the speech signal $s[n]$ is first multiplied with a data analysis window $w[n]$ of finite length to obtain another signal $s_w[n]$ that is zero outside the window. Considering the case where the window of length N is positioned at zero, we have:

$$s_w[n] = w[n]s[n]. \tag{2.6}$$

Several analysis windows of varying shapes have been proposed, but the Hamming window, which is a raised cosine function, is often used. A tapered analysis window, such as the Hamming window, helps reduce the effect of components outside the window on minimizing the squared prediction error in the first and last few values of $s[n]$ for the current analysis window. It has the following form:

$$w[n] = \begin{cases} 0.54 - 0.46\cos\left(\dfrac{2\pi n}{N-1}\right), & 0 \le n \le N-1, \\ 0, & \text{otherwise.} \end{cases} \tag{2.7}$$

The hybrid Hamming-cosine [13] is another popular window.

The energy of the residual should then be minimized to solve for the LP filter coefficients. Let E be the error energy:

$$E = \sum_{n=-\infty}^{\infty} e^2[n] = \sum_{n=-\infty}^{\infty} \left(s_w[n] - \sum_{k=1}^{p} a_k s_w[n-k] \right)^2 . \tag{2.8}$$

where $e[n]$ is the residual corresponding to the windowed signal $s_w[n]$. The values of a_k that minimize E are found by setting $\frac{\delta E}{\delta a_k} = 0$ for $k = 1,2,3, \ldots ,p$. This yields p linear equations in p unknowns a_k, also know as the Yule-Walker equations:

$$\sum_{k=1}^{p} a_k \sum_{n=-\infty}^{\infty} s_w[n-i]s_w[n-k] = \sum_{n=-\infty}^{\infty} s_w[n-i]s_w[n], \qquad 1 \leq i \leq p. \tag{2.9}$$

Since the autocorrelation of the windowed speech segment $s_w[n]$ is given by

$$R(i) = \sum_{n=i}^{N-1} s_w[n]s_w[n-i], \qquad 0 \leq i \leq p. \tag{2.10}$$

where the autocorrelation function is an even function, $R(i) = R(-i)$. The linear equations can be written as:

$$\sum_{k=1}^{p} R(|i-k|)a_k = R(i), \qquad 1 \leq i \leq p. \tag{2.11}$$

In matrix form, the set of linear equations is represented by $\mathbf{Ra} = \mathbf{v}$ which can be rewritten as

$$\begin{bmatrix} R(0) & R(1) & \ldots & R(p-1) \\ R(1) & R(0) & \ldots & R(p-2) \\ \vdots & \vdots & \ddots & \vdots \\ R(p-1) & R(p-2) & \ldots & R(0) \end{bmatrix} \begin{bmatrix} a_1 \\ a_2 \\ \vdots \\ a_p \end{bmatrix} = \begin{bmatrix} R(1) \\ R(2) \\ \vdots \\ R(p) \end{bmatrix}. \tag{2.12}$$

The resulting matrix has a Toeplitz structure. This facilitates the solution of the Yule-Walker equations for the LP coefficients a_k through computationally fast algorithms such as the Levinson-Durbin [12, 14] and the Schur algorithm [15]. The Toeplitz structure guarantees that $A(z)$ is minimum phase (zeros inside the unit circle) [16]. The corresponding LP synthesis filter $H(z) = 1/A(z)$ is thus stable. This property is a major motivating factor for using the autocorrelation method for LP analysis.

2.2.2 Covariance method

Minimization in the covariance method is performed on the windowed error, which is:

$$E = \sum_{n=-\infty}^{\infty} e_w^2[n] = \sum_{n=-\infty}^{\infty} e^2[n]w[n]. \tag{2.13}$$

By letting the partial derivatives $\frac{\delta E}{\delta a_k} = 0$ for k = 1,2,3, ... ,p, once again we have p linear equations:

$$\sum_{k=1}^{p} \phi(i,k)a_k = \phi(i,0), \qquad 1 \leq i \leq p, \tag{2.14}$$

where the covariance function $\phi(i,k)$ is defined as

$$\phi(i,k) = \sum_{n=-\infty}^{\infty} w[n]s[n-i]s[n-k]. \tag{2.15}$$

The expanded covariance matrix system $\Phi\mathbf{a} = \Psi$ has the form:

$$\begin{bmatrix} \phi(1,1) & \phi(1,2) & \dots & \phi(1,p) \\ \phi(2,1) & \phi(2,2) & \dots & \phi(2,p) \\ \vdots & \vdots & \ddots & \vdots \\ \phi(p,1) & \phi(p,2) & \dots & \phi(p,p) \end{bmatrix} \begin{bmatrix} a_1 \\ a_2 \\ \vdots \\ a_p \end{bmatrix} = \begin{bmatrix} \psi(1) \\ \psi(2) \\ \vdots \\ \psi(p) \end{bmatrix} \tag{2.16}$$

where $\psi(i) = \phi(i,0)$ for i=1,2, ... ,p.

The Covariance matrix preserves its symmetric property and is positive definite but is not necessarily Toeplitz, which makes the method computationally less efficient than the autocorrelation method. Cholesky decomposition [17] is usually used as a first step to solve for the a_k's.

Since the covariance method does not guarantee that the synthesis filter $A(z)$ is minimum phase, a modified covariance method can be used to insure the stability of $A(z)$ [18].

2.2.3 High Frequency Compensation and Bandwidth Expansion

When LP analysis is performed on lowpass filtered speech, the missing high-frequency components near half the sampling frequency can significantly bias the resultant values of the predictor coefficients. In this case, the covariance matrix Φ will produce artificially large predictor coefficients. Therefore, high frequency compensation may be required to correct such problems [19].

Furthermore, LP analysis does not estimate accurately the spectral envelope of high-pitch voiced speech sounds. It may generate synthesis filters with artificially sharp spectral peaks. Usually, we use bandwidth expansion [20, 21] to solve these problems. It has the effect of expanding the bandwidth of the formant peaks in the frequency response. The resulting all-pole filter, with the bandwidth expansion factor γ has the following form:

$$H'(z) = \frac{1}{A'(z)} = \frac{1}{A(\gamma z)} \tag{2.17}$$

where the expanded prediction coefficients are

$$a'_k = a_k \gamma^k, \qquad 1 \leq k \leq p. \tag{2.18}$$

The expansion of the bandwidth can be computed as follows [22]:

$$\Delta B = -\frac{1}{\pi T} \log(\gamma). \tag{2.19}$$

For instance, $\gamma = 0.996$ approximately yields a 10 Hz bandwidth expansion in the analysis of speech sampled at 8 kHz. Typically, bandwidth expansions of 10 to 25 Hz are used for speech analysis.

2.3 Representation of Spectral Parameters

The LP coefficients are calculated on a block-by-block basis, usually at a rate of 50 times per sec. For an efficient transmission, the LP coefficients are subjected to quantization and interpolation. Interpolation makes it possible to transmit the information about the LP coefficients less often (i.e. at a lower frame rate), thus reducing the bit rate. However, both straightforward quantization and interpolation of the LP coefficients are problematic because small changes in the coefficients may result in large changes in the power spectrum, and, possibly, in an unstable LP synthesis filter. Therefore, a number of alternate representations of the LP coefficients have been considered in attempts to find representations which minimize these shortcomings. Some of these representations, which are detailed in the following sections, are, reflection coefficients, log-area ratios and the line-spectral frequencies (LSF's).

2.3.1 Reflection coefficients and log-area ratios

The reflection coefficients are found from the Levinson-Durbin recursive procedure [12, 23] which uses the structure in the Toeplitz R matrix. We solve the following set of ordered equations recursively for $m = 1,2, \ldots ,p$:

$$k_m = \frac{R(m) - \sum_{k=1}^{m-1} a_{m-1}(k) R(m-k)}{E_{m-1}} \tag{2.20a}$$

$$a_m(m) = k_m, \tag{2.20b}$$

$$a_k(m) = a_k(m-1) - k_m a_{m-k}(m-1), \qquad 1 \leq k \leq m-1 \tag{2.20c}$$

$$E_m = (1 - k_m^2) E_{m-1}. \tag{2.20d}$$

where initially $E_0 = R(0)$ and $a_0 = 0$. At each cycle m, the coefficients $a_m(k)$ describe the optimal m-th order linear predictor. Since E_m, a squared error, is never negative, $|k_m| < 1$. This condition on the reflection coefficients also guarantees a stable LP synthesis filter $H(z)$. The negatives of the reflection coefficients are called *partial correlation*, or PARCOR, coefficients.

One can find the reflection coefficients from the LP coefficients $a_k = a_p(k)$, by recursively computing the following two equations for $m = p, p-1, \ldots, 3, 2$:

$$a_{m-1}(i) = \frac{a_m(i) k_m a_m(m-i)}{1-k_m^2}, \qquad 1 \le i \le m-1 \tag{2.21a}$$

$$k_{m-1} = a_{m-1}(m-1) \tag{2.21b}$$

With this backward recursion, predictor coefficients can be checked for stability by converting them to reflection coefficients and then using the reflection coefficient stability test.

A drawback of the reflection coefficients is the U-shape of their spectral sensitivity which has large values whenever the magnitude of the coefficients is close to unity. However this drawback can be overcome by the use of an appropriate nonlinear transformation which expands the region near $|k_m| = 1$. Two such transformations are the *log-area ratio* transformation [21] and the inverse sine transformation [24]. The log-area ratios (LARs) are defined as:

$$g_m = \log\left(\frac{1+k_m}{1-k_m}\right), \qquad 1 \le m \le p. \tag{2.22}$$

To convert back to reflection coefficients,

$$k_m = \frac{\mathrm{e}^{g_m}-1}{\mathrm{e}^{g_m}+1}, \qquad 1 \le m \le p. \tag{2.23}$$

and the arcsine reflection coefficients (ASRCs) are defined as:

$$j_m = \arcsin(k_m), 1 \le m \le p. \tag{2.24}$$

2.3.2 Line Spectral Frequencies

Another representation of the LP parameters, called line spectral frequencies (LSF's) or line spectrum pairs (LSP's), was introduced by Itakura [25]. The LSF procedure involves mapping the p zeros of $A(z)$ onto the unit circle through two z-transforms $P(z)$ and $Q(z)$ of $(p+1)$st order:

$$P(z) = A(z) + z^{-(p+1)}A(z^{-1}) \tag{2.25}$$

$$Q(z) = A(z) - z^{-(p+1)}A(z^{-1}) \tag{2.26}$$

It directly follows that:

$$A(z) = \frac{1}{2}[P(z) + Q(z)]. \tag{2.27}$$

Soong and Juang [26] have shown that if $H(z)$ is stable, or $A(z)$ is minimum phase, then all the roots of $P(z)$ and $Q(z)$ lie on the unit circle, alternating between the two polynomials as ω increases. The LSF's correspond to these angular positions. The roots occur in complex-conjugate pairs and hence there are p LSF's lying between 0 and π. The process produces two extraneous zeros at $\omega = 0$ and $\omega = \pi$ which can be ignored. It has also been shown [26] that if the p line spectral frequencies ω_i have an ascending ordering property and are unique, then the inverse prediction filter $A(z)$ is guaranteed to have minimum phase (stable corresponding synthesis filter):

$$0 < \omega_1 < \omega_2 < \cdots < \omega_p < \pi \quad \text{[radians / s]}. \tag{2.28}$$

An additional condition, known to always be true, was added by Gnanasekaran [27]. It states that $|a_p|$ has to be less than unity to have a stable filter.

The LSF's may be found by applying a discrete cosine transformation [26, 28] to the coefficients of the polynomials

$$G(z) = \begin{cases} \dfrac{P(z)}{1+z^{-1}}, & p \text{ even}, \\ P(z), & p \text{ odd}. \end{cases} \tag{2.29}$$

and

$$H(z) = \begin{cases} \dfrac{Q(z)}{1 - z^{-1}}, & p \text{ even}, \\ \dfrac{Q(z)}{1 - z^{-2}}, & p \text{ odd}. \end{cases} \tag{2.30}$$

The roots, corresponding to the LSF's, are found by searching along the $\omega = [0, \pi]$ range iteratively for the changing sign in the polynomials $G(z)$ and $H(z)$. Another method by Kabal and Ramachandran [29] makes use of Chebyshev polynomials

$$T_m(x) = \cos(m\omega) \tag{2.31}$$

where $x = \cos \omega$ maps the upper semi-circle in the z-plane to the real-valued interval $[-1, 1]$. The polynomials $G'(\omega)$ and $H'(\omega)$ can then be expanded as

$$G'(x) = 2 \sum_{i=0}^{l} g_i T_{l-i}(x), \tag{2.32}$$

$$H'(x) = 2 \sum_{i=0}^{m} h_i T_{m-i}(x), \tag{2.33}$$

where $l = m = p/2$ when p is even, and $l = (p+1)/2$ and $m = (p-1)/2$ when p is odd. The roots of these Chebyshev expansions will give the LSF's after the inverse transformation $\omega = \arccos(x)$. The roots are also determined by searching for sign changes of the Chebyshev expansions along the interval $[-1, 1]$.

Several important properties of the LSF can be seen from an LPC spectrum (see Fig. 2.2):

- The spacing of the LSF parameters determines the magnitude of the power spectral density.
- A cluster of two or three LSF's usually signals a formant frequency or a peak while widely spaced LSF values correspond to valleys.
- The bandwidth of a formant depends on the closeness of the corresponding LSF's.

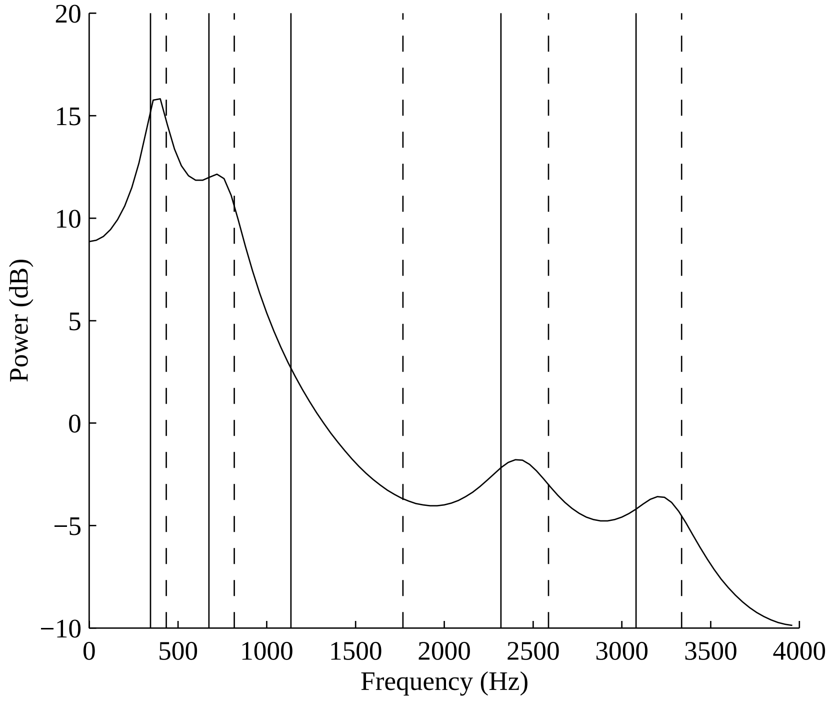


Fig. 2.2 LPC spectra of a 20 ms speech frame with the corresponding LSF's displayed in Hertz (vertical lines).

- In general, the spectral sensitivity of each LSF is localized, i.e. only the neighborhood near the LSF will alter if the value is slightly modified.

As a result of those properties, the LSF parameters can be coded efficiently to produce a low overall distortion. In addition, since higher frequency formants contribute negligible perceived speech distortion, the LSF parameters representing them can be quantized less accurately. On the other hand, the peaks must be coded more precisely.

2.3.3 Cepstral Coefficients

The cepstrum [30] is the inverse Fourier transform of the logarithm of the magnitude spectrum of a signal:

$$c_n = \frac{1}{2\pi} \int_{-\pi}^{\pi} \log H(\mathrm{e}^{\mathrm{j}\omega}) \mathrm{e}^{\mathrm{j}\omega \mathrm{n}} \mathrm{d}\omega \tag{2.34}$$

As a consequence of the fact that $H(z)$ is a minimum phase filter we know that $c_n = 0$ for $n \leq 0$.

Furthermore, an infinite number of cepstral coefficients can be computed from prediction coefficients [31]:

$$c_n = a_n + \sum_{k=1}^{n-1} \frac{k}{n} a_{n-k} c_k \tag{2.35}$$

For a p-th order linear predictor, $a_n = 0$ for $n > p$. It has been found that limiting the cepstral coefficients to three times the number of predictor coefficients is sufficient to provide a good representation of the speech spectrum [32].

2.4 Distortion Measures

Distortion measures play an important role in speech coding. One use of the distortion measures is to evaluate the performance of speech coding systems. The ultimate evaluator of a speech coder's quality and performance in preserving intelligibility and naturalness is the human auditory system. However, extensive perceptual performance of speech coders is time consuming and might be inconsistent. Objective measurements can give an immediate and reliable estimate of the perceptual quality of a coding algorithm [33]. Objective measurements also play a critical role in LP coders that use vector quantization to code the coefficients. Since these coefficients model the spectral envelope of the speech for a short frame of data, one has to select the best matching spectral envelope from the vector quantization codebook for a given vector. What constitutes the best match is the perceptual similarity of the codebook vector to the given vector. Hence quantitative distortion measures are essential to evaluate the perceptual closeness between two spectral envelopes. Presented below are several objective distortion measures.

2.4.1 Time-Domain Measures

The signal-to-noise ratio (SNR) and the segmental SNR (SNRseg) are the most common time-domain measures of the difference between original and coded speech signals.

Signal-to-Noise Ratio

The signal-to-noise (SNR) can be defined as the ratio between the input signal power and the noise power, and is given in decibels (dB) as:

$$\mathrm{SNR} = 10\log_{10}\frac{E_s}{E_\varepsilon} = 10\log_{10}\frac{\sum\limits_{n=-\infty}^{\infty} s^2[n]}{\sum\limits_{n=-\infty}^{\infty}(s[n]-\hat{s}[n])^2}\ \mathrm{dB}, \tag{2.36}$$

where $\hat{s}[n]$ is the coded version of the original speech sample $s[n]$. The principal benefit of the SNR quality measure is its mathematical simplicity. The measure represents an average error over time and frequency for a processed signal. However SNR is a poor estimator for a broad range of speech distortions [34, 35]. The fact that SNR is not particularly well related to any subjective attribute of speech quality and that it weights all time domain errors in the speech waveform equally, makes it a poor measure.

Segmental Signal-to-Noise Ratio

A much-improved quality measure can be obtained if SNR is measured over short frames and the results averaged. The frame-based measure is called the segmental SNR (SNRseg) and is formulated as:

$$\mathrm{SNRseg} = \frac{1}{M}\sum_{j=0}^{M-1} 10\log_{10}\left[\sum_{n=m_j-N+1}^{m_j}\frac{s^2[n]}{(s[n]-\hat{s}[n])^2}\right]\ \mathrm{dB}. \tag{2.37}$$

where $m_0, m_1, \ldots, m_M - 1$ are the end-times for the m frames, each of which is length N. The segmentation of the SNR permits the objective measure to assign equal weights to loud and soft portions of the speech. In some cases, problems can arise with the SNRseg measure if frames of silence are included, since large negative SNR values bias the overall measure of SNRseg. A threshold can be used to exclude any frames that contain unusually high or low SNR values (below 0 dB and over 35 dB are termed unusual). An extension to the segmental SNR is the frequency weighted segmental SNR measure [36] and it can be used to match the listener's perception of quality. It has the following form:

$$\text{SNRfw} = \frac{1}{M}\sum_{j=0}^{M-1}\left[\frac{\sum_i W(j,i) 10\log_{10}\left[\frac{s^2[n]}{(s[n]-\hat{s}[n])^2}\right]}{\sum_i W(j,i)}\right] \quad \text{dB}. \tag{2.38}$$

where the j is the segment index, i is the frequency band index and $W(j,i)$ is the frequency weight.

2.4.2 Spectral Envelope Distortion Measures

A spectral distortion measure is a function of two spectral densities, f and $\hat{f}$ for example, which assigns a nonnegative number $d(f,\hat{f})$ to represent the distortion in using $\hat{f}$ to represent f. Another property is that if $f = \hat{f}$, then $d(f,\hat{f}) = 0$. The most common such measures are difference distortion measures where one uses an L_p norm on the difference $f - \hat{f}$. These are metrics or distances in the sense that they satisfy a symmetry requirement $d(f,\hat{f}) = d(\hat{f},f)$ and a triangular inequality:

$$d(f,g) \leq d(f,h) + d(h,g). \tag{2.39}$$

Usually in spectral envelope distortion measures, f is defined as a vector $\mathbf{x}$ and $\hat{\mathbf{x}}$ as its representation. The overall performance measure is then the long term average of a distortion measure and can be expressed as follows:

$$D = \lim_{n\to\infty}\frac{1}{n}\sum_{i=1}^{n} d(\mathbf{x}_i, \hat{\mathbf{x}}_i). \tag{2.40}$$

In general, the spectral distortion should measure the discrepancies between the original and coded spectral envelopes that will lead to sounds being distinguished as phonetically different [37]. The disparities between the original and coded spectral envelopes include the following:

- Significantly different center frequencies for the resonances or formants of the original and coded spectral envelopes.

- Alteration of the formant bandwidths caused by the coded spectral envelopes.

The log spectral distortion measure, the Itakura-Saito measure, the cepstral distance and the weighted Euclidean distance measure are some of the several spectral distortion measures proposed and are discussed below.

Log Spectral Distortion Measure

The L_p norm-based log spectral distance measure is

$$d_{\mathrm{SD}}^{p} = \frac{2}{2\pi} \int_{-\pi}^{\pi} \left| 10 \log_{10} S(\omega) - 10 \log_{10} \hat{S}(\omega) \right|^{p} d\omega \tag{2.41}$$

where the frequency magnitude spectrum $S(\omega)$ is

$$\begin{aligned} S(\omega) &= \frac{G}{|A(\mathrm{e}^{j\omega})|^2} && (2.42) \\ &= \frac{G}{[1 - \sum_{n=1}^{p} a_n \mathrm{e}^{jn\omega}]^2}. && (2.43) \end{aligned}$$

G is the LP filter gain factor, and $\{a_n\}$ are the LP coefficients.

When $p = 2$, we have the L_2 norm or root mean square (rms) log spectral distortion measure. The rms log spectral distortion measure is defined in dB as

$$d_{\mathrm{SD}} = \sqrt{\frac{1}{\omega_u - \omega_l} \int_{\omega_l}^{\omega_u} \left[10 \log_{10} \frac{S(\omega)}{\hat{S}(\omega)} \right]^2 d\omega} \ \ \mathrm{dB} \tag{2.44}$$

where ω_l and ω_u define the lower and upper frequency limits of integration. Ideally, ω_l is equal to zero and ω_u corresponds to half the sampling frequency.

In practice, the rms log spectral distance is calculated discretely over a limited bandwidth. For speech signal sampled at 8 kHz with a 3 kHz bandwidth, the rms log spectral distortion (SD) is usually calculated as a summation, with a resolution of approximately 31.25 Hz per sample, over 96 uniformly spaced points from 125 Hz to 3.125 kHz[†] [38]. This can be expressed as

$$\mathrm{SD} = \sqrt{\frac{1}{n_1 - n_0} \sum_{n=n_0}^{n_1-1} \left[10 \log_{10} \frac{S(e^{j2\pi n/N})}{\hat{S}(e^{j2\pi n/N})} \right]^2} \quad \mathrm{dB} \tag{2.45}$$

where for $N = 256$, n_0 and n_1 correspond to 1 and 96 respectively.

Paliwal and Atal [43] have suggested that the average spectral distortion is not adequate to measure perceived quality alone. They introduced the notion of spectral outliers which represent the fraction of frames with large spectral distortion. For transparent quality, the fraction of frames having SD between 2 and 4 dB should be less that 2 % with no frames having SD greater than 4 dB.

Itakura-Saito Distortion Measure

The Itakura-Saito measure generally corresponds better to the perceptual quality of speech. Also known as a likelihood ratio distance measure, it measure the energy ratio between the residual signal that results when using the quantized LP filter and the one that results when using the unquantized LP filter. It is defined as follows:

$$d_{\mathrm{IS}} = \frac{1}{2\pi} \int_{-\pi}^{\pi} \left[\mathrm{e}^{V(\omega)} - V(\omega) - 1 \right] d\omega \tag{2.46}$$

where the log spectral difference $V(w)$ between the two spectra is defined as

$$V(\omega) = \log S(\omega) - \log \hat{S}(\omega). \tag{2.47}$$

Evaluating the integrals, this measure can be expressed as the polynomial

$$d_{\mathrm{IS}} = \left(\frac{G}{\hat{G}} \right)^2 \frac{\hat{\mathbf{a}}^T \mathbf{R} \hat{\mathbf{a}}}{\mathbf{a}^T \mathbf{R} \mathbf{a}} - 2 \log \left(\frac{G}{\hat{G}} \right) - 1 \tag{2.48}$$

[†]In this thesis, even though some of the test speech files have a larger bandwidth than 3 kHz, we decided to use this approximation for the rms log spectral distortion to be consistent with figures given in the literature.

where $\hat{\mathbf{a}} = [1, \hat{a}_1, \hat{a}_2, \ldots, \hat{a}_p]^T$, $\mathbf{a} = [1, a_1, a_2, \ldots, a_p]^T$, and $\mathbf{R}$ is the autocorrelation matrix. When the gains are assumed to be equal, the Itakura-Saito measure is simply

$$d_{\mathrm{IS}} = \frac{\hat{\mathbf{a}}^T \mathbf{R} \hat{\mathbf{a}}}{\mathbf{a}^T \mathbf{R} \mathbf{a}} - 1. \tag{2.49}$$

However, the Itakura-Saito measure is not symmetric. For symmetry, a modified Itakura measure can be used:

$$d_{\mathrm{IS}} = \frac{1}{2} \left[\frac{\hat{\mathbf{a}}^T \mathbf{R} \hat{\mathbf{a}}}{\mathbf{a}^T \mathbf{R} \mathbf{a}} - \frac{\mathbf{a}^T \hat{\mathbf{R}} \mathbf{a}}{\hat{\mathbf{a}}^T \hat{\mathbf{R}} \hat{\mathbf{a}}} - 2 \right]. \tag{2.50}$$

A weighting term can be introduced to the Itakura-Saito measure to take advantage of the perceptual discrimination properties of the human ear and is formulated as follows:

$$d_{\mathrm{WIS}} = \frac{1}{2\pi} \int_{-\pi}^{\pi} W(e^{j\omega}) \left[\mathrm{e}^{V(\omega)} - V(\omega) - 1 \right] d\omega \tag{2.51}$$

Some weighting schemes $W(e^{j\omega})$ are proposed in [39].

Log-Area Ratio Measure

The log-area ratio measure is based on the set of reflection coefficients and defined as:

$$d_{\mathrm{LAR}} = \sum_{n=1}^{p} \left[\log \frac{1 - k_i}{1 + k_i} - \log \frac{1 - \hat{k}_i}{1 + \hat{k}_i} \right]^2 . \tag{2.52}$$

with k_i being the set of p reflection coefficients and $\hat{k}_i$ their quantized counterpart.

Cepstral Distance

The L_2 cepstral distance is defined as:

$$d_{\mathrm{CD}}^2 = \sum_{n=-\infty}^{\infty} (c_n - \hat{c}_n)^2 \tag{2.53}$$

and is directly related to the rms log spectral distance:

$$\begin{aligned} d_{\mathrm{CD}}^2 &= 2\sum_{n=1}^{\infty}(c_n - \hat{c}_n)^2 && (2.54) \\ &= \frac{1}{2\pi}\int_{-\pi}^{\pi} \left|\log S(\omega) - \log \hat{S}(\omega)\right|^2 d\omega. && (2.55) \end{aligned}$$

using Parseval's equality and the fact that $c_n = c_{-n}$ and $c_0 = 0$

The log spectral distortion measure suffers from the drawback that Fourier transform and logarithm computations are required for each point in the summation. The cepstral distance (d_{CD}) can be computed efficiently by truncating the summation to a finite number of terms N_c, usually three times the order of the LP analysis filter p:

$$d_{\mathrm{CD}} = 10 \log_{10} \mathrm{e} \sqrt{2\sum_{n=1}^{N_c}(c_n - \hat{c}_n)^2} \ \ \mathrm{dB}. \tag{2.56}$$

The introduction of a weighting term in the cepstral distance has been investigated by several researchers:

$$d_{\mathrm{CD}} = 10 \log_{10} \mathrm{e} \sqrt{2\sum_{n=1}^{N_c} w_n (c_n - \hat{c}_n)^2} \ \ \mathrm{dB}. \tag{2.57}$$

where the weighting term can be:

- $w(n) = n^2$, called quefrency weighted cepstral distance [40].
- $w(n) = 1/v(n)$, $v(n)$ being the variance of the cepstral coefficients [41].
- $w(n) = 1 + 0.75p\sin(\pi n/1.5p)$ used in speech recognition experiments [42].

Weighted Euclidean LSF Distance Measure

The Euclidean distance measure between two vectors is simply defined as:

$$d(\mathbf{x}, \hat{\mathbf{x}}) = (\mathbf{x} - \hat{\mathbf{x}})^T (\mathbf{x} - \hat{\mathbf{x}}) = \|\mathbf{x} - \hat{\mathbf{x}}\|^2. \tag{2.58}$$

In general, we minimize this mean squared error between the unquantized and quantized vector to select the best codeword vector. Since line spectral frequencies (LSF's) have a direct relationship to the shape of the spectral envelope, we associate them with this measure.

The Euclidean distance measure allocates equal weights to individual components of the LSF vector. Spectral sensitivities can be taken into account with a weighted Euclidean distance. In general, the weighted Euclidean LSF distance measure looks like:

$$d(\mathbf{x}, \hat{\mathbf{x}}) = (\mathbf{x} - \hat{\mathbf{x}})^T \mathbf{W} (\mathbf{x} - \hat{\mathbf{x}}) \tag{2.59}$$

where $\mathbf{W}$ is a $m \times m$ symmetric and positive definite weighting matrix which may be dependent on $\mathbf{x}$. If $\mathbf{W}$ is a diagonal matrix with elements $w_{ii} > 0$, the distance can also be expressed as

$$d(\mathbf{x}, \hat{\mathbf{x}}) = \sum_{i=1}^{m} w_{ii} (x_i - \hat{x}_i)^2. \tag{2.60}$$

In [43], Paliwal and Atal have proposed a weighted Euclidean distance measure in the LSF domain which tries to assign weights to individual LSF's according to their spectral sensitivities. The weighted Euclidean distance measure $d(\mathbf{x}, \hat{\mathbf{x}})$ between the test LSF vector $\mathbf{x}$ and the reference LSF vector $\hat{\mathbf{x}}$ is given by:

$$d(\mathbf{x}, \hat{\mathbf{x}}) = \sum_{i=1}^{p} \left[c_i w_i (x_i - \hat{x}_i)^2 \right]. \tag{2.61}$$

where x_i and $\hat{x}_i$ are the i-th LSF's in the test and reference vector, respectively, and c_i and w_i are the weights assigned to the i-th LSF. For a 10'th order LSF vector, these weights are given by:

$$c_i = \begin{cases} 1.0, & \text{for } 1 \leq i \leq 8, \\ 0.8, & \text{for } i = 9, \\ 0.4, & \text{for } i = 10. \end{cases} \tag{2.62}$$

and

$$w_i = [S(\omega_i)]^r \tag{2.63}$$

where $S(\omega_i)$ is the LPC power spectrum associated with the test vector as a function of frequency ω_i, and r is an empirical constant which controls the relative weights given to different LSF's and is determined experimentally. Paliwal and Atal found that a value of r equal to 0.15 provides the best performance. It should be noted that the weights w_i vary from frame-to-frame depending on the LPC power spectrum, while the weights c_i do not change from frame-to-frame. These two types of weighting are called the adaptive weighting and the fixed weighting, respectively.

Our performance results, shown in Table 2.1 shows that the average spectral distortion with weighted LSF's is lower than the non-weighted LSF's. The spectral distortion (SD) is computed using Equation (2.45) on a testing and a training sets when applied on a split vector quantizer (refer to Chapter 3 for details). Figure 2.3 demonstrates the fact that LSF values with correspondingly high power and located in the lower spectrum are quantized more efficiently. Hence, a lower SD is obtained.

Table 2.1 Average spectral distortion showing the difference between having a Euclidean distance and a weighted Euclidean distance.

Type	Test Set			Training Set		
of	Average	SD Outliers (%)		Average	SD Outliers (%)	
Weighting	SD (dB)	2-4 dB	> 4 dB	SD (dB)	2-4 dB	> 4 dB
Weighting	1.21	6.21	0.02	1.12	3.88	0.01
No Weighting	1.39	13.79	0.12	1.28	10.38	0.09

Other adaptive weighting schemes based on the properties of LSF's have been

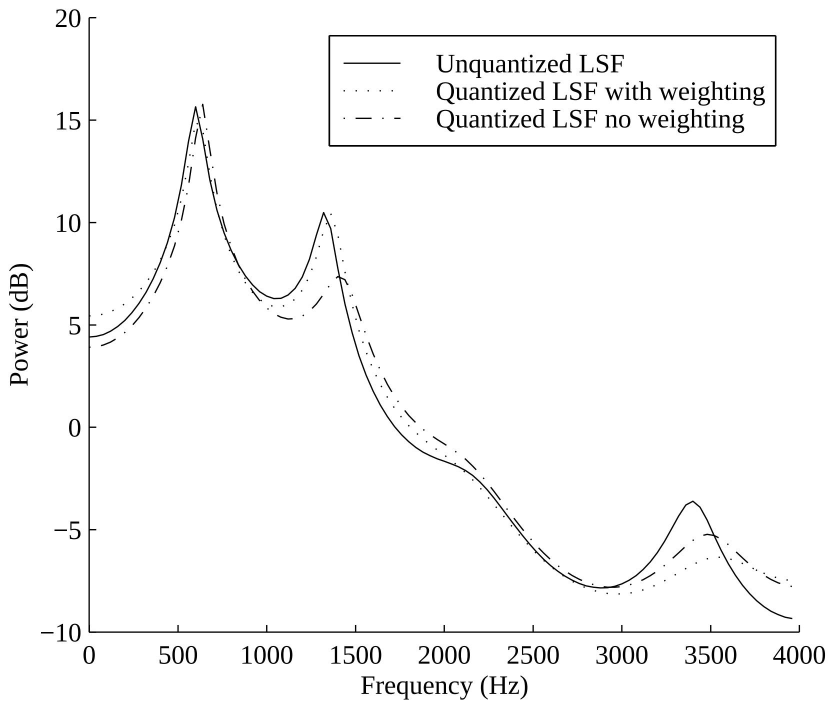


Fig. 2.3 Effect of using weights to search for LSF values on the LP spectrum.

proposed [44]. Laroia *et al* [45] suggested using

$$w_i = \frac{1}{\omega_i - \omega_{i-1}} + \frac{1}{\omega_{i+1} + \omega_i} \tag{2.64}$$

where $\omega_0 = 0$ and $\omega_p = \pi$. Adopting the weighting matrix in [46], Leblanc *et al* [47] reported slightly better performance than the weights suggested by [43] and significantly better performance than the weights used in [45].

2.5 Database

In order to compare the performance of all the different VQ's, a common speech database has to be used. The performance results of spectral coding are based on a *training set* and a separate *testing set* of LSF vectors [48]. The speech database contains a list of phonetically balanced sentences taken from [49]. Approximately 54 minutes of silence-removed speech is recorded by 24 different speakers, half males and half females. The first 2900 seconds of speech are used for training and 340 seconds of speech are used for testing. The original speech was initially recorded with a 48 kHz sampling frequency and then downsampled to 8 kHz with proper lowpass filtering at 4.0 kHz.

A 10'th order LP analysis, based on the autocorrelation method is performed every 20 ms using a 20-ms non-overlapping Hamming analysis window. Thus, there are 145388 LSF vectors for training and 16189 LSF vectors for testing. To avoid sharp spectral peaks in the LPC spectra, a 10 Hz bandwidth expansion is applied (i.e. $\gamma = 0.996$). The LSF's can be converted into other parametric representation such as reflection coefficients or predictor coefficients. Furthermore, different filtering conditions had to be imposed on the database, (i.e. lowpass, bandpass and highpass) to obtain speech with different bandwidths. The filter power spectra are shown in Fig. 2.4. Different cutoff frequencies are also used (Fig. 2.4a)). These filters have cutoff frequencies representative of the anti-aliasing filters used in telephony.

The rms log spectral distortion (SD) measure is used as the primary objective indicator of perceptual coding efficiency for both the training set and test set spectral parameters. Since an SD measure per frame is computationally intensive, the weighted Euclidean LSF distance measure using the weights proposed in [43] is employed during the design and operation of the LSF spectral parameters encoder.

As only the effects of spectral coding are considered in this work, the original residual signal passes directly from the encoder to the decoder. Any degradation in the reconstructed speech signal will be solely attributed to the effects of spectral quantization.

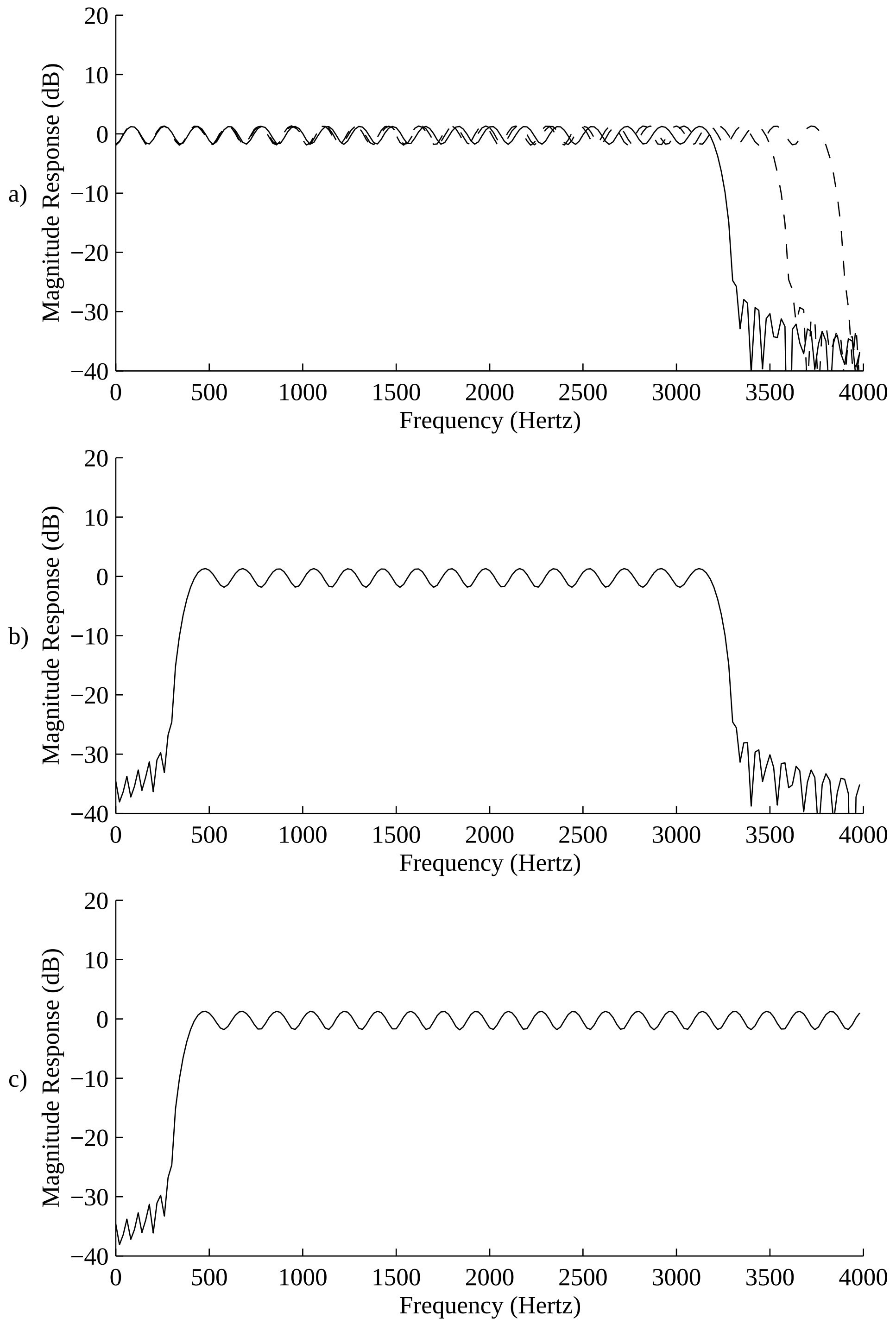


Fig. 2.4 Frequency response of the (a) lowpass filter (different cutoff frequencies), (b) bandpass , (c) highpass filter used in our studies.

Chapter 3

Codebook Transformation Model in VQ

In this chapter, we introduce the notion of vector quantization which is widely used in low bit-rate state-of-the-art speech coders. In particular, encoding speech spectral parameters on a frame-by-frame basis with VQ is known to be an efficient step in speech compression. An introduction to VQ and its various codebook design methodologies will be given. Furthermore, after discussing the drawbacks faced in VQ when used to quantize LSF vectors, we will explain the codebook transformation method. Finally, two different mapping techniques and their limitations are described.

3.1 VQ Design

3.1.1 Introduction

Scalar quantization is the simplest interpretation of VQ. This process is known to be defined when each of a set of parameters (or a sequence of signal values) is quantized separately. In general, scalar quantization assigns to an input value x the closest approximating value from a predetermined finite set, or codebook, of N permissible output values $C = \{y_k \mid k = 1, \ldots, N\}$. If we need to code an m-dimensional spectral parameter vector $\mathbf{x}$ using scalar quantization, we have to independently quantized each vector element x_i in $\mathbf{x}$ as:

$$\hat{x}_i = y_{i,k} = Q_i(x_i), \qquad i = 1, \ldots, m, \tag{3.1}$$

where each of the quantizers $Q_i()$ may be designed separately. The uniform quantizer is the most common of all scalar quantizers. The input-output response is a uniform staircase where the step size is constant. Nonuniform spacing of quantization levels is sometimes used to accommodate a larger dynamic range. Furthermore, it is possible to design a quantizer tailored to a specific input statistics (usually trained with the LBG algorithm discussed in Section 3.1.3). In adaptive quantization, the step breakpoints and output levels are all scaled by a multiplier [50].

VQ is a multi-dimensional extension of scalar quantization in which coding can be performed over the whole parameter set as a single vector. Conceptually, it consists in finding from a codebook of pre-determined parameter vectors, the vector that "matches" best the set of parameters computed for a frame of speech. Once this codevector is found, its index is transmitted to the decoder which contains the replica of the quantization codebook (see Fig. 3.1).

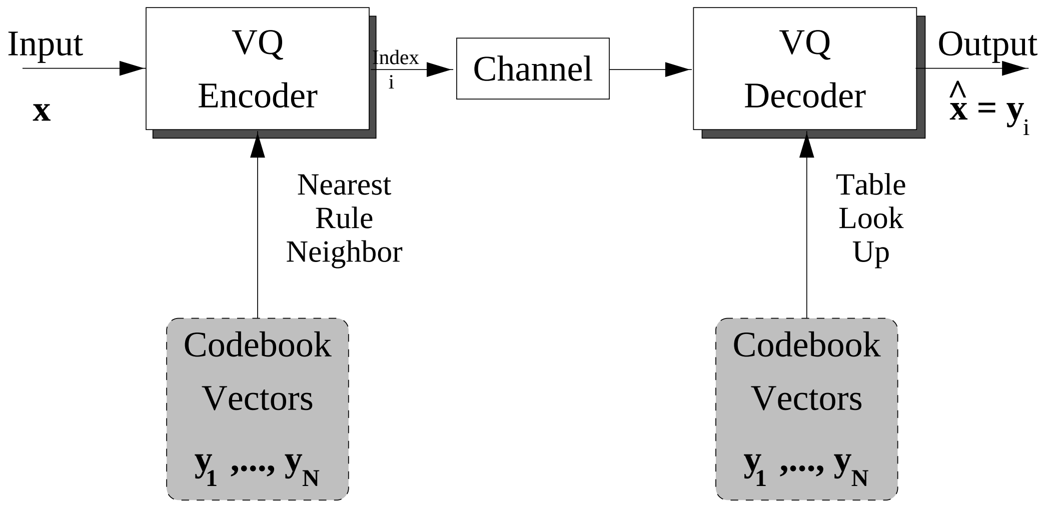

Input
x
VQ Encoder
Index i
Channel
VQ Decoder
Output
$\hat{\mathbf{x}} = \mathbf{y}_i$
Nearest Rule Neighbor
Table Look Up
Codebook Vectors $\mathbf{y}_1, \ldots, \mathbf{y}_N$
Codebook Vectors $\mathbf{y}_1, \ldots, \mathbf{y}_N$

Fig. 3.1 Model of a Vector Quantizer.

As seen in Fig. 3.2, the encoder partitions the input vector multidimensional space

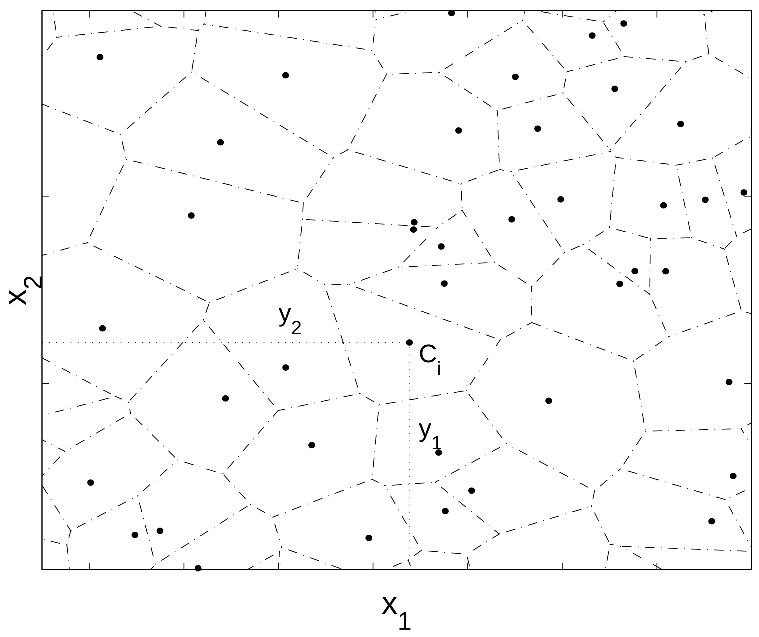


Fig. 3.2 Example of partitioning of a two dimensional space ($N = 2$). All input vectors in cell C_i is quantized as the code vector $\mathbf{y} = [y_1, y_2]$.

into L regions as $P = \{C_1, C_2, \ldots, C_L\}$ where

$$C_i = \{\mathbf{x} \mid d(\mathbf{x}, \mathbf{y}_i) \leq d(\mathbf{x}, \mathbf{y}_j),\ j \neq i\}. \tag{3.2}$$

All vectors lying in a region C_i will be quantized to the set vector $\mathbf{y}_i$. The shapes of the various regions can be very different (squares, hexagons, irregular shapes, etc.) depending on the actual codebook used in the encoder.

Vector quantization always attains as good or better performance in terms of lower distortion than scalar quantization due to its ability to exploit any correlation (linear or non-linear) among the vector components, and to fit the shape of the vector source density [7, 51, 52]. Next, we will discuss the optimality issue in VQ and different designs of VQ.

3.1.2 Optimality Conditions for VQ

The partition space of the encoder and the reproduction vectors of the decoder are directly related to the performance of VQ. Optimality is defined as the minimization of a distortion measure $E[d(\mathbf{X}, \hat{\mathbf{X}})]$ for any input vector sequence $\mathbf{X}$. Three necessary conditions for codebook optimality need to be satisfied during the design: one for the encoder and two for the decoder. These conditions are the *Nearest Neighbor Condition*, the *Centroid Condition* and finally the *Zero Probability Boundary Condition* [53].

Nearest Neighbor Condition

For a given set of output levels, $\mathbf{C}$, the optimal partition cells satisfy

$$R_i \subset \{\mathbf{x} \mid d(\mathbf{x}, \mathbf{y}_i) \leq d(\mathbf{x}, \mathbf{y}_j); \forall j\}. \tag{3.3}$$

That is to say the partition regions are defined by the codevectors $\{\mathbf{y}_i\}$ in $\mathbf{C}$:

$$Q(\mathbf{x}) = \mathbf{y}_i \ \text{ only if } \ d(\mathbf{x}, \mathbf{y}_i) \leq d(\mathbf{x}, \mathbf{y}_j) \ \forall j. \tag{3.4}$$

If an input vector $\mathbf{x}$ is equally distant from two or more code vectors, an arbitrary “tie-breaking” rule may be defined.

Centroid Condition

Given an encoder partition $P = \{R_i \mid i = 1, \ldots, N\}$, the optimal codevectors $\mathbf{y}_i$ in $\mathbf{C}$ are the centroids in each partition cell R_i:

$$\begin{aligned} \mathbf{y}_i &= \text{cent}(R_i) \qquad (3.5) \\ &= \arg\min_{\mathbf{y}} \ E[d(\mathbf{x}, \mathbf{y}) | \mathbf{x} \in R_i]. \qquad (3.6) \end{aligned}$$

When the squared error distortion measure is used for VQ design, the centroids are the centers of mass of the partition cells.

Zero Probability Boundary Condition

If we define B_j as the boundary points of R_j, a necessary condition for optimality is that the boundary points occur with zero probability:

$$P(\bigcup_{j=1}^{N} B_j) = 0. \tag{3.7}$$

3.1.3 Codebook Design

Since no closed-form solutions of the problem of optimal quantization is known, in general, one would have to work iteratively to improve a given vector quantizer provided that the necessary conditions for optimality are met. The iteration begins with a VQ consisting of its codebook and the corresponding optimal Nearest Neighbor (NN) partition and then finds the new codebook which is optimal for that partition. This new codebook and its NN partitions are then a new vector quantizer with average distortion no greater (and usually less) than the original quantizer. This repeated application of the improvement step yields an iterative algorithm which reduces the average distortion at each step.

Initial Codebook

Before improving on a codebook, one needs to start with an initial one. Obtaining a suitable initial codebook is a crucial step for an effective VQ design. There are a variety of techniques for generating a codebook, and here are several of the most useful ones.

Random Coding is the simplest approach to filling a codebook of N code words. The idea is to select randomly the code words according to the source distribution. If the data is highly correlated, a better codebook is produced if one takes, say, every Kth training vector. *Pruning* is another concept for an initial codebook. It refers to the idea of starting with the training set and selectively eliminating (pruning) training vectors as candidate code vector until a final set of training vectors remains as the codebook. A more complicated, but better, method is the *pairwise nearest neighbor* (PNN) clustering algorithm proposed by Equitz [54]. The goal of this method is to merge vectors together into groups or clusters until we have the desired number, say

N. The codebook will then contain the centroids of these clusters. *Product Codes* and *Splitting* are also methods to find a good startup codebook. A variation of the Katsavounidis *et al* [55] technique, named *centroid*, similar to pruning is used for the purpose of this thesis and is as follows:

STEP 1 Calculate the centroid of the training set.

STEP 2 Calculate the unweighted Euclidean distance from each training vector to the centroid.

STEP 3 Choose the training vector with the maximum distance from the centroid as the reference vector.

STEP 4 Calculate the unweighted Euclidean distance from each training vector to the reference vector.

STEP 5 Let M/N be the ratio of training set vectors to codebook vectors. Find the M/N vectors closest to the reference vector, and calculate the centroid (initial codebook vector) for this group of vectors.

STEP 6 Reduce the training set by removing the group of vectors found in Step 5.

STEP 7 Repeat Steps 2 to 7 until no vectors remain in the training set.

Three different initial codebooks are compared in Table 3.1. The initial codebooks contain LSF vector entries obtained from a training set. The average spectral distortion is calculated on a separate testing set before and after training. The results shows that the *pruning* initial codebook has a fewer number of iterations and a smaller percentage of outliers. However, the *centroid* initial codebook gives lower spectral distortion.

Generalized Lloyd Algorithm

After setting up the initial codebook, an iterative algorithm is used to improve the VQ. The *Generalized Lloyd Algorithm* (GLA), also known as the LBG algorithm [56], is perhaps the most commonly used iterative clustering algorithm for optimal VQ design based on training vectors:

Table 3.1 Average spectral distortion of different initial codebooks. The distortion is calculated before training and after training (for a 2-SVQ).

Initial Codebook Type	Before Training Average SD (dB)	Before Training SD Outliers (%) 2-4 dB	Before Training SD Outliers (%) > 4 dB	After Training Average SD (dB)	After Training SD Outliers (%) 2-4 dB	After Training SD Outliers (%) > 4 dB	Num. of Iter
Random	1.56	21.52	0.96	1.23	6.25	0.05	73
Pruning	1.43	9.76	0.04	1.22	5.00	0.01	32
Centroid	1.35	12.61	0.05	1.21	6.21	0.02	59

STEP 1 Start with an initial codebook $\mathbf{C}_1$. Let $m = 1$.

STEP 2 Given the codebook $\mathbf{C}_m$, perform the Lloyd iteration to produce the new codebook $\mathbf{C}_{m+1}$.

STEP 3 Compute the average distortion for $\mathbf{C}_{m+1}$. If it has changed by a small enough amount since the last iteration, stop. Otherwise, let $m + 1 \longrightarrow m$ and go to Step 2.

where **STEP 2** is defined as:

step 2a Given a codebook $\mathbf{C}_m = \{\mathbf{y}_i\}$, partition the training set into cluster sets R_i using the Nearest Neighbor Condition, where $R_i = \{\mathbf{x} \in T \mid d(\mathbf{x}, \mathbf{y}_i) \leq d(\mathbf{x}, \mathbf{y}_j),\ \text{all } j \neq i\}$, and a suitable tie-breaking rule.

step 2b Using the Centroid Condition, compute the centroids for the cluster sets just found in Step 1 to obtain the new codebook $\mathbf{C}_{m+1} = \{\text{cent } (R_i) \mid i = 1, \ldots, N\}$. If an empty cell was generated in Step 2a, an alternate code vector assignment is made (in place of the centroid computation) for that cell.

Each iteration of the GLA monotonically decreases or keeps unchanged the average distortion of a vector quantizer. A Lloyd Algorithm flow chart is shown in Fig. 3.3.

In VQ design, two critical factors are very important: size of the training set and the number of GLA iterations. A small set of training vectors will not approximate the statistical characteristics of the vector sequence and will not give a good VQ. Overly

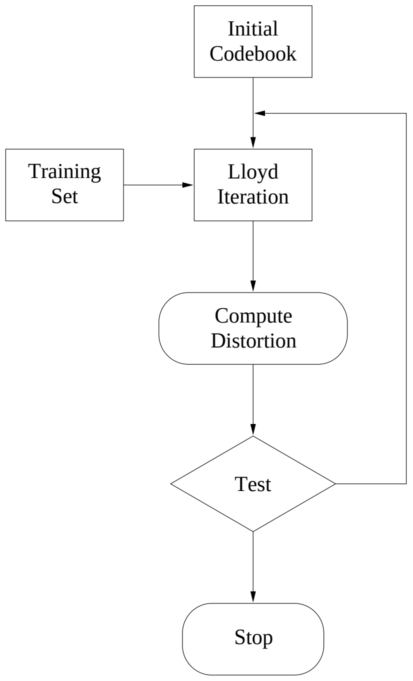


Fig. 3.3 Lloyd Algorithm Flow Chart.

trained codebook does not give an advantage since it will perform poorly when used with other input vector. A reasonable rule of thumb for effective VQ design is that the ratio of training set vectors M to the number of codebook entries N should be above 50 and less than 200 [7, 57].

Robustness

Normally, a VQ will perform properly if used on speech signals that were recorded under similar conditions as those of the training set. However, its performance will be reduced when used with other forms of speech (refer to Section 3.2.1). In most cases, a VQ design is evaluated by its robustness which refers to the resistance of a codebook to degraded performance when tested on data whose distribution is different from that of the training data.

3.1.4 Split Vector Quantization

In unconstrained or full-search VQ, a single codebook containing $N = 2^b$ codevectors is used to quantize a vector $\mathbf{x}$ of dimension k at a rate of r bits per vector component, or $b = kr$ bits per vector. However, the search complexity of an unconstrained VQ codebook increases exponentially with the vector dimension. In addition, the memory requirements for storing the VQ codebook becomes prohibitively large with the number of codevectors and with the dimension of the vector sequence. Split Vector Quantization (SVQ), a form of a *Product Code* technique [58], is used to slightly sacrifice distortion in return for substantial savings in codebook storage and search complexity.

In SVQ, a high dimension vector can be partitioned into two or more subvectors of lower dimensions which are then independently quantized using a codebook designed for that part of the vector. In fact, scalar quantization of a k-dimensional vector is equivalent to k-way split vector quantization in which the vector has been split in k one-dimensional subvectors.

SVQ is often used to quantize the speech spectral parameters in the encoder. The representation often used in codebook design is the LSF representation because of its many desirable properties. Recent developments [43] have showed that splitting the VQ in two or three vector gives reasonable performances for LSF based vector quantization. Splitting the LSF vector corresponds to splitting the LPC power spectrum. Usually one would assign more bits to the lower frequency spectrum than the higher one because of the sensitivity of the ear to lower frequencies. For example, in a 10'th order LSF representation for a 2-SVQ, half the bits are allocated to the first four LSF's and the other half to the last 6 (12.5% per bit for the lower part and 8.33% for the higher part) and for a 3-SVQ, 8 bits are allocated for the first and second three LSF and 8 bits for the last four for a 24 bits VQ (11.11% per bit for the first 6 LSF and 8.33% for the last 4). In Table 3.2, our simulation shows that a 2-SVQ produces a lower distortion measure than a 3-SVQ. However, the search complexity is decreased considerably compared to the SD performance loss of 0.08 dB.

In SVQ, instability of the all-pole speech reconstruction filters is avoided by ensuring that the LSF values in every entry of each codebook are in increasing order. However, the splitting procedure might lead to potential cross-over of the LSF values at the boundaries belonging to the optimally selected codebook subvectors. Figure

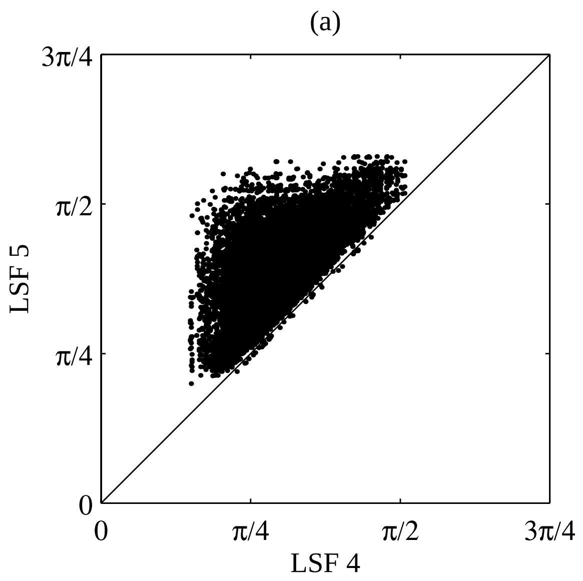
Table 3.2 Average spectral distortion of 2-SVQ and 3-SVQ for a 24 bit codebook when quantizing LSF vectors.

		Test Set		
	Bit	Average	SD Outliers (%)	
	Alloc	SD (dB)	2-4 dB	> 4 dB
2-SVQ	12,12	1.21	6.21	0.02
3-SVQ	8,8,8	1.29	7.00	0.05

3.4a shows a plot of the 4-th LSF values versus the 5-th LSF values from a coded set of LSF vectors. Since each subcodebook is trained separately, some points could cross to the other side of the unit slope straight line. Figure 3.4b shows the histogram of the 4-th LSF and the 5-th LSF of vector entries of a codebook. While most of the values of the 5-th LSF are larger than the 4-th, some of the entries in the lower subcodebook have values larger than in the higher subcodebook. Therefore, there is a small possibility that the resulting coded vector contains unordered LSF's. Many LSF cross-over correction methods have been proposed in previous work [59]. The simplest correction technique to avoid synthesis filter instability is just swapping the values of of the boundary LSF's if they are not ordered properly. However, in this thesis, we supply the subcodebook the last coded LSF value from the previous subcodebook and impose on the search the requirement that the first coded LSF be higher than the last one from the previous subcodebook.

3.2 Mapping Theory

In general, vector quantization works well with a testing set having the same recording conditions as the training set, but its performance deteriorates as different conditions are used. These conditions might include microphone response, acoustic background environments, filtering of the signal prior to digitization, etc. This section will show the deterioration in performance of VQ when subjected to different filtering conditions. Furthermore, we will introduce the mapping method and demonstrate how it can be used to improve the performance of VQ and hence, increase its robustness.



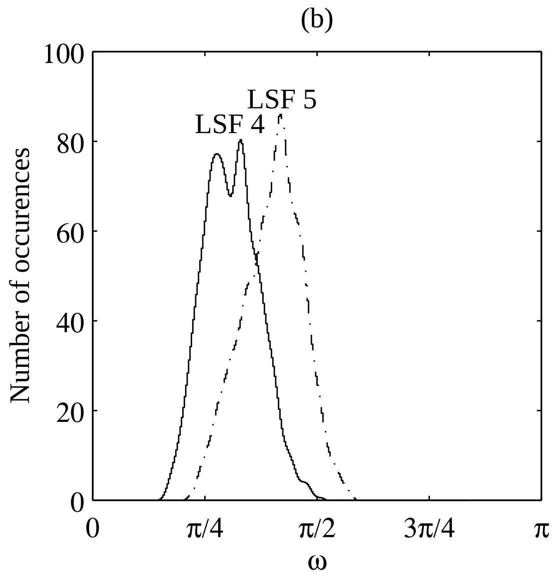


Fig. 3.4 (a) Scattered plot for LSF 4 vs LSF 5 for coded LSF vectors in 2-SVQ (b) A smooth histogram of LSF 4 and LSF 5 from a 2-SVQ codebook.

3.2.1 Performance Degradation of VQ

Different filtering conditions are applied on the speech testing set. After extracting the LSF vectors from the filtered speech, they are quantized using a codebook trained with clean LSF vectors. The same procedure is also done for codebooks trained with filtered vectors. The lowpass filter has a cutoff of 3.2 kHz, the highpass filter has a cutoff of 300 Hz while the bandpass filter is a combination of both of the latter.

When we quantize the differently filtered LSF's, we obtain the spectral distortion results shown in Table 3.3. The left column shows the worst case scenario for each codebook while the last column shows the results of similar condition quantization. The performance degrades enormously for some combinations (clean vs. bandpass and lowpass vs. highpass). In general, an SD measure of 3.00 dB is not an acceptable performance.

For the last column results, when the bandwidth of the spectral envelope is smaller, one would think that the spectral distortion should be lower. The reason is that the range of values of the LSF is more limited and hence should result in a slightly smaller SD. But in fact, filtering creates an additional peak in the power spectrum

Table 3.3 Spectral Distortion for a 2-SVQ subjected to different filtering. The first column shows the worst case scenario while the last column shows the ideal case.

Condition of Training Set	Condition of Testing Set Spectral Distortion Measure in (dB)			
	Bandpass	Lowpass	Highpass	Similar
Clean	2.95	2.56	1.62	1.21
	Highpass	Clean	Bandpass	Similar
Lowpass	3.00	2.57	1.74	1.38
	Clean	Lowpass	Highpass	Similar
Bandpass	3.87	3.09	2.57	1.32
	Lowpass	Clean	Bandpass	Similar
Highpass	3.87	3.04	2.65	1.16

(see Fig. 3.5) which makes the spectrum more difficult to approximate.

In the case of highpass filtered speech, a lower spectral distortion (1.16 dB) is obtained compared to the clean filtered speech (1.21 dB) due to the fact that the lower band of the frequency spectrum is not usually well approximated by the LP parameters in the first place.

The robustness to filtering of the VQ can be solved by building several codebooks to target each condition. This will work effectively as seen in the last column of Table 3.3 (1.27 dB of average SD over all 4 conditions). However, the storage of all the codebooks becomes a significant problem when many conditions need to be covered. This problem aggravates as the codebook size and the vector dimension increase. A more affordable scheme to solve this robustness issue is the codebook mapping technique which is discussed in the following section.

3.2.2 Block Diagram of the Mapping Model

After analyzing the behavior of LP parameters when different filtering conditions are applied, we observe that a slight alteration of the probability density functions (pdfs) of an LP parameter can cause the VQ to perform poorly. The modification to the

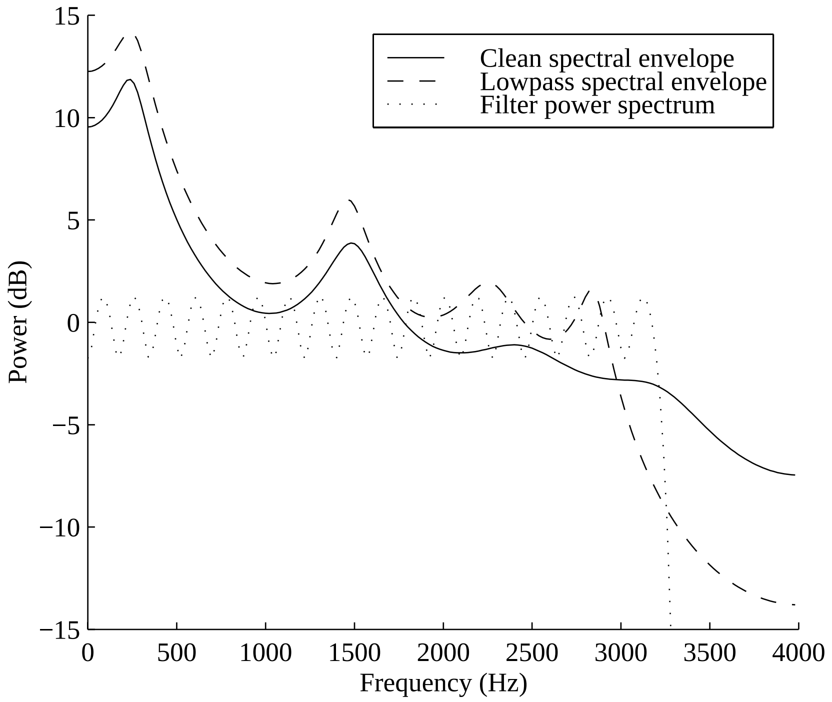


Fig. 3.5 LPC spectra of a 20 ms clean and filtered speech frame. A peak is observed around the cutoff frequency of the filter.

pdf depends on the type of processing done on the speech prior to quantization. By knowing what alteration on the pdf are caused by a specific condition, one could manipulate these pdfs to satisfy this condition. In this way, each filtering condition can be represented by a pdf transformation. Therefore, the robustness of VQ to any type of filtering is increased.

The mapping model consist of transforming the existing parameter pdfs in the codebook to meet any incoming pdf. Mapping functions are used on the original pdfs to approximate the filtered pdf. Therefore, we avoid having multiple codebooks by just storing a set of transformation functions. This idea is further illustrated in Fig. 3.6.

To determine which transformation to use on the input vector, the power spectrum is analyzed and depending on the type of speech, the mapping function will be tuned to it. The decoder knowing which mapping conditions to use will reconstruct the signal after adjusting its codebook.

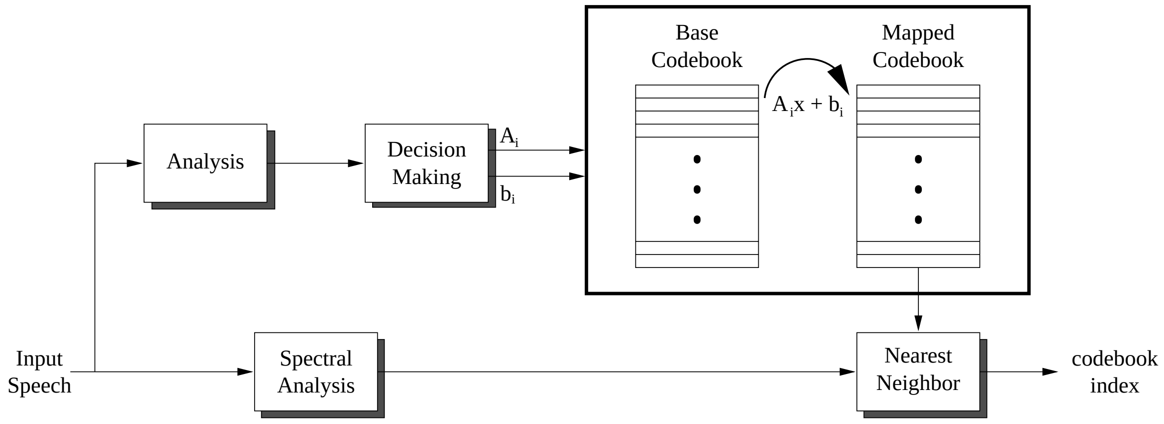


Fig. 3.6 Block Diagram of the mapping system.

Mapping the Input Vector

Rather than modifying each entry in the codebook using the mappings and then searching in the modified codebook, we can inverse map the input vector.

If the transformation matrix is applied to an LP parameter vector from the base codebook $\mathbf{x}_o$

$$\mathbf{x}_o = \left[\begin{array}{ccc} x_o^0 & \ldots & x_o^p \end{array} \right]^T \tag{3.8}$$

we obtain the corresponding converted codebook vector $\mathbf{x}_m$

$$\mathbf{x}_m = \mathbf{A}\mathbf{x}_o + \mathbf{b} \tag{3.9}$$

where $\mathbf{A}$ is a $p \times p$ matrix. The current vector to code $\mathbf{x}_c$ is quantized to $\hat{\mathbf{x}}_c$ using the weighted Euclidean distance (Section 2.4.2):

$$\hat{\mathbf{x}}_c = \arg\min_{\mathbf{x}_m} \left[\mathbf{W}(\mathbf{x}_c - \mathbf{x}_m)\right]^2 \tag{3.10}$$

where $\mathbf{W}$ is a matrix representing the weights and $\mathbf{x}_m$ as defined previously. Since we want the transformation to be done on the input vector $\mathbf{x}_c$, $\hat{\mathbf{x}}_c$ becomes:

$$\begin{aligned}\hat{\mathbf{x}}_c &= \arg\min_{\mathbf{x}_o} \left[\mathbf{W}(\mathbf{x}_c - \mathbf{A}\mathbf{x}_o - \mathbf{b})\right]^2 && (3.11a)\\ &= \arg\min_{\mathbf{x}_o} \left[\mathbf{W}\mathbf{A}(\mathbf{A}^{-1}(\mathbf{x}_c - \mathbf{b}) - \mathbf{x}_o)\right]^2. && (3.11b)\end{aligned}$$

When transforming the input vector to meet the specification of the codebook, we need to modify the weights.

The decoder will use the following demapping function to extract the quantized version of the original LP parameters:

$$\hat{\mathbf{x}}_c = \mathbf{A}^{-1}(\mathbf{x}_m - \mathbf{b}) \tag{3.12}$$

where $\mathbf{A}^{-1}$ denotes the inverse matrix of $\mathbf{A}$. The non-diagonal entries in $\mathbf{A}$ correspond to any correlation between x_i and x_j.

In the next section, a method to obtain the entries of the mapping functions is explained.

3.2.3 Transformation Matrix

Finding the best transformation matrices for different filtering conditions is critical to the VQ performance. A wrong set of mapping functions could revert all the improvements resulting from this model. Two types of mapping techniques are studied: *individual mapping* and *difference mapping*.

Individual Mapping: diagonal A

Individual mapping is defined as mapping each coefficient independently, with no relation to any other coefficient in the vector set. The matrix **A** is then diagonal

$$A = \begin{bmatrix} a_1 & 0 & \dots & 0 \\ 0 & a_2 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & a_p \end{bmatrix}, \qquad b = \begin{bmatrix} b_0 \\ \vdots \\ b_p \end{bmatrix} \tag{3.13}$$

The values of a_i's represent either the expanding or the compressing of the pdf. On the other hand, the values of b_i's represent the shifting of the pdf means.

Difference Mapping: triangular A

Another technique is to map the difference of two consecutive LP parameters. Each difference is mapped with a_i and b_i except the first value which is mapped relative to zero frequency:

$$x_1 = a_1\tilde{x}_1 + b_1 \tag{3.14a}$$

$$x_2 - x_1 = a_1(\tilde{x}_2 - \tilde{x}_1) + b_2 \tag{3.14b}$$

$$\vdots = \vdots \tag{3.14c}$$

$$x_p - x_{p-1} = a_p(\tilde{x}_p - \tilde{x}_{p-1}) + b_p \tag{3.14d}$$

where x_i is an LP parameter and $\tilde{x}_i$ is the filtered version. In matrix form, this results in:

$$A = \begin{bmatrix} a_1 & 0 & 0 & \dots & 0 \\ a_1 - a_2 & a_2 & 0 & \dots & 0 \\ a_1 - a_2 & a_2 - a_3 & a_3 & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ a_1 - a_2 & a_2 - a_3 & \dots & \dots & a_p \end{bmatrix}, \quad b = \begin{bmatrix} b_1 \\ b_1 + b_2 \\ b_1 + b_2 + b_3 \\ \vdots \\ b_1 + b_2 + \cdots + b_p \end{bmatrix} \tag{3.15}$$

The matrix is triangular and can be easily inverted to:

$$A^{-1} = \begin{bmatrix} \frac{1}{a_1} & 0 & 0 & \dots & 0 \\ \frac{a_2 - a_1}{a_2 a_1} & \frac{1}{a_2} & 0 & \dots & 0 \\ \frac{a_2 - a_1}{a_2 a_1} & \frac{a_3 - a_2}{a_3 a_2} & \frac{1}{a_3} & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \frac{a_2 - a_1}{a_2 a_1} & \frac{a_3 - a_2}{a_3 a_2} & \dots & \dots & \frac{1}{a_p} \end{bmatrix} \tag{3.16}$$

One disadvantage in using the difference method is that the quantization error introduced in the first few terms gets accumulated on the whole vector and in some cases can generate poor results.

Estimation of a_i's and b_i's

The parameters in the mapping functions are estimated on the bases of the pdfs of x_i and $\tilde{x}_i$, the clean and filtered LP parameters respectively (Fig. 3.7a). The transformation parameters a_i and b_i are predicted using the mean and standard deviation of x_i and $\tilde{x}_i$ as follows:

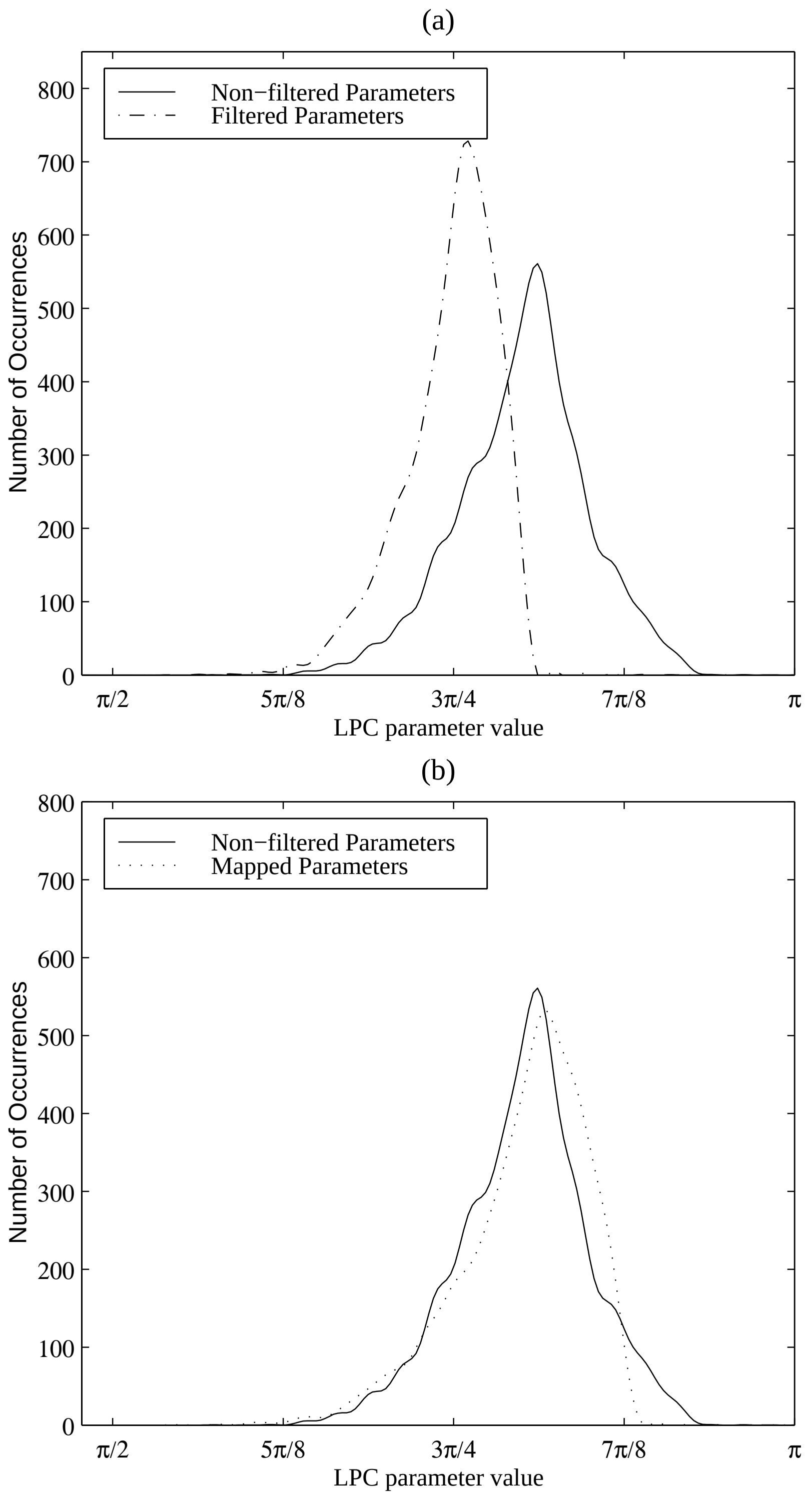


Fig. 3.7 (a) Probability density function of one non-filtered and filtered LP parameter, (b) Probability density function of the LP parameter after transformation.

$$a_i = \frac{\sigma_{x_i}}{\sigma_{\tilde{x}_i}} = \frac{\sqrt{\sum_{k=1}^{M} x_{ik}^2 - \left[\sum_{k=1}^{M} x_{ik}\right]^2}}{\sqrt{\sum_{k=1}^{M} \tilde{x}_{ik}^2 - \left[\sum_{k=1}^{M} \tilde{x}_{ik}\right]^2}}, \tag{3.17}$$

and

$$b_i = \mu_{x_i} - \mu_{\tilde{x}_i}\frac{\sigma_{x_i}}{\sigma_{\tilde{x}_i}} \tag{3.18a}$$

$$= \frac{1}{M}\sum_{k=1}^{M} x_{ik} - \frac{1}{M}\sum_{k=1}^{M} \tilde{x}_{ik}\frac{\sigma_{x_i}}{\sigma_{\tilde{x}_i}} \tag{3.18b}$$

where μ and σ denotes respectively the expected value and the standard deviation and M represent the number of values available to approximate the pdfs. For example, by applying the values obtained in the previous equations, the transformed pdf shown in Fig. 3.7b is obtained.

After finding the appropriate mapping matrices, the next step is to find a method to distinguish the different types of filtered speech signals.

3.2.4 Classification of the Filtered Speech

Using a Fast Fourier Transform, analysis on the input speech can be done easily to estimate the bandwidth of the input signal and correspondingly use the appropriate mapping function. In Fig. 3.8b, the values of a power band from 3.0 kHz to 3.2 kHz and from 3.2 kHz to 3.4 kHz for a clean speech signal are plotted with respect to time. Since the speech is not bandlimited, the values are very close. But in Fig. 3.8c, a lowpass filtered speech will generate much lower power in the 3.2 kHz to 3.4 kHz and can be distinguished from the clean signal.

If we need to determine exactly the bandwidth of the speech signal to be coded, the power spectrum can be divided into enough bands to have a good approximation of the cutoff frequencies of the filter used. In this case, we can modify our transformation

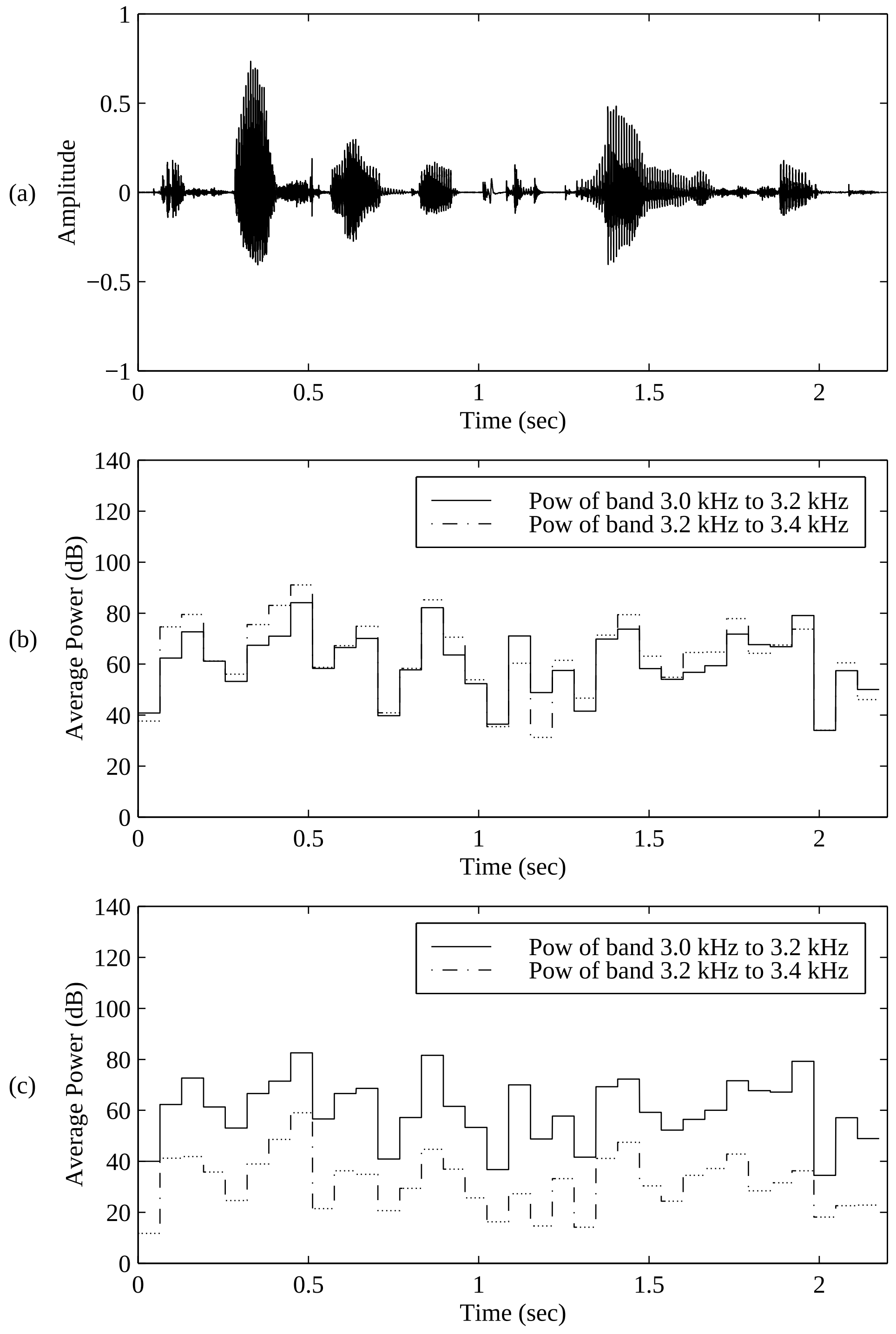


Fig. 3.8 (a) Time domain speech signal, (b) the power spectrum is partitioned into bands and two of the bands are plotted with respect to time for a clean speech signal, (c) same as in (b) but for a lowpass filtered speech signal.

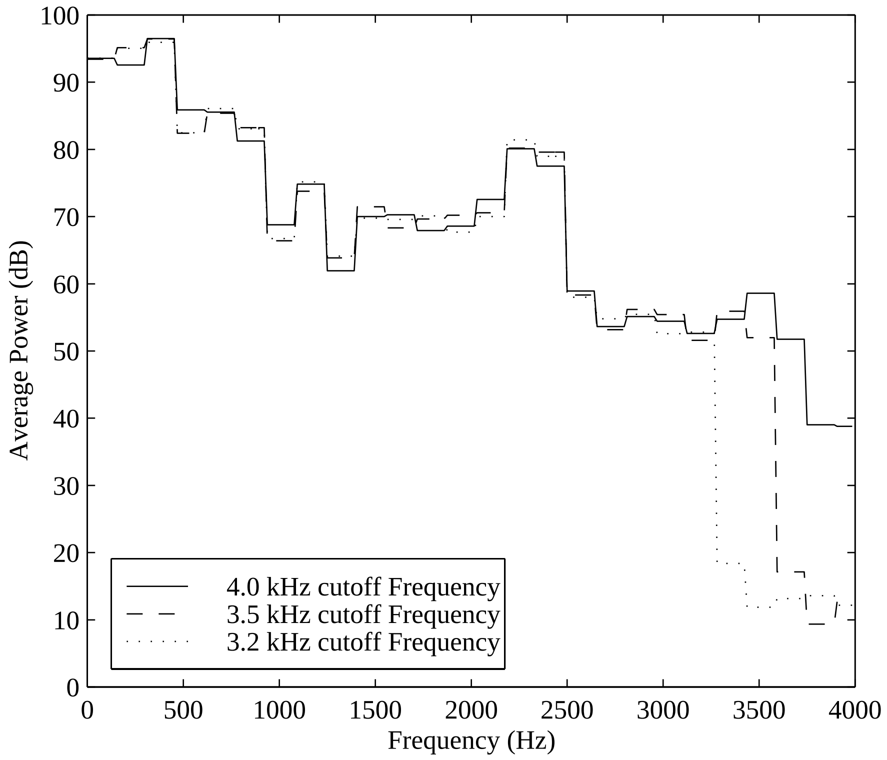


Fig. 3.9 Power spectrum of a speech signal lowpass filtered with 3.2 kHz, 3.5 kHz and 4.0 kHz cutoff frequencies. The power spectrum is average into bands of 150 Hz of bandwidth.

matrices to handle different cutoff frequencies for a specific type of filter by including a multiplier μ. Figure 3.9 shows how the power bands vary in speech signals filtered with different cutoff frequencies.

By estimating the cutoff frequency, we can interpolate between two matrices to obtain a mapping function for a new cutoff frequency. The new interpolated mapping functions are calculated as follows:

$$\mathbf{A}_{int} = \mathbf{A}_c + \mu(\mathbf{A}_f - \mathbf{A}_c) \tag{3.19a}$$

$$\mathbf{b}_{int} = \mu \mathbf{b}_f \tag{3.19b}$$

where $\mathbf{A}_{int}$ is the interpolated matrix and $\mathbf{A}_c$ is the matrix for the clean codebook, in this case the identity matrix, and $\mathbf{A}_f$ is the mapping matrix for the filtered parameters. The value of μ is computed as follows:

$$\mu = \frac{f_d - f_f}{f_c - f_f} \tag{3.20}$$

where f_d is the new cutoff frequency, f_c is the clean cutoff frequency, in this case 4000 Hz, and f_f is the cutoff frequency of the other filter, in the case of a lowpass it is 3200 Hz. By using this interpolation, we can estimate the mapping functions for filters with any cutoff frequency.

3.2.5 Stability Check

The transformation functions can be applied to any representation of the LP parameters (i.e. line spectral frequencies, predictor coefficients, reflection coefficients, etc.). In all cases, a stability check have to be applied after mapping to prevent any unstable synthesis filters at the decoder. For the LSF representation, the property which guarantees the stability of the p-th order LP synthesis filter is that the values in the vector $\mathbf{x}$ must be in ascending order (Equation (2.28)).

In some cases, the mapping function might cause certain reconstructed LSF's to cross over. Swapping those elements is a simple solution to reorder the parameters. On the other hand, some constraints can be imposed on the entries of the transformation matrices to prevent any cross-over on the expense of a small degradation in performance. Two different constraints arise for *individual mapping* and *difference mapping.*

Individual Mapping

In the case of individual mapping, if we consider two consecutive LSF's ω_i and ω_{i+1} as follows:

$$\tilde{\omega}_i = \frac{\omega_i - b_i}{a_i} \tag{3.21a}$$

$$\tilde{\omega}_{i+1} = \frac{\omega_{i+1} - b_{i+1}}{a_{i+1}} \tag{3.21b}$$

Since we know that $\omega_i < \omega_{i+1}$ we get:

$$\tilde{\omega}_i < \frac{a_{i+1}}{a_i}\,\tilde{\omega}_{i+1} - \frac{b_{i+1} - b_i}{a_i} \tag{3.22}$$

So to insure that $\tilde{\omega}_i < \tilde{\omega}_{i+1}$ we must have the strict constraints:

$$b_{i+1} > b_i \tag{3.23a}$$

$$a_{i+1} > a_i \tag{3.23b}$$

These constraints will guarantee that the LSF's remain ordered after transformation.

Difference Mapping

In the case of difference mapping, a simple constraint is obtained. If we consider the following:

$$\omega_i - \omega_{i-1} = a_i\,(\tilde{\omega}_i - \tilde{\omega}_{i-1}) + b_i \tag{3.24a}$$

$$\omega_{i+1} - \omega_i = a_{i+1}\,(\tilde{\omega}_{i+1} - \tilde{\omega}_i) + b_{i+1} \tag{3.24b}$$

Since we know that $\omega_i < \omega_{i+1}$ we get:

$$\omega_i < a_{i+1}\,(\tilde{\omega}_{i+1} - \tilde{\omega}_i - b_{i+1}) + b_{i+1} + w_i \tag{3.25a}$$

$$\tilde{\omega}_i < \tilde{\omega}_{i+1} + \frac{b_{i+1}}{a_{i+1}} \tag{3.25b}$$

So to insure that $\omega_i < \omega_{i+1}$ we must have this constraint:

$$b_i > 0 \tag{3.26}$$

In practice, those constraints were not used because the percentage of unordered LSF's was minimal and can be easily corrected by just swapping the mis-ordered LSF's.

Chapter 4

Performance Analysis

In this chapter, we evaluate the mapping model under different conditions. First, we study which spectral representation is the most suitable for the model. Then, an investigation is done on different ways of computing the transformation parameters as well as choosing which type codebook is used as the *base* codebook. Furthermore, performance results are presented and compared to two different codebook designs. Finally, this method is tested under different bit rates and on speech filtered with different cutoff frequencies.

4.1 Effect of Filtering on Various LP Representations

Even though quantization of the LP parameters is often done in the LSF domain due to its many attractive properties, the mapping can be done on any representation of the spectral envelope. In this section, the effects of filtering the speech is demonstrated on three representations, namely, the predictor coefficients, the reflection coefficients and the line spectral coefficients. Note that we use these different representations for mapping purposes, but quantization itself is carried out on the LSF's derived from the mapped parameters.

4.1.1 Predictor Coefficients

The predictor coefficient (PC) domain is a natural representation of linear prediction parameters. They are simply the coefficients of the all-pole filter model. In practice, their values can range from -8 to 8 for speech data and have no simple spectral interpretation.

A 10'th order LP analysis is done on clean and filtered speech data and the resulting probability density functions of the predictor coefficients are shown in Fig. 4.1. The impact of filtering on the PC's is very noticeable. The pdfs of the PC's change considerably when the speech data is lowpass filtered, but the highpass data generates predictor coefficients very similar to the non-filtered data. Further, the means of the filtered pdfs of the even PC's (2nd, 4th, ...) shift to larger values no matter what type of filtering is applied. On the other hand, the odd PC's (1st, 3rd, ...) alternate, i.e., the lowpass PC values get larger and the highpass values smaller. Finally, all PC's are affected by the filtering process.

For mapping purposes, there are several disadvantages to using the PC domain:

- Quantizing the highpass LSF's with a clean codebook gives a 1.62 dB spectral distortion (refer to Table 3.3), a 0.46 dB difference with respect to a codebook trained only with highpass data. However, the small difference in the PC pdfs for highpass and clean is too little with respect to that distortion difference. In other words, if a transformation is applied on the highpass PC's, the mapped codebook will give similar results as the clean codebook.
- By trying to map the PC coefficients, we obtained a rather high percentage of unstable filters resulting from the transformation. There is no easy way to fix that problem.
- A small error in the mapping of one PC can have a significant change on the whole power spectrum

4.1.2 Reflection Coefficients

The reflection coefficient (RC) domain is another representation of LP parameters. The range of values is limited, and falls within -1 and 1 if they result in a minimum phase filter.

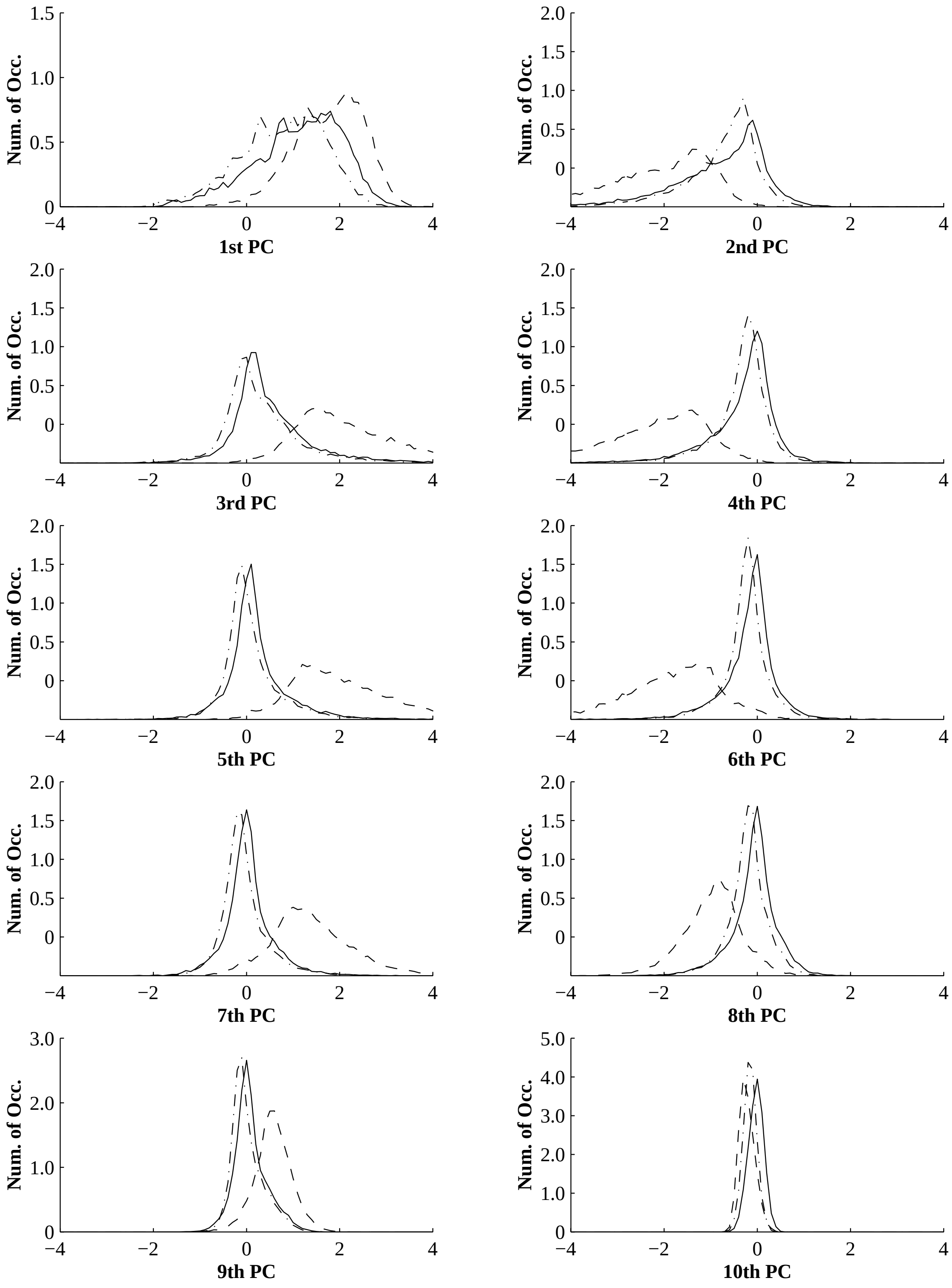


Fig. 4.1 Probability density functions for all 10 predictor coefficients. Straight lines represent clean PC, dashed lines represent lowpass PC and dasheddot represent highpass PC.

Figure 4.2 shows the pdfs of the 10 reflection coefficients resulting from lowpass and highpass filtering the speech signals. Similar observations to the PC representation can be made:

- The pdfs of the even filtered RC's shift to higher values regardless of the type of filtering.
- The pdfs of the odd filtered RC's shift to the left when lowpass filtered and to the right when highpass filtered.

In most cases, the shapes of the pdfs remain unchanged, and only a mean shift to lower or higher values is necessary for the mapping function. Another benefit of this domain over the previous one is that the additional stability check is unnecessary.

On the other hand, in the majority of vectors, the first two coefficients have values very close to -1 and 1, boundaries that cause an unstable filter. Furthermore, since there was no clear pattern to the effect of filtering, we did not use this domain for mapping.

4.1.3 Line Spectral Frequencies

Line Spectral Frequency (LSF) is one of the recently used spectral parameter representation. The LSF representation has a number of properties, including a bounded range, a sequential ordering of the parameters and a simple check for the filter stability. In addition, since it is a frequency-domain representation, LSF's approximate the locations of the formant frequencies, and exhibit distinct localized distributions.

Figure 4.3 shows the variations of the LSF coefficients with respect to time. In the case of lowpass speech, the first five LSF vectors are not affected by this processing, however the last 5 are shifted downward due to the absence of high frequency power (see Fig. 4.3a). For highpass speech, the first 3 values are shifted upward and the rest are not altered (see Fig. 4.3b).

Figure 4.4 displays the pdfs of all LSF coefficients for clean and bandpass filtered speech. The results shows that only a few pdfs are altered. The altered pdfs are shown separately in Fig. 4.5 and 4.6. The variances and the means of the pdfs of LSF 1, LSF 2, LSF 9 and LSF 10 are clearly changed and must be transformed to decrease the quantization error. It should also be noted that the lower LSF's react in

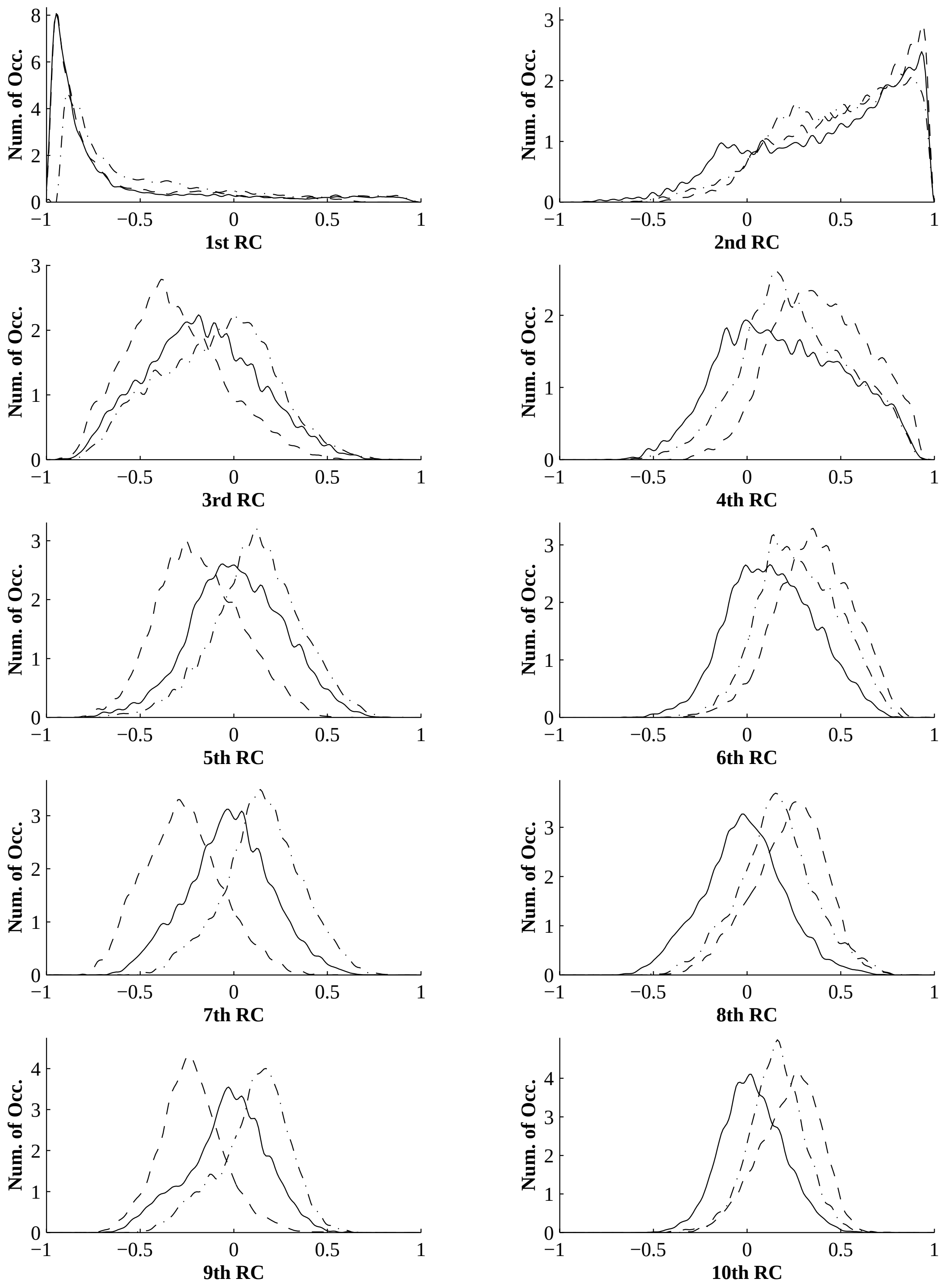
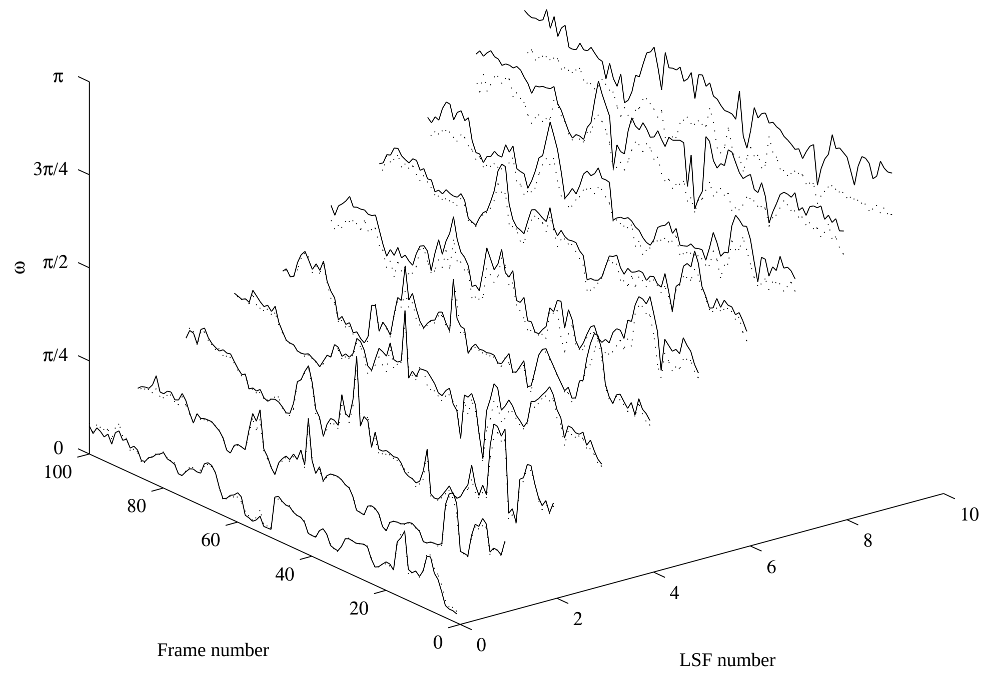


Fig. 4.2 Probability density functions for all 10 predictor coefficients. Straight lines represent clean RC, dashed lines represent lowpass RC and dasheddot represent highpass RC.

(a)



(b)

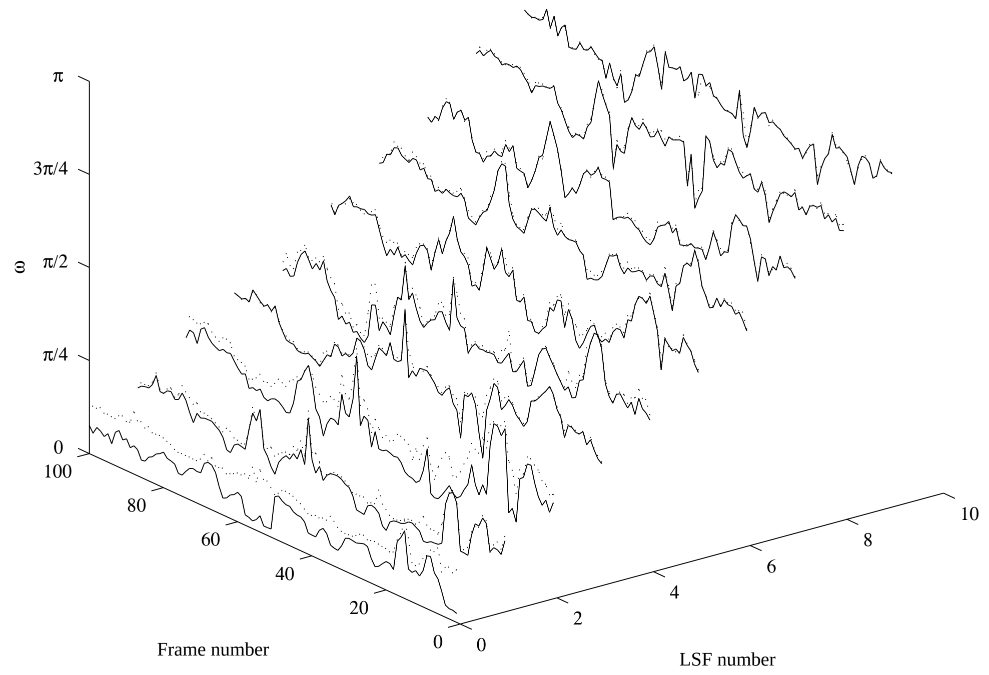


Fig. 4.3 LSF vector values changing with time (frame) (a) effect of lowpass filtering the speech, (b) effect of highpass filtering the speech signal.

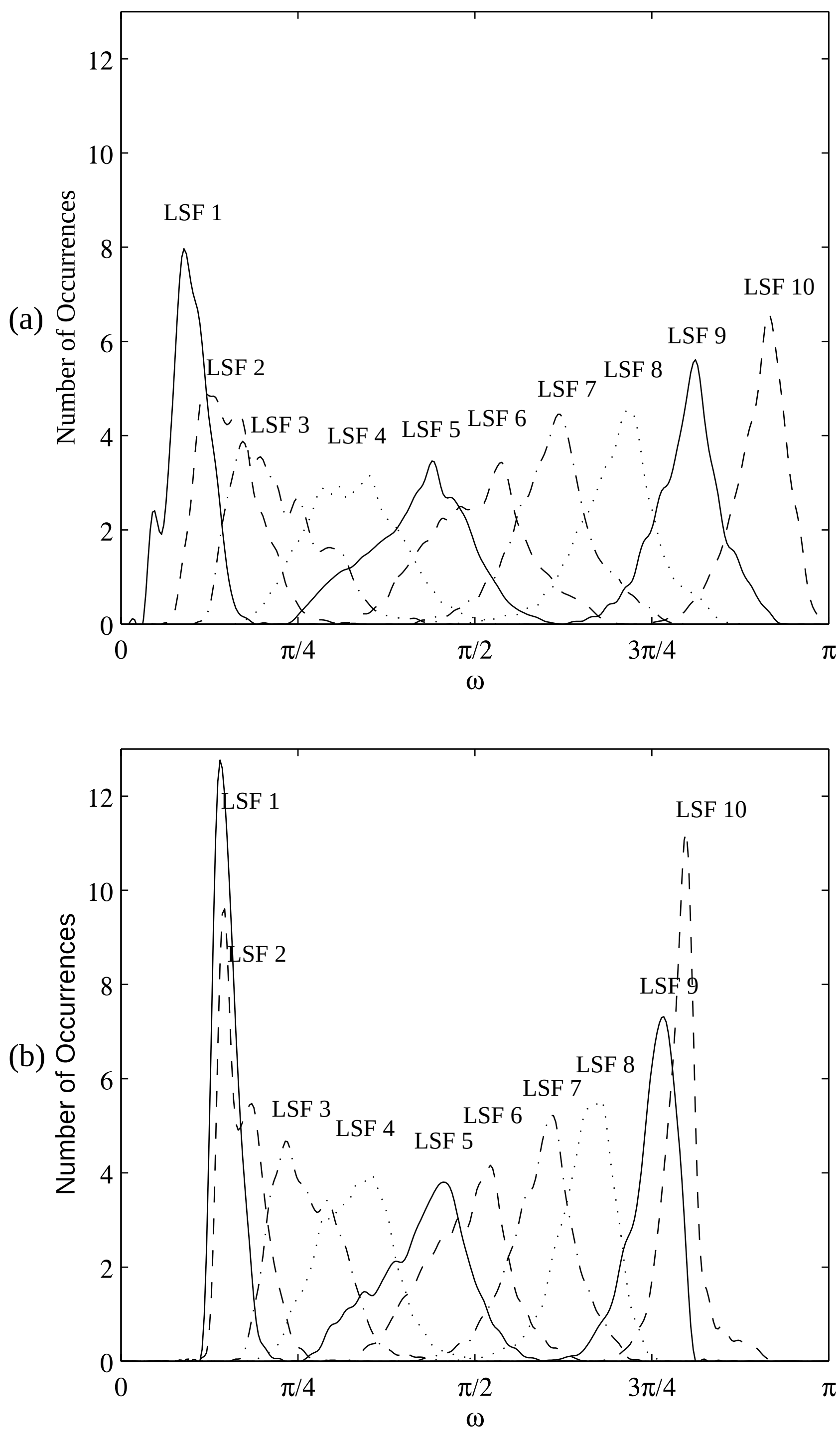


Fig. 4.4 (a) Pdfs of all 10 LSF coefficients, (b) pdfs of all 10 LSF coefficients when the speech signal is bandpass filtered.

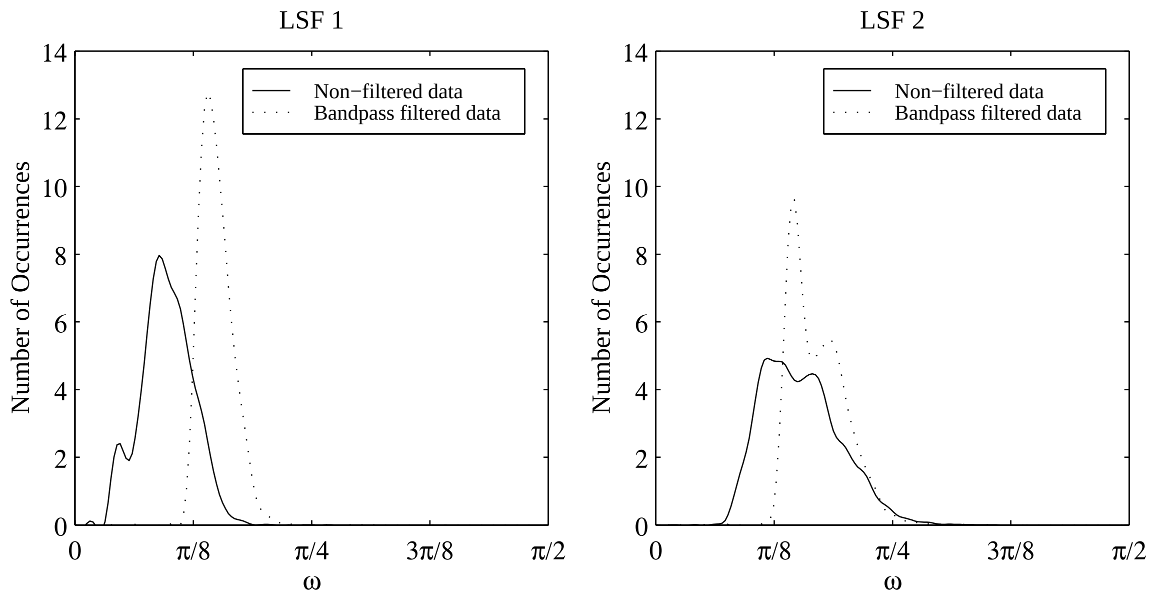


Fig. 4.5 LSF 1 and LSF 2 of a non-filtered data speech and a bandpass filtered data speech. The pdfs are approximately the same if the speech is highpass filtered.

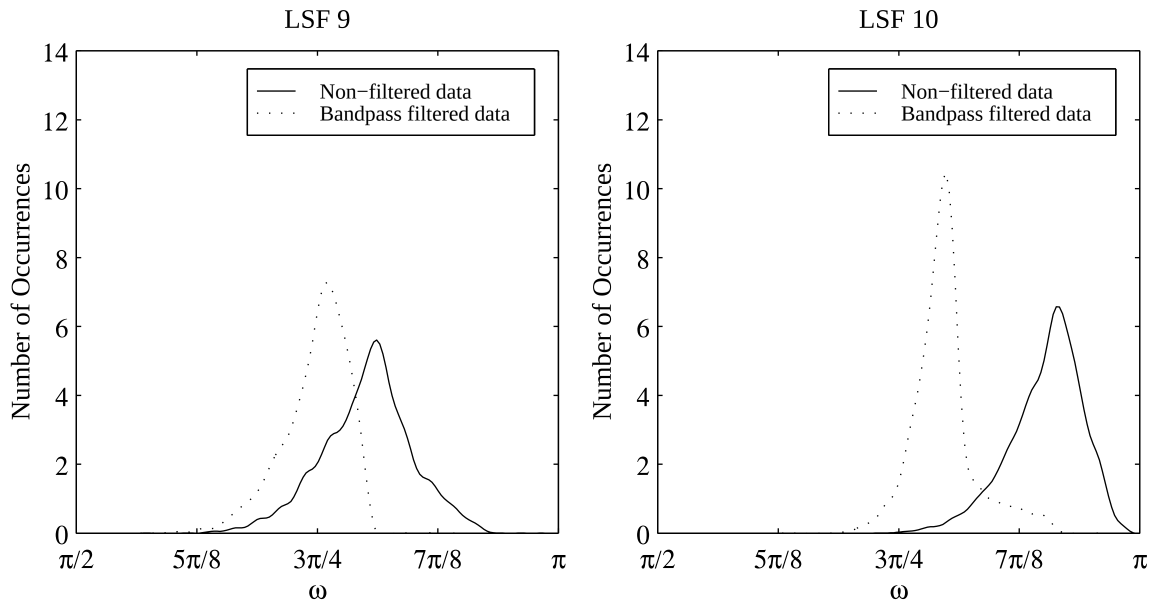


Fig. 4.6 LSF 9 and LSF 10 of a non-filtered data speech and a lowpass filtered data speech. The pdfs are approximately the same if the speech is lowpass filtered.

the same manner if the speech is lowpass or bandpass filtered. A similar observation for the highpass and bandpass filtered speech is noticed for the higher LSF's.

Since a subset of the coefficients is affected by filtering the speech signals, this domain is the most appropriate to use for mapping purposes since transformation is not needed for the LSF's that remain the same. Another reason for using this representation is the ease of stability checking. Furthermore, due to their low spectral sensitivity, a small variation in the LSF parameters caused by the transformation generally only affects local frequency regions in the power spectral density. Finally, since quantizing the difference between consecutive LSF's in a frame is widely used, one can transform the difference. In this case the stability check is simple — the resulting mapped difference should be positive.

4.2 Codebook Transformations

This section explores how the codebook transformation can be optimized to achieve lower spectral distortion. Furthermore, since we are ignoring the constraints set in Section 3.2.5, we need a transformation that produces a low percentage of unstable filters. In addition, since correction algorithms introduce additional error beyond that due to quantization, a low percentage of unstability is preferred. LSF vector sets with different sizes are used to compute the entries of the mapping matrices and then an SD evaluation is performed on those entries. In addition, different *base* codebooks for the model are evaluated.

4.2.1 Computation of the Mapping Parameters

Three sets of vectors are used to compute the entries of the **A** and **b** matrices: a large set (145 388 vectors) composed of the training vectors, the small set (16 189 vectors) composed of the testing set and the codebook (4 096 vectors). An LP analysis is done on the four types of filtered speech to obtain four sets of training vectors and testing vectors. Then, Equation (3.17) and Equation (3.18) are applied on those sets to generate the entries of the transformation matrices. For the case of the codebook, we take the four codebooks trained with filtered LSF's (4 096 vectors) and use them to get the transformation coefficients. Then, we compute the spectral distortion for the newly computed **A** and **b** matrices on the three kinds of filtered speech (testing set). In the simulations, the *base* codebook was generated for unfiltered speech and was mapped accordingly to satisfy the three filtered testing sets, namely LP, BP and HP.

Table 4.1 summarizes the spectral distortion performance results. The first column specifies which vector set is used to compute the matrices and the last column tabulates the percentage of unstable filters. Since the training set has the most vectors, it is evident that it produces the lowest SD measure and the lowest percentage of non-minimum phase filters at the synthesis stage.

Another observation from Table 4.1 is that the SD from the training and the codebook set are very close. Since using approximately 4 000 elements to estimate the pdf is sufficient, we can develop a method which adaptively compute the mapping functions if we identify the type of filtering used dynamically. The adaptive method consists of updating periodically the entries in the matrices depending on a certain number of previous vectors. On the other hand, updating the entries too often does not pay off in terms of the complexity and transmission overhead.

Table 4.1 Results showing the SD performance for the mapping entries calculated using either the training, testing or codebook vectors for a 2-SVQ (with a clean *base* codebook). The percentage of unstable filters resulting from the mapping is also shown.

Map	Type	Average	SD Outliers (%)		Unstable
Set	Filt	SD (dB)	2-4 dB	> 4 dB	filter (%)
	LP	1.62	22.55	0.16	0.05
Train	BP	1.65	24.02	0.51	5.27
(145 388)	HP	1.32	11.46	0.14	1.27
	LP	1.64	22.81	0.23	0.22
Test	BP	1.67	24.99	0.57	7.79
(16 189)	HP	1.33	11.28	0.12	1.64
	LP	1.67	23.89	0.16	0.03
CB	BP	1.70	25.44	0.51	6.50
(4 096)	HP	1.36	11.50	0.14	2.12

4.2.2 Selection of the Base Codebook

Analysis is done on different *base* codebooks so as to search for the best performance codebook. Four types of codebooks are first trained using vectors from the four types of filtered speech. Each type of codebook is then used as the *base* codebook to be transformed. The SD measure of each type of *base* codebook is computed on the four different filtering conditions in Table 4.2. The diagonal numbers in this table requires no transformation since the *base* codebook is filtered in the same way as the testing set. The bandpass *base* codebook gives an approximately equal performance for all four filtered testing sets (1.41 dB SD) while the worst mismatch occurs when a highpass *base* codebook is used to quantize the lowpass filtered testing set (1.72 dB SD).

Table 4.2 Spectral distortion measure for different base codebooks. Each codebook is transformed to meet the condition of the testing set.

Base CB	Testing Set Condition Spectral Distortion Measure in (dB)			
Condition	Clean	Lowpass	Bandpass	Highpass
Clean	1.22	1.62	1.65	1.32
Lowpass	1.38	1.39	1.47	1.39
Bandpass	1.51	1.58	1.32	1.29
Highpass	1.38	1.73	1.53	1.16

Table 4.3 shows the average SD measure, the average percentage of outliers and the average percentage of unstable filters for all four conditions on each *base* codebook. Although, the lowpass *base* codebook yields the lowest SD measure and the lowest percentage of outliers, it gives a higher percentage of unstable filters. We prefer the clean *base* codebook since it can be transformed into the other conditions and has the lowest percentage of unstable filters after transformation.

Hereafter, all the transformations is done on the clean codebook with the matrices entries calculated using the training set.

Table 4.3 Overall average of SD performance and its outliers of the four base codebooks done on the testing and training sets. The percentage of unstable filters resulting from the mapping procedure is also computed.

Base CB	Average	SD Outliers (%)		Unstable
Condition	SD (dB)	2-4 dB	> 4 dB	Filter (%)
Clean	1.45	21.70	0.21	1.65
Lowpass	1.41	18.57	0.10	2.10
Bandpass	1.43	19.32	0.04	2.56
Highpass	1.45	15.07	0.16	2.83

4.3 Performance of the Mapping Model

In this section, the computed entries of the matrices for the individual and the difference mapping are presented. In addition, the SD performance of these two mapping methods are compared to a regular codebook and its modified version. This modified version, which we will refer to as a *mixed* codebook hereafter, is trained on a set of vectors that includes all four filtering speech data. In practice, such a codebook is difficult to train due to the large number of vectors in the training set which can easily increase the computational efforts of training by four times. This codebook is generated just for comparing the various results of the mapping model.

4.3.1 Individual Mapping

For individual mapping, the mapping is done on each coefficient independently. The values for a_i's and b_i's computed using the training set and Equation (3.17) and (3.18) for the three types of filtered speech are tabulated in Table 4.4. In the case of the lowpass filtered mapping function parameters, the first few values of a_i's are close to one while the values of b_i's are close to zero. This is simply because the first few LSF parameters are not affected by the lowpass filtering as mentioned in Section 4.1.3). Similarly the highpass mapping function has the last few values of a_i close to one and b_i close to zero. It should be noted that filtering has a small effect on entries that should not be altered. A value of 0.9420 shows that LSF 1 in clean and lowpass filtered are not exactly the same.

Table 4.4 The entries of the individual mapping functions of the transformation model for lowpass, bandpass and highpass filtered speech used in the simulations.

LSF	Lowpass mapping		Highpass mapping		Bandpass mapping	
Number	a_i	b_i	a_i	b_i	a_i	b_i
LSF 1	0.9420	0.0198	0.6220	0.2557	0.6092	0.2555
LSF 2	0.9088	0.0546	0.5376	0.2578	0.5190	0.2537
LSF 3	0.9262	0.0305	0.8950	0.2192	0.8524	0.2221
LSF 4	0.9839	−0.0359	0.8635	0.1922	0.8141	0.1877
LSF 5	0.8990	0.0933	0.9441	0.1286	0.8786	0.1723
LSF 6	0.8483	0.1518	0.9392	0.1212	0.8135	0.2264
LSF 7	0.9101	0.0871	0.9973	0.0409	0.8514	0.2264
LSF 8	0.8121	0.2571	0.9907	0.0327	0.8127	0.2601
LSF 9	0.7216	0.5146	1.0198	−0.0302	0.7162	0.5362
LSF 10	0.6210	0.7196	1.0257	−0.0673	0.5780	0.8235

4.3.2 Difference Mapping

For difference mapping, the differences of consecutive LSF values in a frame are mapped rather than each LSF individually. Table 4.5 shows the computed entries of the two matrices for all three filtering conditions. Similar observation to the individual mapping can be made for the difference mapping. Since the pdfs of the difference is being mapped, the mean values are smaller. Hence the values of b_i are close to zero. The only exception being the case of b_1 (especially for highpass and bandpass) where LSF 1 is mapped, not the difference.

4.3.3 Comparison of Different Codebook Designs

Four different codebook designs, namely, *clean*, *mixed*, *individual mapped* and *difference mapped* codebooks are tested. The spectral distortion and the percentage of outliers are used as performance evaluation for an 2-SVQ and an 3-SVQ. The bit allocation for the 2-SVQ is 12 bits for the lower subcodebook and 12 bits for the upper subcodebook while for the 3-SVQ, 8 bits are allocated for each subcodebook.

The performance results for a lowpass filtered testing speech is shown in Table 4.6. Table 4.7 and 4.8 summarize the simulation results performed on a bandpass and

Table 4.5 The entries of the difference mapping functions of the transformation model for lowpass, bandpass and highpass filtered speech used in the simulations.

LSF	Lowpass mapping		Highpass mapping		Bandpass mapping	
Number	a_i	b_i	a_i	b_i	a_i	b_i
LSF 1	0.9420	0.0198	0.6220	0.2557	0.6092	0.2555
LSF 2-1	0.9040	0.0094	0.4223	−0.0126	0.4522	−0.0108
LSF 3-2	0.9985	0.0054	1.1392	0.0405	1.1693	0.0459
LSF 4-3	0.9556	−0.0153	0.7503	−0.0358	0.8239	−0.0319
LSF 5-4	0.9793	0.0087	0.9654	0.0209	0.9937	−0.0009
LSF 6-5	0.7903	0.0173	0.6624	0.0121	0.8149	0.0208
LSF 7-6	1.0696	−0.0035	1.0298	0.0020	1.0375	−0.0046
LSF 8-7	0.8381	−0.0237	0.7289	−0.0162	0.9112	0.0016
LSF 9-8	0.8982	−0.0119	0.9051	0.0138	1.0479	−0.0097
LSF 10-9	0.5100	−0.0198	0.4132	0.0010	0.9790	−0.0075

highpass testing speech respectively. Several observations can be noted as follows:

- Generally, 3-SVQ generates higher SD values than 2-SVQ due to the splitting of the codebook into more subvectors. As the VQ is split further, the distortion becomes similar to that of scalar quantization. Since breaking the VQ into lower dimensions decreases the number of codewords in each subcodebook, a 3-SVQ mixed codebook does not contain enough vectors to represent its wide variety of training vectors. Thus, a 3-SVQ results in higher SD spectral distortion as confirmed by our results (1.88 dB SD for 3-SVQ and 1.83 dB SD for 2-SVQ)

- The mapping functions become more and more useful as codebooks have fewer codevectors. Our results shows that a small mixed codebook can not effectively quantize a wider variety of LSF vectors. The mixed codebook has 1.88 dB SD compared to 1.58 dB SD for the individual mapping. Furthermore, since the mapping codebooks can match any of the three pdfs, splitting does not worsen the results. This is the reason why we get approximately similar performance for the 2-SVQ and 3-SVQ even though the latter has significantly fewer codevectors (Diff Map 2-SVQ: 1.68 dB SD and Diff Map 3-SVQ: 1.66 dB SD).

Table 4.6 Spectral distortion measure for 2-SVQ and 3-SVQ when the input testing speech is **lowpass** filtered. Results for *clean*, *mixed*, *individual mapped* and *different mapped* codebooks are shown.

2-SVQ Results

Type of Codebook	Bit Alloc	Test Set		
		Average SD (dB)	SD Outliers (%)	
			2-4 dB	> 4 dB
Clean	12,12	2.56	69.46	3.89
Mixed	12,12	1.83	34.96	0.39
Indep. Mapped	12,12	1.62	22.55	0.16
Diff. Mapped	12,12	1.68	25.70	0.26

3-SVQ Results

Type of Codebook	Bit Alloc	Test Set		
		Average SD (dB)	SD Outliers (%)	
			2-4 dB	> 4 dB
Clean	8,8,8	2.60	70.02	2.99
Mixed	8,8,8	1.88	36.56	0.39
Indep. Mapped	8,8,8	1.58	18.51	0.19
Diff. Mapped	8,8,8	1.66	23.82	0.19

Table 4.7 Spectral distortion measure for 2-SVQ and 3-SVQ when the input testing speech is **bandpass** filtered.

2-SVQ Results

Type of Codebook	Bit Alloc	Test Set		
		Average SD (dB)	SD Outliers (%)	
			2-4 dB	> 4 dB
Clean	12,12	2.95	77.60	8.92
Mixed	12,12	1.94	41.70	0.43
Indep. Mapped	12,12	1.65	24.02	0.51
Diff. Mapped	12,12	1.67	24.46	0.55

3-SVQ Results

Type of Codebook	Bit Alloc	Test Set		
		Average SD (dB)	SD Outliers (%)	
			2-4 dB	> 4 dB
Clean	8,8,8	2.99	79.23	9.48
Mixed	8,8,8	2.00	46.18	0.44
Indep. Mapped	8,8,8	1.65	22.19	0.55
Diff. Mapped	8,8,8	1.71	25.21	0.57

Table 4.8 Spectral distortion measure for 2-SVQ and 3-SVQ when the input testing speech is **highpass** filtered.

2-SVQ Results

Type of Codebook	Bit Alloc	Test Set		
		Average SD (dB)	SD Outliers (%)	
			2-4 dB	> 4 dB
Clean	12,12	1.62	21.97	0.10
Mixed	12,12	1.33	8.65	0.01
Indep. Mapped	12,12	1.32	11.46	0.14
Diff. Mapped	12,12	1.30	9.74	0.19

3-SVQ Results

Type of Codebook	Bit Alloc	Test Set		
		Average SD (dB)	SD Outliers (%)	
			2-4 dB	> 4 dB
Clean	8,8,8	1.78	29.27	0.36
Mixed	8,8,8	1.43	10.35	0.07
Indep. Mapped	8,8,8	1.42	13.99	0.35
Diff. Mapped	8,8,8	1.45	14.06	0.41

- The quality of a VQ is greatly dependent on the percentage of outliers between 2 and 4 dB encountered. A value of 70% is unsuitable for quantizing speech spectral parameters. Using a mixed codebook decreases this percentage only to a certain extent but not enough for the purpose of speech coding. On the other hand, the individual and difference mapping give a lower percentage of 22%.

- The percentage of outliers bigger than 4 dB is sometimes better in the case of the mixed codebook. The transformation can destroy the naturalness of the LSF order in the VQ and cause it to choose a bad codeword from the subcodebooks which, in turn, results in a very high spectral distortion. For example, for a bandpass testing LSF vectors, a 2-SVQ with mixed codebook has 0.43 % of outliers while the mapped has 0.51%.

- Individual mapping always results in better performance than the difference mapping. Mapping each element independently gives a higher degree of freedom and thus the pdfs can be matched in a much more consistent matter. Furthermore, in some cases, the pdfs of the difference between LSF's can not be mapped accurately due to their change in skewness, "tilted" to the right or to the left. Figure 4.7 shows a case where the mapping could not exactly match the pdfs of the filtered data. In the case where the tilt of the pdfs is modified after the filtering of speech, the mapping is not as effective as when the tilt is unchanged. On the other hand, difference mapping has an exceptional feature in which the chances of having unstable filter after mapping is nil.

The mapped codebook has an improvement of approximately 0.95 dB in spectral distortion compared to a clean codebook when used on a filtered signal. Such an improvement is greater than usual because the training vectors for the clean codebook have considerably different filtering conditions than the testing vectors. If one expects different filtering conditions, a mixed codebook is more appropriate than a codebook trained only on clean speech. But mapping still offers an average of 0.2 dB improvement over a mixed codebook.

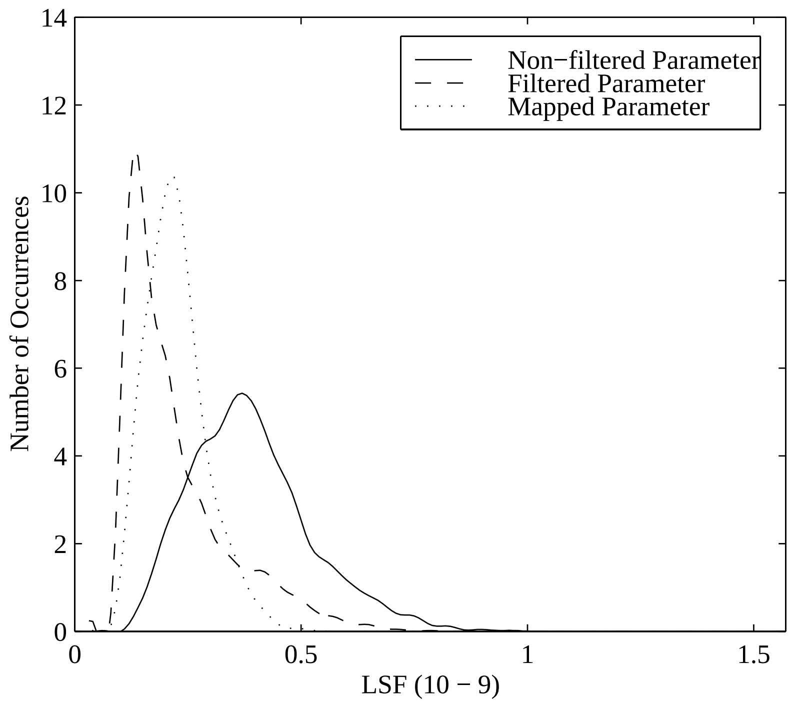


Fig. 4.7 Probability density function of the difference between LSF 10 and LSF 9 when it is clean, lowpass filtered and mapped.

4.4 Performance of the Mapping over Various Bit Rates

We tested the mapping model under different bit rates. For each bit rate, the subcodebook sizes are chosen in such a way that an equal number of bits is allocated for each subcodebook. For example, for a 2-SVQ, if 28 bits are assigned to a frame, then 14 bits goes to each codebook.

Figure 4.8 plots the average spectral distortion for all four different codebook designs for a 2-SVQ. Evidently, the more bits are allocated for the codebooks, the lower the SD distortion is, for all designs. The mapping model, including the individual and difference techniques, performs well over the whole range of bit rates with a slight preference for the individual case. In addition, the performance of the mixed codebook and the mapped codebook converge with increasing bit rate. Since a higher bit rate is associated with a larger number of codeword entries, the mixed codebook would be able to handle a wider variety of input signals, if it is trained appropriately to match all possible filtering conditions. Furthermore, at an average

SD level of 2 dB, the mixed codebook has an advantage of 6 bits/frame over the clean codebook when a lowpass input is applied. The mapped codebook has an additional 3 bits/frame over the mixed codebook.

The same experiment is done for highpass and bandpass testing speech. Similar observations can be made from Fig. 4.9 and Fig. 4.10. In the highpass case, the mixed and mapped codebook has an advantage of 3 bits/frame over the clean codebook. Furthermore, the difference mapping in these two cases produces the same distortion level as the individual mapping for higher bit rates.

The 3-SVQ is then tested for a bandpass input speech and the results are illustrated in Fig. 4.11. Again, the clean codebook has the worst performance out of all four designs. For low bit rates, the mapped codebook has an advantage of 5 bits/frame over the mixed codebook. Furthermore, unlike in 2-SVQ, the performance difference between the mixed and mapped codebooks in 3-SVQ is no longer negligible for high bit rates. This can be explained by the fact that 3-SVQ has many fewer codewords than 2-SVQ. For example, at 30 bits, a 2-SVQ has 32 768 codevector entries while a 3-SVQ has only 1 024. Hence, a 2-SVQ has 31 744 more codevectors to represent a wider range of possible LP parameters.

In summary, mapping codebooks generate consistent performances over all bit rates. For lower bit rate frames, the importance of using transformation functions becomes more critical since the numbers of codevectors entries is small and therefore can not satisfy the pdfs of a diverse input vectors.

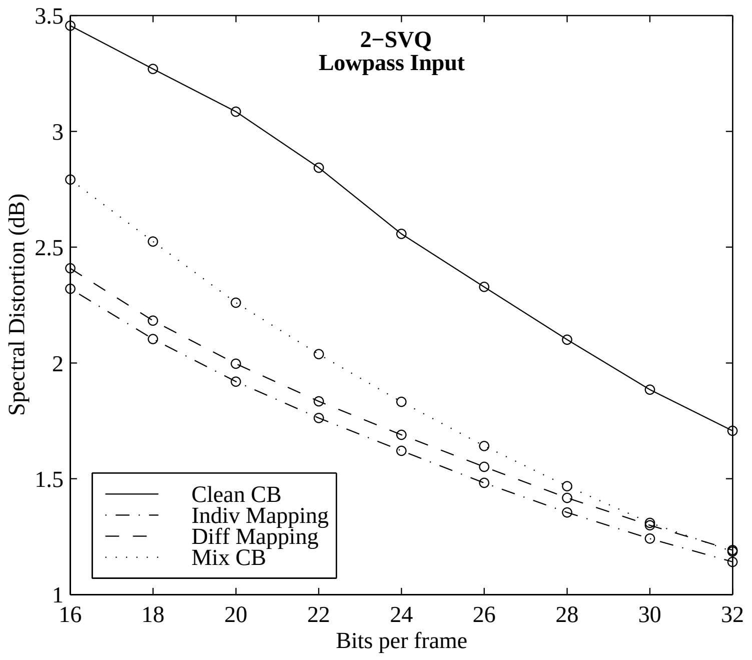


Fig. 4.8 SD performance of 2-SVQ for clean, mixed, individual and difference mapping for *lowpass* filtered speech over different bit rates.

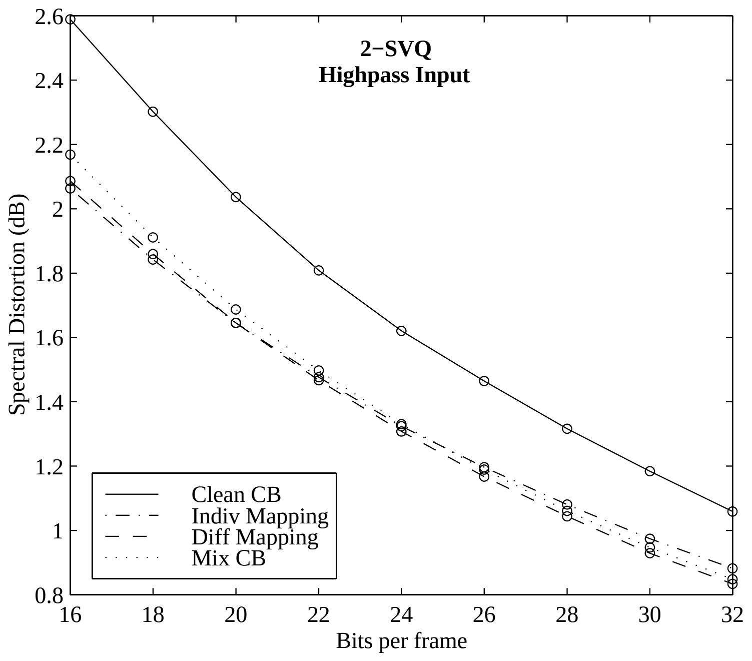


Fig. 4.9 SD performance of 2-SVQ for clean, mixed, individual and difference mapping for *highpass* filtered speech over different bit rates.

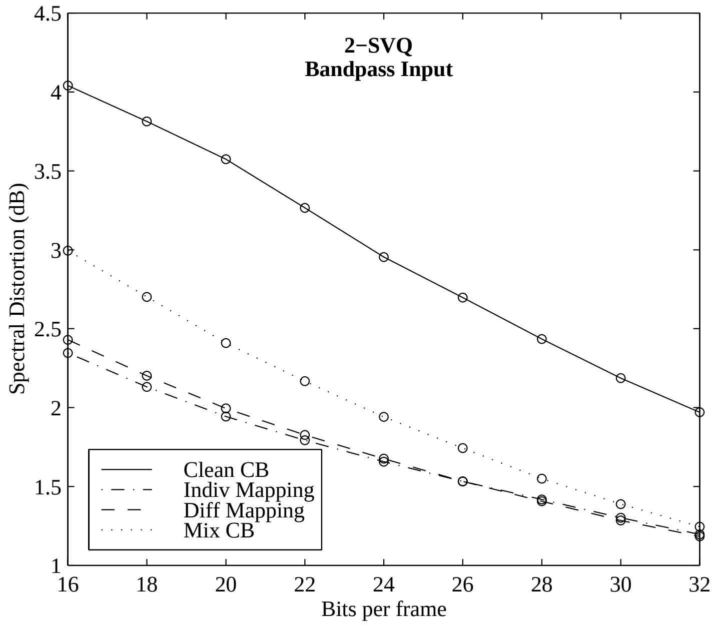


Fig. 4.10 SD performance of 2-SVQ for clean, mixed, individual and difference mapping for *bandpass* filtered speech over different bit rates.

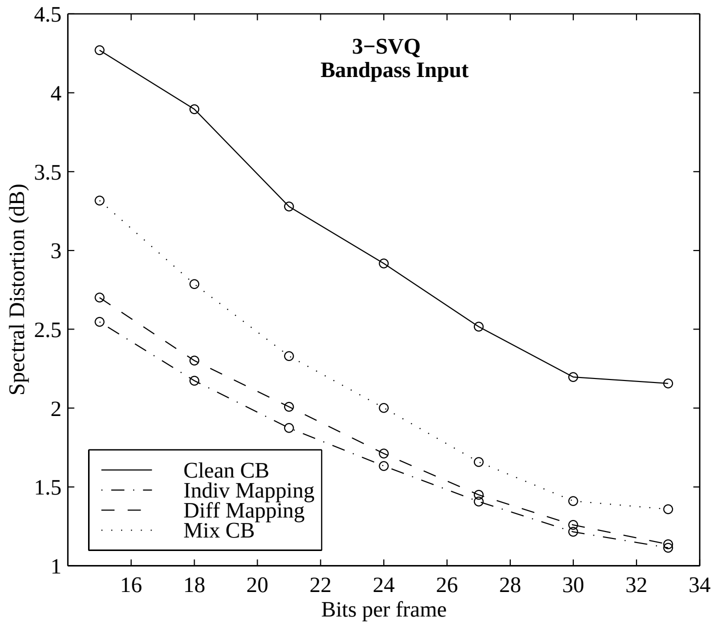


Fig. 4.11 SD performance of 3-SVQ for clean, mixed, individual and difference mapping for *bandpass* filtered speech over different bit rates.

4.5 Interpolation Performance in the Mapping Model

Since filters can have different frequency responses, one needs to develop a method that could accommodate different cutoff frequencies. bandwidth and cutoff frequencies. As described in Section 3.2.4, the mapping model can use the *interpolation method* to compensate for that effect. In this section, we implement filters with various cutoff frequencies and observe how well this interpolation method performs.

Figure 4.12 and 4.13 shows the performance of interpolated individual mapping and interpolated difference mapping respectively. The clean and codebook performance results are presented as well to confirm the improvement of the mapping model over all cutoff frequencies. Interpolation can be used to prevent the calculation of the mapping functions for each filtering condition. In conclusion, interpolation allows the mapping model to operate with reasonable performance on any kind of filtering conditions.

4.6 Summary

In conclusion, the mapping model generally gives a significant improvement in which a lower spectral distortion and a lower percentage of outliers are yielded. It was also tested on different bit rates and different cutoff frequencies and the results are satisfactory. Individual mapping performed better in most cases than the difference mapping but suffers from a higher filter unstability caused by the transformation.

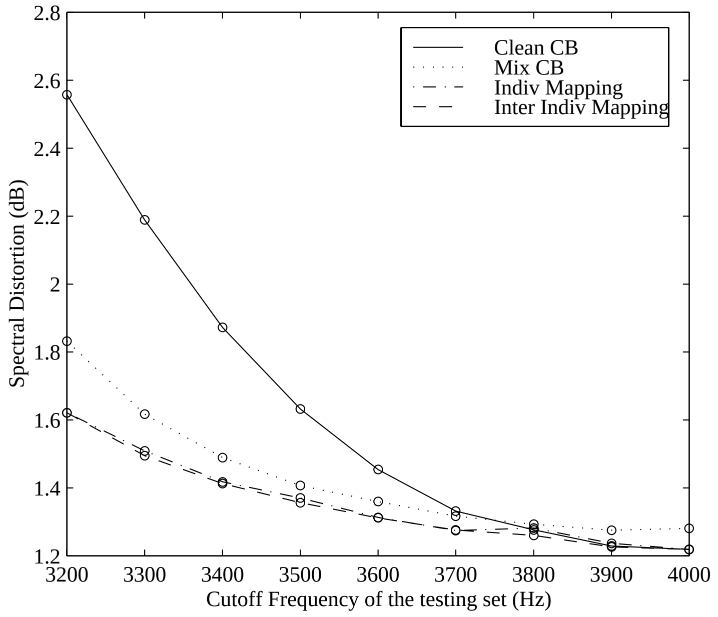


Fig. 4.12 SD performance of the individual mapping over different cut-off frequencies for *lowpass* filtered spectral parameters.

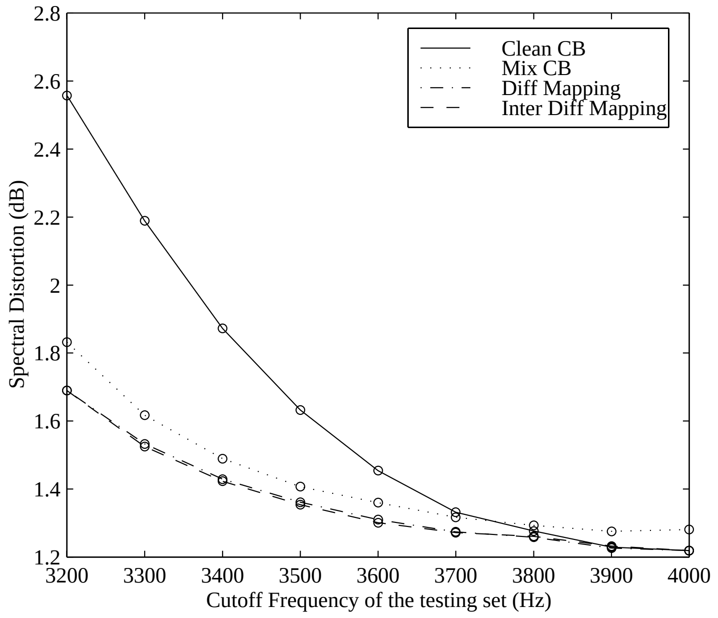


Fig. 4.13 SD performance of the difference mapping over different cut-off frequencies for *lowpass* filtered spectral parameters.

Chapter 5

Final Remarks and Future Work

There is a continuing need for robust coding of digital speech signals at ever lower bit rates. Linear predictive coders are most commonly used to model frames of speech and to extract its significant features. In this thesis we have studied a novel method to tackle the robustness of vector quantization, used to encode the speech spectral parameters, when the speech data is subjected to different filtering conditions. Considering a speech coder in a wireless base station, the input speech can come from wireline sources or decoded speech from other wireless links. In other words, the input to the base station may have been subjected to quite different filtering conditions.

Section 5.1 summarizes our work, and Section 5.2 contains suggestions for future research in robust spectral quantization.

5.1 Summary of Our Work

In Chapter 1, we have presented a brief overview of speech coding and most of the well known properties of speech. In speech coding, quantization is a critical block and needs to be used in the most efficient manner. Vector quantization is one way of coding the spectral parameters but can lack of robustness.

Chapter 2 provided a review of linear predictive analysis of speech. The formant structure of speech is modeled by an all-pole filter. Alternative parametric representations of the filter coefficients such as reflection coefficients and line spectral frequencies are introduced. Furthermore, several objective distortion methods are defined for the purpose of quantization and performance estimation. Information on the database is also presented.

Vector quantization was introduced in Chapter 3 as an extension to scalar quantization. It is often used to encode the LP filter coefficients as a single entity. VQ can exploit any correlation that exists among the vector components. The optimality of VQ is explained and a method for designing a codebook is presented. Unconstrained VQ of an entire LSF frame vector is costly in terms of codebook storage and search complexity. Split vector quantization, on the other hand, is a way of reducing both problems faced in vector quantization.

VQ performance is dependent on the way the training set is constructed. A problem with robustness arises whenever the testing set has a different distribution than the training set. Codebook transformation is a method that solves this problem by mapping the codebook to match the new distribution. The transformation model and its constraints were introduced in the second part of Chapter 3. In particular, two techniques of mapping, individual and difference, were discussed.

Different representations were investigated in Chapter 4 to see which one is more suitable to be transformed. Predictor coefficients suffers from the lack of consistency and difficulty with stability while reflection coefficients have no apparent interpretation of the transformed pdfs. Line spectral frequencies are the most appropriate representation due to the localized distinction between processed speech and possess an easy stability check.

Several optimization methods are applied on the mapping model to increase its performance, that is to have a lower spectral distortion and a lower resulting unstable filters. The spectral distortion of the mapping model is then compared to a clean codebook and a mixed codebook. The mapped codebook records an improvement of 0.9 dB and 0.2 dB in spectral distortion over the mixed and clean codebook respectively. The mapping model is then tested for different bit rates which shows that this model might not be that effective when higher bit rates are applied, but becomes essential when the number of allocated bits per frame are small. To complete the transformation model, an interpolation method has been used and analyzed to support different cutoff frequencies.

5.2 Future Considerations

In this thesis, we have studied a transformation model to improve the robustness of vector quantization, more specifically split vector quantization. This method can also be tested on *Matrix quantization* (MQ) [60, 61]. MQ groups together a sequence of successive frame vectors and encodes it as a single matrix. Similar to VQ, MQ can have *split matrix quantization* [62, 63].

We have noted that some distributions were not properly mapped. For example in difference mapping, a change in the tilt can be introduced to the pdfs by the filtering process. A method needs to be explored to change the skewness of the probability density function by some appropriate transformation.

Furthermore, the integration of the frequency cutoff estimator and the interpolation method should be evaluated to see how effective the estimation of bandwidth is.

Finally, more testing can be done to generalize the transformation procedure to distinguish female and male differences, individual speaker variations and various background noise as well. Any additional knowledge of these types of difference that can reflect any behavior in the probability density functions of the LP parameters can be used to generalize the mapping method.

References

[1] K. W. Gould, R. V. Cox, N. S. Jayant, and M. J. Melchner, "Robust speech coding for the indoor wireless channel," *AT&T Technical Journal.*, pp. 64–73, Jul 1993.

[2] M. Phythian, L. John, and S. Sridha, "Robust speech coding for the preservation of speaker identity," in *Proc. Int. Symp. Signal Proc. Appl.*, (New Jersey), pp. 395–398, 1996.

[3] R. P. Ramachandran, M. M. Sondhi, N. Seshadri, and B. S. Atal, "A two codebook format for robust quantization of line spectral frequencies," *IEEE Trans. Speech and Audio Proc.*, vol. 3, pp. 157–168, May 1995.

[4] J. R. Deller, Jr., J. G. Proakis, and J. H. L. Hansen, *Discrete-Time Processing of Speech Signals.* New York: MacMillan, 1993.

[5] N. Jayant and P. Noll, *Digital Coding of Waveforms: Principles and Applications to Speech and Video.* Englewood Cliffs, New Jersey: Prentice-Hall, 1984.

[6] D. O'Shaugnessy, *Speech Communication: Human and Machine.* Reading, MA: Addison-Wesley, 1987.

[7] J. Makhoul, S. Roucos, and H. Gish, "Vector quantization in speech coding," *Proceedings of the IEEE*, vol. 73, pp. 1551–1558, November 1985.

[8] H. Abut and S. A. Luse, "Vector quantizers for subband coded waveforms," in *Proc. Int. Conf. Acoust., Speech, Signal Proc.*, (San Diego, CA), p. paper 10.6, March 1984.

[9] V. Cuperman and A. Gersho, "Vector predictive coding of speech at 16 kbits/s," *IEEE Trans. Communications*, vol. COM-33, pp. 685–696, July 1985.

[10] N. R. Dixon and T. B. Martin, *Automatic Speech and Speaker Recognition.* New York, New York: IEEE-PRESS, 1979.

[11] P. Knagenhjelm, "How good is your index assignment," in *Proc. Int. Conf. Acoust., Speech, Signal Proc.*, (Minneapolis, MN), pp. II–423–II–426, May 1993.

[12] J. Makhoul, "Linear prediction: A tutorial review," *Proceedings of the IEEE*, vol. 63, pp. 561–580, April 1975.

[13] R. Salami, C. Laflamme, J. P. Adoul, and D. Massaloux, “A toll quality 8 kb/s speech codec for the personal communications system (PCS),” *IEEE Trans. Vehicular Tech.*, vol. 43, pp. 808–816, August 1994.

[14] G. Golub and C. Loan, *Matrix Computations.* Baltimore: Johns Hopkins University Press, 1996.

[15] J. Leroux and C. Gueguen, “A fixed point computation of partial correlation coefficients,” *IEEE Trans. Acoust., Speech, Sig. Proc.*, vol. ASSP-25, no. 3, pp. 257–259, 1979.

[16] S. Haykin, *Adaptive Filter Theory.* Upper Saddle River: Prentice Hall, third ed., 1996.

[17] D. Watkins, *Fundamentals of Matrix Computation.* New york: John Wiley and sons, 1991.

[18] B. S. Atal and M. R. Schroeder, “Predictive coding of speech signals and subjective error criteria,” *IEEE Trans. Acoust., Speech, Signal Proc.*, vol. ASSP-27, pp. 247–254, June 1979.

[19] B. S. Atal, “Predictive coding of speech at low bit rates,” *IEEE Trans. Communications*, vol. COM-30, pp. 600–614, April 1982.

[20] Y. Tohkura, F. Itakura, and S. Hashimoto, “Spectral smoothing techniques in PARCOR speech analysis-synthesis,” *IEEE Trans. Acoust., Speech, Signal Proc.*, vol. ASSP-26, pp. 587–596, December 1978.

[21] R. Viswanathan and J. Makhoul, “Quantization properties of transmission parameters in linear predictive systems,” *IEEE Trans. Acoust., Speech, Signal Proc.*, vol. ASSP-23, pp. 309–321, 1978.

[22] W. Kleijn and K. Paliwal, *Speech Coding and Synthesis.* Amsterdam, Netherlands: Elsevier, 1995.

[23] J. D. Markel and A. H. Gray, *Linear Prediction of Speech.* Berlin, Germany: Springer Verlag, 1976.

[24] A. H. Gray and J. D. Markel, “Quantization and bit allocation in speech processing,” *IEEE Trans. Acoust., Speech, Signal Proc.*, vol. ASSP-24, pp. 459–473, 1976.

[25] F. Itakura, “Line spectrum representation of linear prediction coefficients of speech signals,” *Journal Acoustical Society America*, vol. 57, p. S35, 1975. (abstract).

[26] F. Soong and B.-H. Juang, “Line spectrum pair and speech data compression,” in *Proc. Int. Conf. Acoust., Speech, Signal Proc.*, (San Diego, CA), pp. 1.10.1–1.10.4, March 1984.

[27] R. Gnanasekaran, “A note on the new 1–D and 2–D stability therorems for discrete systems,” *IEEE Trans. Acoust., Speech, Sig. Proc.*, vol. ASSP-29, pp. 1211–1212, December 1981.

[28] F. Soong and B.-H. Juang, “Optimal quantization of LSP parameters,” *IEEE Trans. Speech and Audio Proc.*, vol. 1, pp. 15–24, January 1993.

[29] P. Kabal and R. P. Ramachandran, “The computation of line spectral frequencies using Chebyshev polynomials,” *IEEE Trans. Acoust., Speech, Signal Proc.*, vol. ASSP-34, pp. 1419–1426, December 1986.

[30] L. R. Rabiner and R. Schafer, *Digital Processing of Speech Signals.* Englewood Cliffs, New Jersey: Prentice-Hall, 1979.

[31] R. Hagen, “Spectral quantization of cepstral coefficients,” in *Proc. Int. Conf. Acoust., Speech, Signal Proc.*, (Adelaide), pp. I–509–I–512, April 1994.

[32] A. H. Gray and J. D. Markel, “Distance measures for speech processing,” *IEEE Trans. Acoust., Speech, Signal Proc.*, vol. ASSP-24, pp. 380–391, October 1976.

[33] S. Wang, A. Sekey, and A. Gersho, “An objective measure for predicting subjective quality of speech coders,” *IEEE Journal Sel. Areas in Communications*, vol. 10, pp. 819–829, June 1992.

[34] B. J. McDermott, C. Scagliola, and D. J. Goodman, “Perceptual and objective evaluation of speech processed by adaptive differential PCM,” in *Proc. Int. Conf. Acoust., Speech, Signal Proc.*, (Tulsa), pp. 581–585, April 1978.

[35] J. M. Tribolet, M. P. Noll, and B. J. McDermott, “A study of complexity and quality of speech waveform coders,” in *Proc. Int. Conf. Acoust., Speech, Signal Proc.*, (Tulsa), pp. 586–590, April 1978.

[36] S. Quackenbush, T. Barnwell, and M. Clements, *Objective Measures of Speech Quality.* New Jersey: Prentice Hall, 1988.

[37] L. R. Rabiner and B. H. Juang, *Fundamentals of Speech Recognition.* Englewood Cliffs, New Jersey: Prentice-Hall, 1993.

[38] B. Atal, R. Cox, and P. Kroon, “Spectral quantization and interpolation for CELP coders,” in *Proc. Int. Conf. Acoust., Speech, Signal Proc.*, (Glasgow), pp. 69–72, 1989.

[39] F. Soong and M. M. Sondhi, “A frequency-weighted Itakura spectral distortion measure and its application to speech recognition in noise,” in *Proc. Int. Conf. Acoust., Speech, Signal Proc.*, (Dallas, TX), pp. II–625–II–628, April 1987.

[40] K. K. Paliwal, “On the performance of the quefrency-weighted cepstral coefficients in vowel recognition,” *Speech Commun.*, pp. 151–154, May 1982.

[41] Y. Tohkura, “A weighted cepstral distance measure for speech recognition,” *IEEE Trans. Acoust., Speech, Signal Proc.*, vol. ASSP-35, pp. 1414–1422, October 1987.

[42] B. H. Juang, L. R. Rabiner, and J. G. Wilpon, “On the use of bandpass liftering in speech recognition,” in *Proc. Int. Conf. Acoust., Speech, Signal Proc.*, (Tokyo), pp. 765–768, April 1986.

[43] K. K. Paliwal and B. S. Atal, "Efficient vector quantization of LPC parameters at 24 bits/frame," *IEEE Trans. Speech and Audio Proc.*, vol. 1, pp. 3–14, January 1993.

[44] J. Grass, "Quantization of predictor coefficients in speech coding," M.Eng. thesis, McGill University, Department of Electrical Engineering, September 1990.

[45] R. Laroia, N. Phamdo, and N. Farvardin, "Robust and efficient quantization of speech LSP parameters using structured vector quantizers," in *Proc. Int. Conf. Acoust., Speech, Signal Proc.*, (Toronto), pp. 641–644, May 1991.

[46] F. F. Tzeng, "Analysis-by-synthesis linear predictive speech coding at 2.4 kbit/s," in *Proc. Globecom*, pp. 1253–1257, 1989.

[47] W. P. Leblanc, B. Bhattacharya, S. A. Mahmoud, and V. Cuperman, "Efficient search and design procedures for robust multi-stage VQ of LPC parameters for 4 kb/s speech coding," *IEEE Trans. Speech and Audio Proc.*, vol. 1, pp. 373–385, October 1993.

[48] E. Paksoy, W.-Y. Chan, and A. Gersho, "Vector quantization of speech LSF parameters with generalized product codes," in *Proc. Int. Conf. Spoken Language Proc.*, (Banff, Canada), pp. 33–36, October 1992.

[49] IEEE Subcommittee on Subjective Measurements, "IEEE recommended practice for speech quality measurements," *IEEE Trans. Audio and Electroacoustics*, vol. AU-17, pp. 225–246, Sept. 1969. IEEE Standards Publication No. 297.

[50] J. D. Gibson, *Principles of Digital and Analog Communications.* New York: Macmillan, 1992.

[51] T. Lookabaugh and R. M. Gray, "High-resolution quantization theory and the vector quantizer advantage," *IEEE Trans. Information Theory*, vol. IT-35, pp. 1020–1033, September 1989.

[52] R. M. Gray, *Source Coding Theory.* Boston: Kluwer Academic Press, 1990.

[53] A. Gersho and R. M. Gray, *Vector Quantization and Signal Compression.* Boston: Kluwer Academic Press, 1992.

[54] W. H. Equitz, "A new vector quantization clustering algorithm," *IEEE Trans. Acoust., Speech, Sig. Proc.*, pp. 1568–1575, October 1989.

[55] I. Katsavounidis, C.-C. J. Kuo, and Z. Zhang, "A new initialization technique for Generalized Lloyd Algorithm," *IEEE Signal Processing Letters*, vol. 1, pp. 144–146, October 1994.

[56] Y. Linde, A. Buzo, and R. M. Gray, "An algorithm for vector quantizer design," *IEEE Trans. Communications*, vol. COM-28, pp. 84–95, January 1980.

[57] J. S. Collura, "Vector quantization of linear predictor coefficients," in *Modern Methods of Speech Processing* (R. P. Ramachandran and R. J. Mammone, eds.), Kluwer Academic Press, 1995.

[58] W.-Y. Chan and A. Gersho, "Generalized product code vector quantization: A family of efficient techniques for signal compression," *Digital Signal Processing*, vol. 4, pp. 95–126, April 1994.

[59] J. Grass and P. Kabal, "Quantization of predictor coefficients in speech coding," Tech. Rep. 91-01, INRS-Télécommunications, Verdun, Canada, 1991.

[60] D. Y. Wong, B. H. Juang, and D. Y. Cheng, "Very low data rate speech compression with LPC vector and matrix quantization," in *Proc. Int. Conf. Acoust., Speech, Signal Proc.*, (Boston, MA), pp. 65–68, April 1983.

[61] C. Tsao and R. M. Gray, "Matrix quantizer design for LPC speech using the Generalized Lloyd Algorithm," *IEEE Trans. Acoust., Speech, Sig. Proc.*, vol. ASSP-33, pp. 537–545, June 1985.

[62] T. Ohya, H. Suda, and T. Miki, "5.6 kbits/s PSI-CELP of the half-rate PDC speech coding standard," in *Proc. IEEE Vehic. Tech. Conf.*, pp. 1680–1684, 1994.

[63] W.-Y. Chan, I. A. Gerson, and T. Miki, "Half-rate standards," in *The Mobile Communications Handbook* (J. D. Gibson, ed.), CRC Press, 1995.