Improved Pitch Modelling for Low Bit-Rate Speech Coders

Costantinos Papacostantinou

Department of Electrical Engineering
McGill University
Montreal, Canada

August 1997

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

During the last several years, there has been a dramatic growth of digital services, such as digital wireless and wireline communications, satellite communications and digital voice storage systems. Such services require the use of high-quality low bit-rate coders to efficiently code the speech signal before transmission or storage. The majority of such coders employ algorithms that are based on Code-Excited Linear Prediction (CELP).

The goal of this thesis is to improve the quality of CELP coded speech, while keeping the basic coding format intact. The quality improvement is focused on voiced speech segments. A Pitch Pulse Averaging (PPA) algorithm has been developed to enhance the periodicity of such segments, where during steady state voicing the pitch pulse waveforms in the excitation signal evolve slowly in time. The PPA algorithm extracts a number of such pitch pulse waveforms from the past excitation, aligns them, and then averages them to produce a new pitch pulse waveform with reduced noise.

The PPA algorithm has been simulated and tested on a floating point C-simulation of the G.729 8 kbps CS-ACELP coder. Objective tests verified that the algorithm contributes most during steady state voiced speech. Thus a simple voicing decision mechanism has been developed to deactivate the algorithm during unvoiced segments and voicing onsets of speech. Results verified that the algorithm has generally improved the periodicity of voiced segments by reducing the average of the weighted mean-squared error.

While we were able to demonstrate improvements in objective measures, informal listening tests indicate that the already high perceptual quality of G.729 is generally not audibly altered. Nonetheless, the technique may be useful for improving the quality at lower rates, particularly for next generation low bit-rate coders operating near 4 kbps.

Acknowledgments

First I would like to thank my supervisor Prof. Peter Kabal for his guidance, continuous support and experienced advice that made the completion of this thesis possible.

Special thanks to my family and loved ones for their love, support, encouragement and understanding during the last 5 years at McGill University.

Finally I would like to thank my colleagues here at McGill for their help and useful suggestions.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Motivation for Our Research

In the last several years, there has been an explosion of Research and Development activity in the area of speech compression (coding). The growth of applications such as mobile communications and voice storage systems has increased the need to conserve bandwidth in wireless, wireline and satellite communications as well as to reduce memory requirements of voice storage systems. Since the bandwidth of a signal is a function of its bit-rate, high-quality low bit-rate coders have been the major focus of current research in this field.

Many speech coding algorithms have been developed for coding telephone bandwidth (200 Hz – 3.4 kHz) speech signals. Amongst these, Code-Excited Linear Prediction (CELP) [1, 2] is the most widely studied and promising algorithm for high quality speech at low to medium (4 kbps – 8 kbps) bit-rates. This family of techniques, exploit models of human speech production and auditory perception which enables them to provide a quality versus bit-rate tradeoff that outperforms most existing compression techniques at these rates. Its popularity is witnessed by the fact that numerous CELP based coders have been standardized for various applications and are widely used in the commercial world.

For example, the FS1016 4.8 kbps CELP coder [3] has been standardized by the U.S. Department of Defense (DoD) for secure voice terminals. The G.728 16 kbps LD-CELP [4] and lately the G.729 8 kbps CS-ACELP [5], were standardized by the International Telecommunications Union - Telecommunications Sector (ITU-T) for use in Personal Mobile Communications, digital satellite communications, store and forward systems etc. Currently in Europe, the European Telecommunications Standards Institute (ETSI), is moving towards the standardization of the new Global System for Mobile Communications Enhanced Full Rate (GSM-EFR) coder, targeted for new market groups such as Personal Communications Services (PCS) applications. In North America, the Telecommunications Industry Association (TIA) has standardized the IS-96 8.5 kbps QCELP [6] for Code Division Multiple Access (CDMA) digital cellular telephony and currently, efforts are made towards the standardization of the new IS-127 9.6 kbps Enhanced Variable Rate Codec (EVRC). In addition, a 7.4 kbps ACELP coder has been proposed for the new TIA IS-641 Enhanced Full Rate Codec to be used in Time Division Multiple Access (TDMA) digital cellular systems. Finally, the Research and Development Center for Radio Systems (RCR) in Japan has standardized the Japanese Digital Cellular (JDC) Half-Rate 5.6 kbps PSI-CELP coder [7] to double the capacity of the Japanese TDMA personal digital cellular system.

All of the examples of standards outlined above are based on CELP algorithms, appropriately modified and enhanced to meet the required specifications. Because of this wide installation base of CELP coders in the commercial world, there is a need to further improve the quality of CELP coded speech while keeping the basic encoding format intact.

1.2 Problem Statement and Objectives of Our Research

It is widely accepted that reconstructed speech in CELP coders suffers from some quality degradation which can be described as hoarseness. This degradation is caused by the poor reproduction of the input speech signals periodicity and the use of a noisy stochastic excitation which elevates the perceived background noise level during voiced speech, alternatively viewed as inter-harmonic noise in the speech spectrum. The perceived level of periodicity in the reconstructed signal tends to decrease with decreasing bit rate and with high pitch input speech signals, in the range of 200 Hz – 400 Hz.

Several algorithms [7, 8, 9, 10, 11, 12, 13, 14, 15] have been proposed to reduce the presence of noise between the harmonics, while maintaining reasonable computational complexity. Various filtering techniques attempted to reduce coding noise between the harmonics. Other proposals deliberately limited the amount of noisy stochastic excitation thus minimizing the amount of inter-harmonic noise. These techniques will be presented in more detail in Chapter 3.

In this thesis we describe a novel approach to enhance the periodicity of the synthesized speech, based on the evolution of pitch pulses during onsets and steady-state voiced segments of speech. The primary goal of our work is to perceptually enhance the quality of synthetic speech in the CELP algorithm. We verify our claims by tests carried out on a floating point version of the G.729 8 kbps CS-ACELP coder, standardized by ITU-T in 1996.

1.3 Organization of the Thesis

In Chapter 2 we provide a basic background theory on linear prediction and perceptual weighting which constitute an integral part of an important class of coders referred to as *linear prediction based analysis by synthesis* (LPAS) coders. The CELP algorithm, which belongs to the general class of LPAS coders, will then be described, emphasizing the way the excitation is represented. In Chapter 3 we introduce the notion of the *adaptive codebook* as used in CELP coders to model the periodic segments of speech and examine several techniques proposed towards the enhancement of periodicity. The development of the pitch pulse averaging (PPA) technique is then presented. Chapter 4 describes the implementation of the PPA technique and includes simulation results and performance evaluations. In conclusion, Chapter 5 summarizes our work and offers suggestions for future investigation.

Chapter 2

Analysis-by-Synthesis Linear Prediction

2.1 Model of Speech Production

Speech is generally generated by exhaling air through the glottis and the vocal tract, as shown in Fig. 2.1. The airflow, coming from the lungs, is modulated by the vibrations of the vocal cords and the shape of the vocal tract.

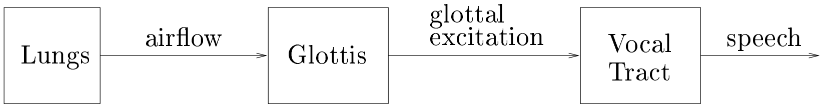


Fig. 2.1 Human speech production model

Speech can be classified into two general categories:

Voiced speech characterized by quasi-periodic and in general high energy segments of sounds such as vowels.

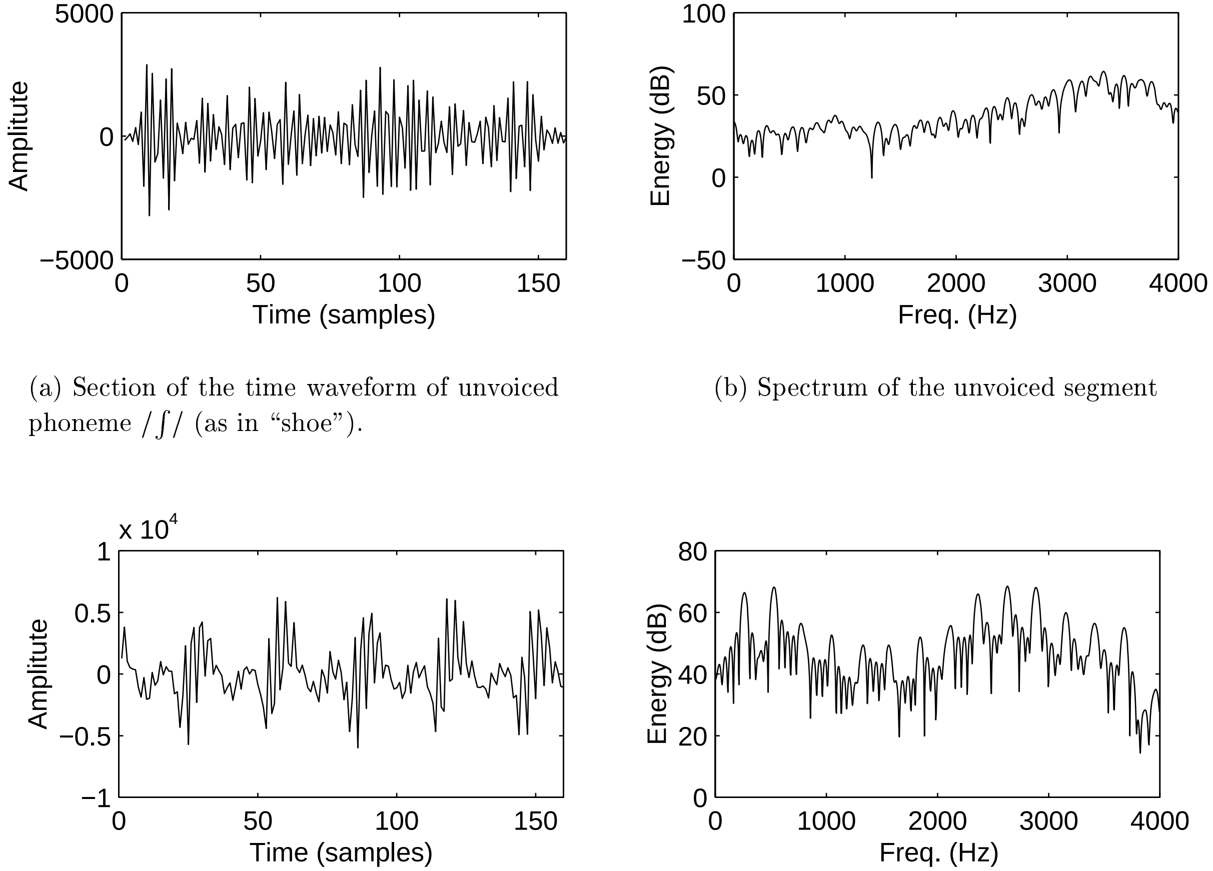
Unvoiced speech which generally describes the low energy segments such as consonants.

Voiced speech is produced when the air flowing from the lungs is interrupted by a periodic opening and closing of the vocal cords, generating a periodic glottal excitation for the vocal tract. The rate at which the vocal cords open and close is called the *fundamental frequency* (denoted as F0) and corresponds to the physically perceived *pitch*. Its value varies with the size of the vocal cords. Typical average values are 150 Hz and 250 Hz for males and females respectively. Since F0 and the vocal tract shape changes over time, voiced speech is not truly periodic but can be characterized as *quasi-periodic* over short intervals of time. Unvoiced speech is produced when the vocal cords do not vibrate and the vocal tract is excited by a turbulent noise generated when the air flowing from the lungs passes through a narrow constriction in the vocal tract.

Speech production can be alternatively viewed as a filtering operation in which a sound source excites a vocal tract filter [16]. The sound source represents the noise generated at a constriction of the vocal tract during unvoiced sounds or the glottal pulses during voicing, or a combination of the two. The spectrum of the sound source during voiced sounds, contains harmonics spaced by F0 with most of the energy concentrated at low frequencies, whereas during unvoiced sounds the spectrum is approximately flat and without harmonic structure.

The vocal tract will finally modify the distribution of energy in the spectrum of the sound source (depending on its particular shape) and introduce resonances (*formants*) and anti-resonances. Representing the vocal tract as a time-varying filter, the resonances and anti-resonances are due to the poles and zeros of the vocal tract frequency response, respectively. Typical time waveforms of voiced and unvoiced segments of speech along with their spectrum are shown in Fig. 2.2. Note the noise-like character of the unvoiced waveform in Fig. 2.2(a), compared to the periodic character of the voiced waveform of Fig. 2.2(c) where sections of the waveform are repeated approximately every 40 samples. The periodicity in the voiced waveform also appears in its spectrum (see Fig. 2.2(d)) as well defined peaks (harmonics), spaced at the fundamental frequency (in this case, approximately 200 Hz for an 8 kHz sampled signal). The spectrum of the unvoiced waveform has no such structure and is more random.

Low bit-rate coders try to reduce the bit rate, while preserving speech quality, by taking advantage of redundancies in the speech signal and perceptual limitations of the human ear [16, 17]. The former arises from the following observations: (a) in general the speech spectrum changes relatively slowly (except during the articulation of stops), (b) successive pitch periods are generally similar and (c) the spectral envelope is relatively smooth, with most of the energy concentrated at low frequencies. These are attributed to the mechanical limitations of the speech organs, i.e vocal tract and vocal cords. The latter, relates to the fact that the human ear is insensitive to phase, more sensitive to low than high frequencies



(a) Section of the time waveform of unvoiced phoneme /ʃ/ (as in "shoe").

(b) Spectrum of the unvoiced segment

(c) Section of the time waveform of voiced phoneme /i/ (as in "beat").

(d) Spectrum of the voiced segment

Fig. 2.2 Examples of (a) noise-like unvoiced and (c) quasi-periodic voiced segments of speech. The speech signal is sampled at 8000 samples/sec and each segment is 20 msec (160 samples) long.

and places more importance to spectral poles than spectral zeros due to masking effects.

The redundancy in the speech signal led to the conclusion that speech samples are correlated. The spectral envelope corresponds to the short-term correlations and the harmonic structure corresponds to the long-term correlations. These correlations can be exploited to yield a lower bit rate by using *linear prediction*.

2.2 Linear Prediction

Linear prediction is one of the most important tools in speech analysis. Its relative simplicity of computation and its ability to provide accurate estimates of the speech parameters, make this method predominant in low bit-rate coding of speech. The philosophy behind linear prediction is that a speech sample can be approximated as a linear combination of past samples. Then, by minimizing the sum of the squared differences between the actual speech samples and the linearly predicted ones over a finite interval, a unique set of predictor coefficients can be determined.

Prediction can be used to either remove redundancies from the speech signal, or to create a model for the vocal tract. The redundancy removal is performed with a linear prediction (LP) filter. During short-term prediction, i.e removal of near-sample redundancies, this filter is commonly called the *LP analysis filter*. The LP analysis filter removes the formant structure of the speech signal (thus also referred to as *formant analysis* filter) and leaves a lower energy output prediction error which is often called the *LP residual* or *excitation* signal. The inverse LP analysis filter, i.e., the *LP synthesis* filter, models the vocal tract and its transfer function describes the spectral envelope of the speech signal. As the spectral envelope is also referred to as “formant structure” of the signal, thus the *LP synthesis* filter is also known as the *formant synthesis* filter. Further refinement can be obtained by considering the long-term correlations of voiced speech using long-term prediction. In this case another LP filter can be used to remove far-sample redundancies. This filter is usually called the *pitch predictor* and exploits the periodicity of the signal. The inverse of the pitch predictor, often called the *pitch filter*, models the effect of the glottis and its transfer function describes the harmonic structure of the speech signal. Pitch prediction will have no useful effect for unvoiced speech since the unvoiced excitation is random and its spectrum is flat.

2.2.1 Short-term prediction

The source-filter model allows us to use linear prediction to remove short-term redundancies from the speech signal. Within a frame of N speech samples, the speech signal $s[n]$ can be considered to be the output of some system with some unknown input excitation $u[n]$ such the following relation holds [18]:

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

where $\{a_k\}$, $\{b_l\}$ and gain G are system parameters. In the above equation, the speech signal is *predicted* as a *linear combination* of past outputs and past and present inputs. The z-transform of the system, $H(z)$, is thus given by

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

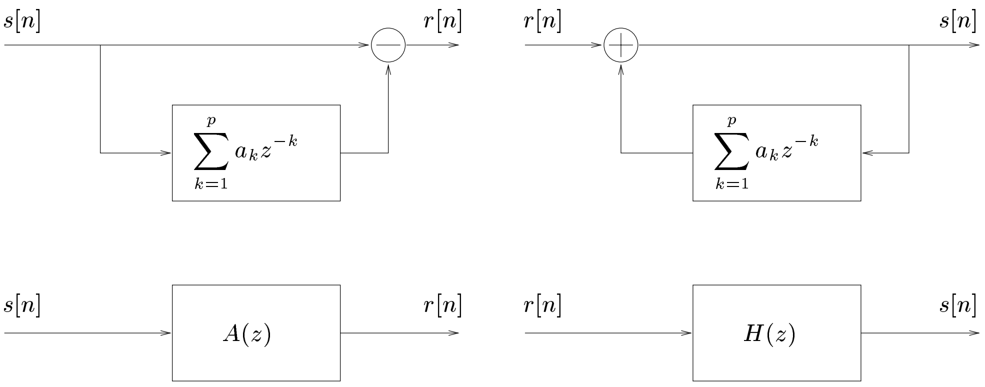
where $S(z)$ and $U(z)$ are the z-transforms of $s[n]$ and $u[n]$ respectively.

$H(z)$ in Eq. (2.2), is the general pole-zero model and is also known as the *autoregressive moving average* (ARMA) model. The polynomial roots of the numerator and denominator of Eq. (2.2) correspond to the zeros and poles, respectively, of the system. This model can be divided into two special cases: (1) When $a_k = 0$ for $k = 1, \ldots, p$, $H(z)$ becomes an all-zero model, alternatively known as the *moving average* (MA) model, and (2) an all-pole, or *autoregressive* (AR) model, when $b_l = 0$ for $l = 1, \ldots, q$.

The all-pole model is preferred for most applications because it is computationally more efficient and fits the acoustic tube model for speech production [17]. Although it provides a very good representation of the vocal tract effects during vowel sounds which are acoustically resonant, it is only an approximation for phoneme classes such as nasal and fricatives which contain spectral nulls that are best modeled by the zeros of the vocal tract transfer function. Nevertheless, as stated earlier, the human ear is more sensitive to spectral poles than spectral nulls which makes the simplification a reasonable assumption. In addition, it has been shown that the effect of a zero in the transfer function can be achieved by including more poles [19].

Based on the all-pole model, the current speech sample is predicted by a linear combination of p past samples; i.e.,

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



(a) Formant Prediction.

(b) Formant Synthesis stage.

Fig. 2.3 Block diagrams of formant (a) analysis and (b) synthesis stages.

and the output $r[n]$, called prediction error or LP residual signal, given as

$$r[n] = s[n] - \sum_{k=1}^{p} a_k s[n-k]. \tag{2.4}$$

Taking the z-transform of both sides in Eq. (2.4), it follows that

$$R(z) = A(z)S(z), \tag{2.5}$$

where $R(z)$ is the z-transform of the LP residual signal, and

$$A(z) = 1 - \sum_{k=1}^{p} a_k z^{-k}. \tag{2.6}$$

The filter $A(z)$ is known as the *LP analysis filter*. The all-pole *LP synthesis filter* $H(z)$,

$$H(z) = \frac{1}{A(z)}, \tag{2.7}$$

models the short-time power spectral envelope of the speech signal. A block diagram of the format analysis and synthesis stage is shown in Fig. 2.3.

The choice of the order p of the model is a compromise among spectral accuracy, computation time/memory and transmission bit rate. In general, a pair of poles is allowed for each formant present in the speech spectrum, plus an additional 2 – 4 poles to approximate possible zeros. For 8 kHz sampled speech, p typically ranges from 8 to 16.

Linear prediction can be classified as *forward adaptive*, in which case the prediction is based on past speech samples and the prediction coefficients have to be transmitted to the receiver as side information, and *backward adaptive* where prediction is based on past reconstructed speech samples $\hat{s}[n]$. In the latter, no side information need to be transmitted to the receiver. For the purpose of this thesis, as in most low bit-rate coders, forward adaptive linear prediction is assumed.

The filter coefficients $\{a_k\}$ (also known as LP coefficients) are estimated every frame of speech samples. This is done by using either the *least-squares* or *lattice* method [16, 17, 18, 20]. The latter method is more computationally complex because the parameters can be updated instantaneously. In the least-squares method, the speech or error signal is multiplied by a window and the set of coefficients $\{a_k\}$ are chosen to minimize the energy of the error signal. The first of the two least-square techniques is the *autocorrelation* method. In this method, the speech signal is multiplied by a window whereas in the alternative least squares technique, the *covariance* method, the error signal is windowed instead. The autocorrelation method guarantees that the resulting LP analysis filter $A(z)$ is minimum phase, which means that the all-pole synthesis filter $H(z)$ is always stable. This property makes the autocorrelation method the most popular technique for the estimation of the filter coefficients.

2.2.2 Long-term prediction

Voiced speech segments show strong long-term correlation and is maintained in the LP residual signal. These far-sample redundancies can further be exploited by the use of a *pitch predictor*. For this purpose, a single tap filter can be employed, of the following form

$$P(z) = \beta z^{-D}, \tag{2.8}$$

where β is the predictor coefficient and D is the estimated pitch period (delay) in samples. The error signal is expressed as

$$e(n) = r(n) - \beta r(n - D), \tag{2.9}$$

and is called the *pitch residual signal.* Taking the z-transform on both sides and rearranging the terms, the resulting *pitch analysis filter* is expressed as

$$P_a(z) = 1 - \beta z^{-D}, \tag{2.10}$$

In time domain, the pitch analysis filter subtracts the speech sample (weighted by β) at a delay equal to the estimated period from the current sample. In frequency domain, the pitch analysis filter removes the harmonic structure from the input signal and in this case the LP residual. Pitch analysis will have no useful effect during unvoiced speech since unvoiced excitation is random (no harmonic structure). The predictor coefficient relates to the degree of waveform periodicity and takes on values in the range $0 \leq \beta < 1$. Thus β is near 0 for a signal with no detectable periodic structure (and in which case the value of D is irrelevant) and close to unity for steady-state voiced speech.

At the decoder, the *pitch synthesis filter* is given as

$$P_s(z) = \frac{1}{P_a(z)} = \frac{1}{1 - \beta z^{-D}}, \tag{2.11}$$

and is used to introduce a harmonic structure to the synthesized speech signal.

A single tap predictor is not sufficient to provide an exact representation of the periodicity of the signal because the true pitch period is unlikely to be an exact multiple of the sampling period. Whereas, higher order predictors allow interpolation of speech samples in the delayed version of the signal, to more precisely match the original. Atal [21], has considered a three-tap predictor of the form

$$P(z) = \sum_{k=-1}^{1} \beta_k z^{-D+k}. \tag{2.12}$$

2.2.3 Estimation of the predictor parameters

A general formulation for determining the predictor coefficients for both formant and pitch predictors in traversal form was presented in [22].

Based on the model shown in Fig. 2.4, the windowed error signal $e_w[n]$ is given as

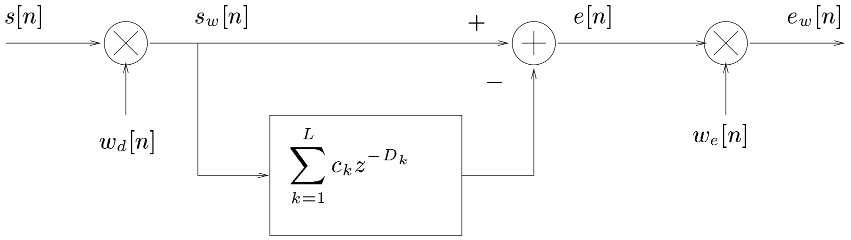


Fig. 2.4 Analysis model for transversal predictors.

$$\begin{aligned} e_w[n] &= w_e[n]e[n] \\ &= w_e[n]s_w[n] - w_e[n]\sum_{k=1}^{L} c_k s_w[n-D_k], \end{aligned} \tag{2.13}$$

where $s[n]$ is the input signal and $w_d[n]$, $w_e[n]$ are the data and error windows respectively. The values of D_k are arbitrary but distinct integers corresponding to delays of the weighted input signal $s_w[n]$.

The energy of the error, or otherwise mean-squared error (MSE), is given as

$$\epsilon = \sum_{n=-\infty}^{\infty} e_w^2[n]. \tag{2.14}$$

The coefficients c_k are computed by minimizing ϵ. This is done by taking the partial derivative of Eq. (2.14) with respect to each of the coefficients c_k, for $k = 1, \ldots, L$, and setting each of the resulting L equations to zero. This leads to a linear system of equations that can be written in a matrix form ($\mathbf{\Phi c} = \mathbf{a}$):

$$\begin{bmatrix} \phi(D_1,D_1) & \phi(D_1,D_2) & \cdots & \phi(D_1,D_L) \\ \phi(D_2,D_1) & \phi(D_2,D_2) & \cdots & \phi(D_2,D_L) \\ \vdots & \vdots & & \vdots \\ \phi(D_L,D_1) & \phi(D_L,D_2) & \cdots & \phi(D_L,D_L) \end{bmatrix} \begin{bmatrix} c_1 \\ c_2 \\ \vdots \\ c_L \end{bmatrix} = \begin{bmatrix} \phi(0,D_1) \\ \phi(0,D_2) \\ \vdots \\ \phi(0,D_L) \end{bmatrix} \tag{2.15}$$

where

$$\phi(i,j) = \sum_{n=-\infty}^{\infty} w_e^2[n]s_w[n-i]s_w[n-j]. \tag{2.16}$$

The matrix $\mathbf{\Phi}$ is always symmetric and positive definite. It is also Toeplitz if the intercoefficient delays are equal. Depending on whether $\mathbf{\Phi}$ is Toeplitz or not, either the Levinson recursion or the Cholesky decomposition can be used to solve the system of equations.

For a formant predictor $D_k = k$ for $k = 1, \ldots, p$ and for a pitch predictor of order N_p, $D_k = D + k$ for $k = 0, \ldots, N_p - 1$. When $w_e[n] = 1 \quad \forall n$, the above formulation results in the autocorrelation method. The covariance method results if $w_d[n] = 1 \quad \forall n$. For formant prediction, the autocorrelation method method can be shown to give a minimum phase LP analysis filter [23], whereas the LP synthesis filter resulting from the covariance method might be unstable. In the case of pitch prediction, both the autocorrelation (except the case of a single tap pitch predictor) and covariance methods might result in unstable pitch synthesis filters. However, mild instability is often useful to model increasing amplitudes in the excitation signal. As the covariance method gives high prediction gains is thus preferred for pitch prediction. Whenever the pitch synthesis filter is found unstable, efficient stabilization schemes can be employed to limit the instability to desirable limits [22].

So far we have assumed that the pitch period D is known. Often a single tap pitch filter is used and D is determined separately from the predictor coefficient using the covariance method. This procedure avoids an exhaustive search for the optimal D in multi-tap pitch filters [24]. Using the covariance method, the mean-squared error in Eq. (2.14) can be rewritten in matrix form as $\epsilon = \phi(0,0) - 2\mathbf{c^T a} + \mathbf{c^T \Phi c}$ and given that the optimum coefficients are given by $\mathbf{c} = \mathbf{\Phi^{-1} a}$, the resulting mean-squared error is $\epsilon = \phi(0,0) - \mathbf{c^T a}$.

For a single tap pitch predictor $N_p = 1$ and Eq. (2.15) is simplified to

$$\phi(D,D)c_1 = \phi(0,D) \tag{2.17}$$

and the optimal coefficient β_{opt} is given as

$$\beta_{opt} = c_1 = \frac{\phi(0,D)}{\phi(D,D)}. \tag{2.18}$$

Thus, the resulting mean-squared error for a single tap pitch filter reduces to

$$\epsilon = \phi(0,0) - \frac{\phi^2(0,D)}{\phi(D,D)}. \tag{2.19}$$

The resulting MSE in Eq. (2.19) is minimized by maximizing $\phi^2(0,D)/\phi(D,D)$. This function is computed over all possible values of D, and its maximum indicates the best choice for the pitch period D. The range of values over which the pitch period is searched is typically between 20 and 143 samples (for speech sampled at 8 kHz) which covers most pitch values encountered in human speech. Other practical methods for choosing D can be found in [24, 25].

2.3 Analysis by Synthesis Principle

An important class of coders that use linear prediction is *linear prediction analysis by synthesis* (LPAS) coders. A conceptual diagram of a generic LPAS coder is shown in Fig. 2.5 [26].

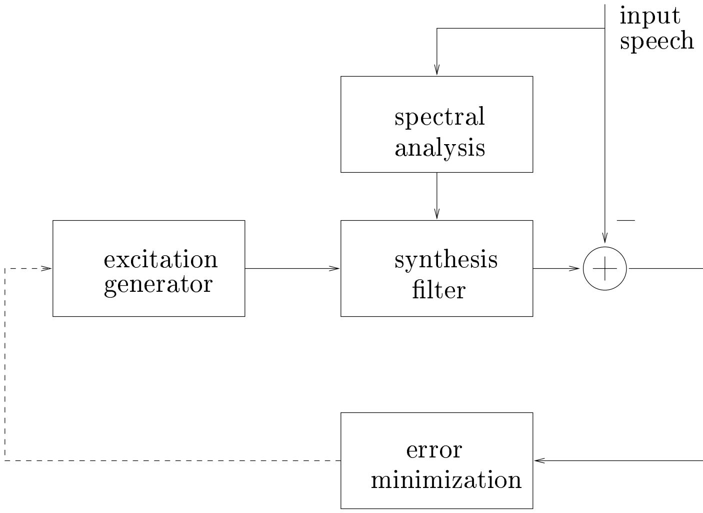


Fig. 2.5 Generic LPAS coder

In LPAS coding, the input speech signal is analyzed and the excitation signal is determined one segment at a time. Typically, LP analysis is carried out every frame (which might vary from 80 to 240 samples for speech signals sampled at 8 kHz) and the excitation signal is determined every subframe (usually 40 to 80 samples long). In this figure, an excitation signal is filtered through the synthesis filter to produce the reconstructed speech signal. The input speech is then subtracted form the reconstructed signal and the resulting quantization error is minimized using the mean-squared error criterion. The excitation signal that minimizes the energy of the quantization error is selected and the parameters corresponding to this signal are transmitted to the receiver. The receiver uses the same synthesis structure to reconstruct the original signal. Since the quantization procedure at the encoder requires the calculation of the reconstructed speech for a number of excitation signals, this type of coding is called *analysis-by-synthesis* coding.

A major reason for the success of LPAS coding is the fact that it is easy to incorporate into its structure knowledge about human auditory perception and in particular, exploit auditory *spectral masking*. This is achieved by the use of perceptual frequency weighting on the error signal during the selection of the best excitation signal as shown in Fig. 2.6. The coder parameters are now selected on the basis of the perceptually frequency-weighted mean-squared error criterion.

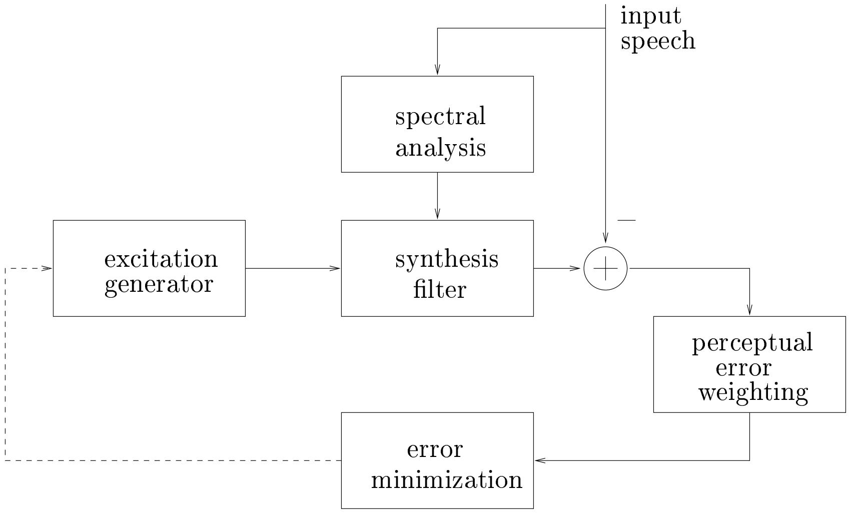


Fig. 2.6 LPAS coder with error weighting

In Fig. 2.6 the error signal is passed through a time-varying weighting filter which emphasizes the error in frequency regions where the input speech has valleys (spectral local minima) and deemphasizes the error near formants (spectral peaks). This is done by using an IIR filter $W(z)$, whose rational transfer function is obtained from the LP analysis filter and scaling down the magnitude of the poles and zeros by a constant factor (bandwidth expansion). The transfer function of a typical weighting filter is given as

$$W(z) = \frac{A(z/\gamma_1)}{A(z/\gamma_2)}, \qquad 0 < \gamma_2 < \gamma_1 \leq 1. \tag{2.20}$$

With γ_1 and γ_2 as in Eq. (2.20), the response of $W(z)$ has valleys at the corresponding formant regions of $A(z)$ and the regions between the formants are emphasized. This technique is justified by the masking feature in the human auditory system: the audibility of low energy noise is reduced near higher energy frequency regions of the speech signal.

The method for determining the excitation forms the main difference between many LPAS systems; for example multi-pulse excitation (MPE) coding [27] and regular-pulse excitation (RPE) coding [28] which provide good quality speech with reasonable complexity at rates around 10 kbps. In multi-pulse coding, the excitation waveform is modeled as pulses which can take on arbitrary amplitudes and can be located in arbitrary positions. The excitation waveform is obtained by optimizing the positions and amplitudes of a fixed number of pulses within an excitation frame, to minimize the frequency weighted mean-squared error. RPE coding uses the same idea but the spacing between the pulses is fixed, so that only the position of the first pulse and the amplitudes of the pulses are optimized. The pulse excitation in these two techniques is used to model voiced, unvoiced and mixed excitation signals, by optimizing the amplitudes and positions of the available pulses.

Further gains in coding efficiency are obtained by incorporating vector quantization techniques [29, 30]. Vector quantization techniques are exploited in CELP coders. The major difference between the CELP coding algorithm and the aforementioned multi-pulse coding, is the incorporation of a pitch synthesis filter and the replacement of the pulse excitation with a *codebook excitation*. The pitch synthesis filter obviates the need for the excitation waveform to contain pitch pulse information. As originally described by Atal and Schroeder [1], the codebook consists of a fixed dictionary of randomly generated Gaussian excitation waveforms. Such waveforms can very well model unvoiced excitation and the pitch synthesis filter provides the desired periodicity to synthesized speech.

2.4 The Code-Excited Linear Prediction (CELP) Algorithm

After formant and pitch prediction, the resulting residual is very noise-like and with appropriate gain normalization has a distribution which is nearly Gaussian. It is the noise-like nature of the residual after the two stages of prediction that motivates the use of a dictionary (or codebook) of randomly generated Gaussian waveforms. Due to their random nature, these waveforms are also called *stochastic* and the dictionary of waveforms is commonly referred to as *stochastic codebook*. CELP coders do not transmit the residual signal after prediction. Instead, an index to a waveform stored in the stochastic codebook along with a gain factor is sent. In a sense, the difference between the residual signal after formant and pitch prediction and the chosen excitation waveform is the quantization error.

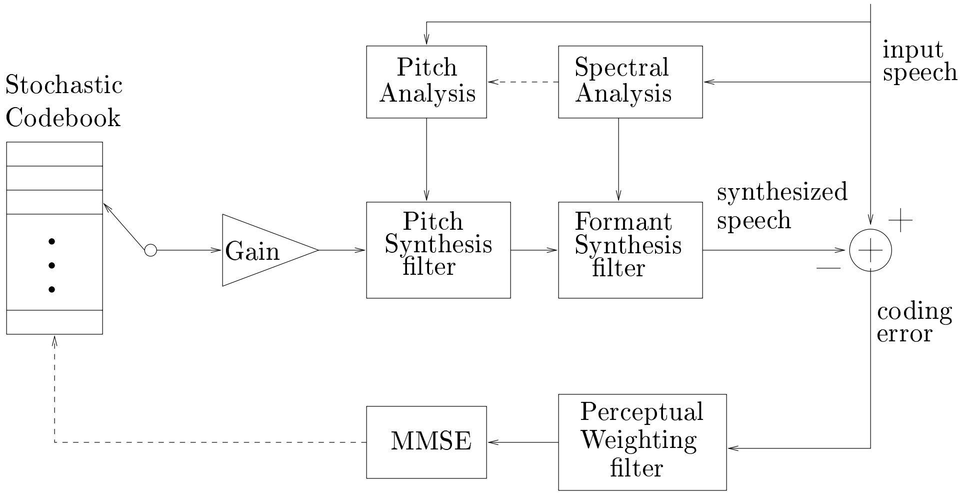


Fig. 2.7 Generic model for the CELP encoder.

Since the excitation signal is chosen block by block, the selection of the best waveform can be viewed as quantization of a *vector* of samples. At the receiving end, the decoder uses this index and gain to excite the synthesis filters and reconstruct the speech signal.

2.4.1 CELP encoder/decoder

The generic model for the CELP encoder is shown in Fig. 2.7. Conceptually, each waveform in the stochastic codebook is scaled and passed through the synthesis filters to determine which waveform produces synthesized speech that best matches the input speech (based on the weighted mean-squared error criterion). The index of the best waveform is transmitted to the receiver, along with the parameters for the synthesis filters.

The coefficients for the formant synthesis filters are typically found by analyzing the input speech. Conventionally, the pitch synthesis filter is determined by analyzing the input speech or the LP residual, but it can also be chosen (under certain constraints) to optimize the synthesized speech signal. For the purposes of the thesis, the process of selecting parameters to optimize the synthesized speech will be referred to as *closed-loop* search. In other words, during closed-loop search a range of values of a parameter are used to re-synthesize the original signal and the one that results in the most accurate reconstruction is selected. However, the process of determining parameters by directly analyzing the input speech will be referred to as *open-loop* estimation. The open-loop determination of the analysis, and hence the synthesis filter parameters is done using the methods described in the previous sections.

The synthesis filters are updated at regular intervals. The formant synthesis filter is updated once per frame of samples, while the gain, excitation selection and pitch filter parameters are updated at the subframe level. This information is also transmitted to the decoder to allow reconstruction of the speech signal.

The decoder has a copy of the codebook and uses the index and gain parameters to construct the excitation signal. The constructed excitation is then used to excite the pitch and formant synthesis filters which are formed by using the pitch period and pitch coefficient(s), and formant synthesis filter coefficients respectively.

2.4.2 Selecting the synthesis parameters

For the purpose of this thesis, the formant synthesis filter parameters are derived by analyzing the input speech. This filter is specified by p coefficients which are updated once per frame.

The remaining parameters to be selected consist of the input waveform and the pitch filter parameters; that is the waveform index i, the gain factor G, the pitch period D and the pitch filter coefficient β. The optimization procedure is based on the frequency weighted error criterion as shown in Fig. 2.8.

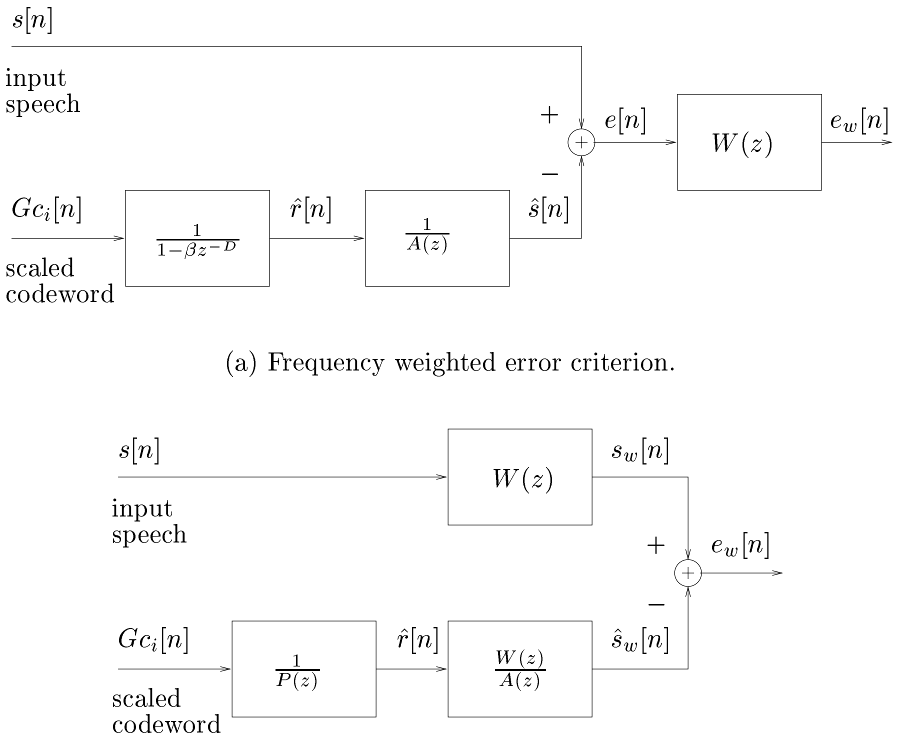
The parameter selection process involves computing the weighted squared error,

$$\epsilon = \sum_{n=0}^{N-1} \left(s_w[n] - \hat{s}_w[n]\right)^2, \tag{2.21}$$

where N is the subframe size, and minimizing it over the choice of the synthesis parameters.

Various methods exist for the determination of the above parameters, with varying computational complexity and performance. Moncet and Kabal in [31] have studied and proposed methods for optimizing these parameters and some of their results are briefly summarized here.

- An optimum scheme finds a jointly optimal set of values for the synthesis parameters i, G, β and D. In this scheme G and β are determined for all combinations of the pair (i, D) and the values that minimize the weighted squared error are selected. The solution for the optimum gain factor and filter coefficient is linear in the parameters if the pitch period is at least as large as the subframe size, i.e., $D \geq N$. This scheme



(a) Frequency weighted error criterion.

(b) Alternative model.

Fig. 2.8 Models for the calculation of the frequency error criterion during the selection of the synthesis parameters.

was compared with the method of determining the pitch parameters D and β by analyzing the input speech and its was found that the optimum scheme gives higher weighted SNR and the harmonic structure was better reproduced.

- The optimum scheme is computationally burdensome even for a small number of waveforms. Instead, a sequential approach is suggested in which the pitch period is optimized for a zero input waveform, i.e., $G = 0$. Once D is found, it is kept fixed and the rest of the parameters are determined in two ways:

  1. The optimum gain and pitch coefficient are found for each value of i.
  2. The pitch coefficient is also determined for zero excitation. Keeping the pitch filter fixed, the optimum gain is found for each waveform index.

Here, the first variant can be viewed as the being same as the second variant, but with the pitch coefficient re-optimized for each waveform. Comparing the two variants, it was concluded that the first variant may be the method of choice.

The limitation that the pitch period be greater than the subframe size causes some problems for high pitched speech. With the above schemes, when the pitch period is smaller than the subframe size the pitch filter is forced to operate around multiples of the fundamental period instead. Pitch doubling makes the harmonic structure of the signal vary and subjectively it introduces some wavering in the reconstructed speech. The formulation for the optimization of the gain and pitch coefficient results in a nonlinear set of equations which is impractical to solve. Specifically, for the case of a single tap pitch filter and with $G = 0$ and $N/2 \leq D < N$, the optimum value of β can be found in two ways:

1. The excitation signal for the current subframe in this case is given as

$$\hat{r}[n] = \begin{cases} \beta \hat{r}[n-D] & 0 \leq n < D \\ \beta^2 \hat{r}[n-2D] & D \leq n < N \ , \end{cases} \tag{2.22}$$

   and the optimal value of β is found by solving a cubic equation.

2. Using a more empirical approach, the past pitch filter output is periodically continued,

$$\hat{r}[n] = \begin{cases} \beta \hat{r}[n-D] & 0 \leq n < D \\ \beta \hat{r}[n-2D] & D \leq n < N \ . \end{cases} \tag{2.23}$$

   With this formulation, the equation for β is linear. Subjective comparisons showed that this method gives slightly poorer results than the first approach [31].

In conclusion, the sequential approach to choosing the pitch filter parameters is computationally attractive. With this approach one can view the excitation signal, used to excite the formant synthesis filter, to consist of two components. The first is a scaled and delayed version of the previous excitation signal and is generated by the pitch synthesis filter. During voiced speech, this supplies the pitch component. The gain scaled waveform selected form the dictionary fills in the details that are missing in the excitation signal. This also supplies the startup component for the pitch excitation during transitions from silence or unvoiced to voiced.

2.4.3 Fixed excitation codebooks

The unstructured character of the stochastic codebook initially proposed for CELP, is not amenable to efficient search methods. In order to reduce the excessive computational load to search for the optimum excitation vector, a variety of structural constraints on the excitation codebook has been proposed. The goal is to achieve one or more of the following features:

- Fast search procedures
- Reduced storage space
- Reduced sensitivity to channel errors
- Increased speech quality

Some of the key innovations are *overlapped, sparse, ternary, lattice, algebraic* and *trained* codebooks. The reader is referred to [32] for a brief description of all the above.

Henceforth, we will refer to all the above structural and non-structural codebook designs as *fixed codebooks.* A fixed codebook can generally be described as:

(a) Stochastic or,

(b) Deterministic.

The above distinction arises from the way the codevectors are chosen, for example:

- Noise-like signals, characterized by white Gaussian random variables, or
- Speech-derived signals, determined iteratively during the closed-loop search.

Chapter 3

Modeling Periodicity in CELP coders

3.1 Adaptive Codebook Paradigm

As was discussed in Section 2.4.2, the pitch synthesis filter models the long-term correlations of the speech signal by repeating scaled segments of the previously constructed excitation, alternatively referred to as past excitation. The past excitation is defined here as the excitation constructed before the current subframe.

Ideally, the past excitation is delayed by an interval which equals the pitch period as shown in Fig. 3.1.

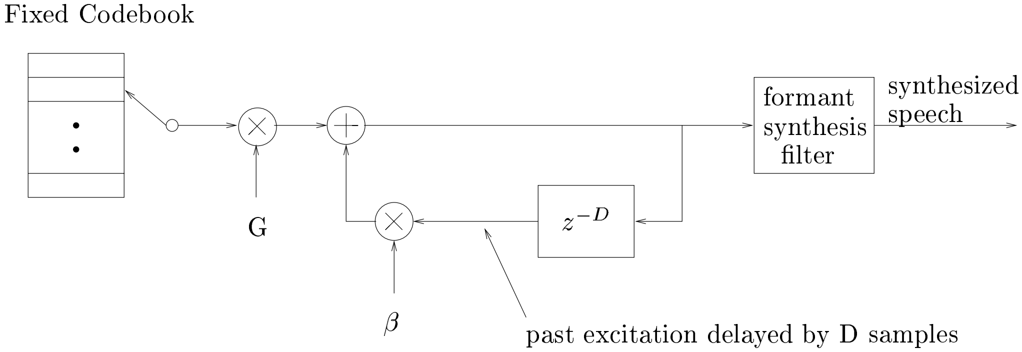


Fig. 3.1 Pitch Synthesis in CELP coders viewed as a *filtering operation*.

This operation can also be viewed as selecting a vector of samples from an *adaptive codebook* [33], as illustrated in Fig. 3.2. In this interpretation, the excitation is considered to be a linear combination of an adaptive codebook and a fixed codebook contribution.

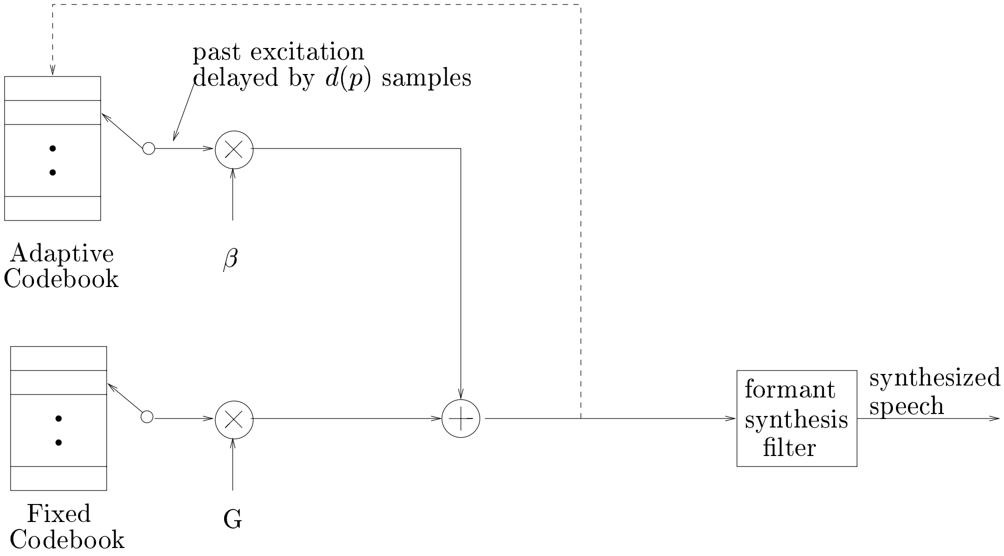


Fig. 3.2 CELP synthesis with an *adaptive codebook*.

3.1.1 Definition and realization

The adaptive codebook stores vectors of samples from past excitation, each representing a segment of the past excitation at a specific delay. The adaptive codebook vector $\upsilon[n]$, $n = 0, \ldots, N-1$, is constructed from the excitation signal $\hat{r}[n]$ as follows. Let the current subframe start at $n = 0$ and contain N samples. First set

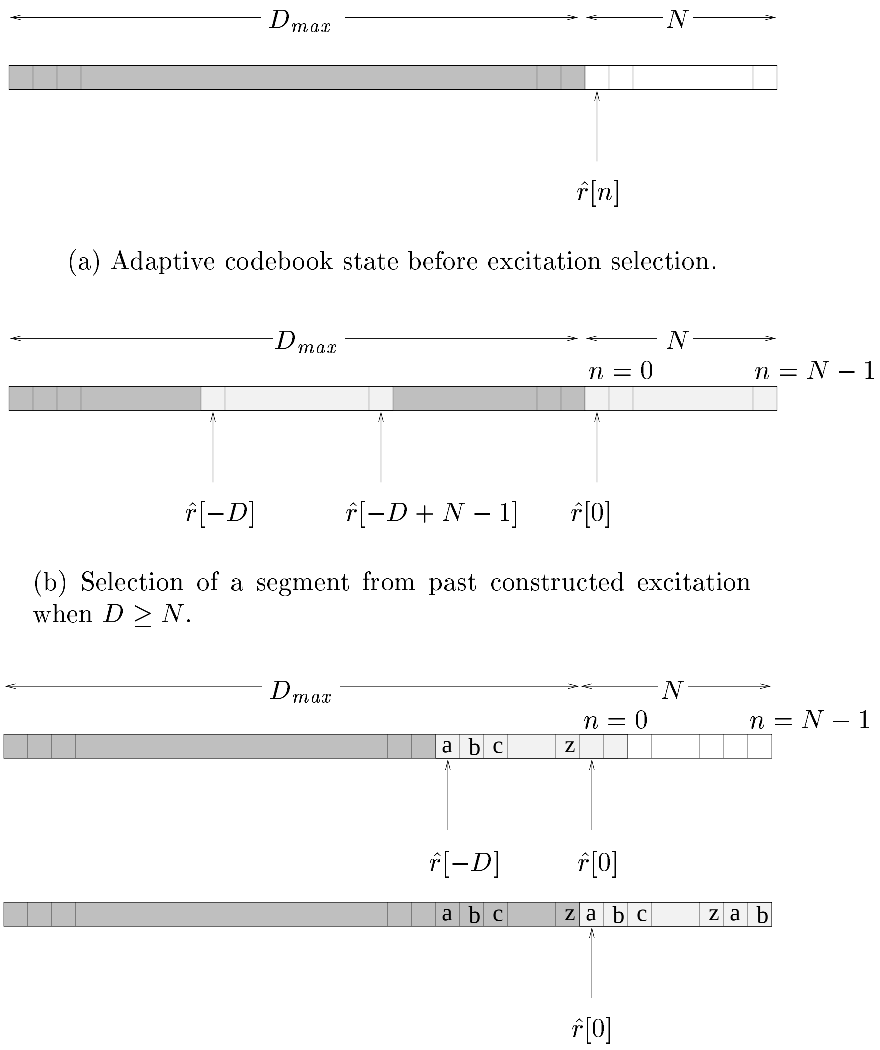
$$\upsilon[n] = \hat{r}[n] \qquad \text{for } -D_{max} \le n < 0,$$

where D_{max} is the maximum allowable adaptive codebook delay (pitch period), typically 143 samples for 8 kHz sampled speech. Then construct $\upsilon[n]$ for $n \ge 0$ by recursive copying:

$$\upsilon[n] = \upsilon[n - d(p)] \qquad \text{for } n = 0, 1, \ldots, N-1, \tag{3.1}$$

where $d(p)$ is the delay for the adaptive codebook entry with index p. The data sequence $\upsilon[n]$, $n = 0, \ldots, N-1$, forms the adaptive codebook entry for delay $d(p)$.

In practice, an array of samples is specified, of length at least as large as $D_{max} + N$. The first D_{max} samples represent past constructed excitation and the next N samples represent



(a) Adaptive codebook state before excitation selection.

(b) Selection of a segment from past constructed excitation when $D \geq N$.

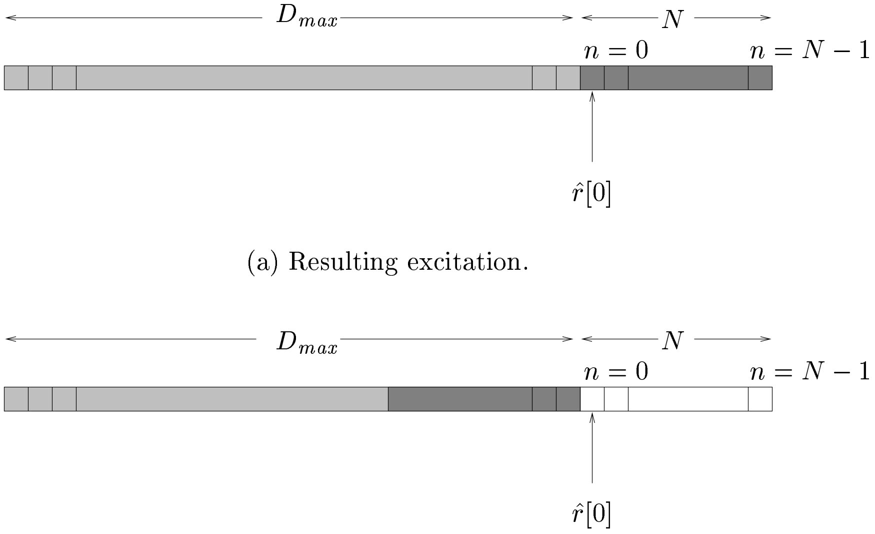
(c) Selection of a segment from past constructed excitation when $D = N - 2$.

Fig. 3.3 An alternative view of the adaptive codebook selection procedure.

the excitation for the current subframe, as illustrated in Figs. 3.3 and 3.4.

As shown in Fig. 3.4, the adaptive codebook is updated every subframe by shifting its contents to the left by N samples. The shifting is done after the resulting excitation is copied to the current subframe. The resulting excitation is defined here as the sum of the scaled adaptive codebook contribution and scaled fixed codebook contribution.

The adaptive codebook procedure is similar to the closed loop pitch prediction (filtering



(a) Resulting excitation.

(b) Adaptive codebook update.

Fig. 3.4 Adaptive codebook update procedure.

approach), if the pitch period is greater than a subframe length. In Section 2.4.2 we noted that a computational problem occurs when the pitch period is smaller than the subframe length. There are several alternatives as how to handle this problem:

1. In the filtering approach the samples are calculated recursively.
2. In the adaptive codebook procedure this case is treated differently [33]. The difference is seen in Eq. (3.1) and illustrated in Fig. 3.3(c), where the sequence representing the adaptive codebook contribution (without scaling) repeats itself when the delay $d(p)$ is less than the subframe length N.

Essentially, the adaptive codebook procedure differs from the filtering approach in the relative scaling of the repeated waveform. That is, in the adaptive codebook procedure the repeated samples are given the same weight β as the rest of the samples in the sequence, whereas in the filtering approach the corresponding samples are scaled by β^2.

3.1.2 Determining the excitation

In this section we describe practical procedures for selecting a good excitation for the formant synthesis filter. First we provide an analytical method to find the optimum adaptive codebook vector and gain. This method is similar to the one described in Section 2.2.3, but we derive the formulae once again by considering that the resulting excitation is now constructed from an adaptive and a fixed codebook contribution. In subsequent sections we describe methods that are being used in latest technology coders.

For convenience, we describe the signals within a subframe as a vector. The weighted speech vector is denoted as $\mathbf{s_w} \equiv [s[0], \ldots, s[N-1]]^T$ and the resulting excitation vector as $\hat{\mathbf{r}} \equiv [\hat{r}[0], \ldots, \hat{r}[N-1]]^T$. An adaptive codebook vector with index p from a codebook of N-dimensional vectors is denoted as $\boldsymbol{\upsilon}_p$, where

$$\upsilon_p[n] = \hat{r}[n-d(p)] = \hat{r}[n-D], \qquad n = 0, \ldots, N-1, \tag{3.2}$$

and let β be the associated scaling factor. Furthermore, let $\mathbf{c}_i$ and G be the fixed codebook vector with index i from a codebook of N-dimensional vectors and associated scaling factor respectively. Then the resulting excitation is given as

$$\hat{\mathbf{r}}_{ip} = \beta \boldsymbol{\upsilon}_p + G\mathbf{c}_i. \tag{3.3}$$

The selected codebook entries and scaling factors are those that result in the smallest frequency-weighted difference with the original speech signal on filtering with the formant synthesis filter. The adaptive and fixed codebooks have to be searched simultaneously for optimal performance. However, in order to reduce the amount of computation involved, the search is often done sequentially with the adaptive codebook being search first. This choice is justified by the fact that the adaptive codebook normally provides the largest contribution to the resulting excitation in voiced speech. Finally, the resulting optimal excitation vector is expressed as

$$\hat{\mathbf{r}}_{opt} = \beta_{opt} \boldsymbol{\upsilon}_{opt} + G_{opt}\mathbf{c}_{opt}. \tag{3.4}$$

To select an entry from the adaptive codebook, each vector $\boldsymbol{\upsilon}_p$ must be filtered with the weighted formant synthesis filter of Fig. 2.8 using the same filter memory (i.e. same zero-input response) and evaluate their reconstruction performance. Let $\hat{\mathbf{s}}_p$ be the synthesized weighted speech vector for adaptive codebook entry p and $\hat{\mathbf{s}}_0$ be the zero-input response of the weighted formant synthesis filter. Then the filtering operation can be represented as

$$\hat{\mathbf{s}}_p = \beta\,\mathbf{H}\boldsymbol{v}_p + \hat{\mathbf{s}}_0. \tag{3.5}$$

where

$$\mathbf{H} = \begin{bmatrix} h_0 & 0 & 0 & \cdots & 0 \\ h_1 & h_0 & 0 & \cdots & 0 \\ h_2 & h_1 & h_0 & \cdots & 0 \\ \vdots & \vdots & \vdots & & \vdots \\ h_{N-2} & h_{N-3} & h_{N-4} & \cdots & 0 \\ h_{N-1} & h_{N-2} & h_{N-3} & \cdots & h_0 \end{bmatrix}$$

and $\{h_0, h_1, h_2, \ldots, h_{N-1}\}$ is the impulse response of the weighted formant synthesis filter. The mean-squared difference between the weighted original speech vector and the weighted synthesized speech vector is to be minimized. The error criterion is written as

$$\begin{aligned} \epsilon_0 &= |\mathbf{s_w} - \hat{\mathbf{s}}_p|^2 \\ &= \beta^2 {\boldsymbol{v}_p}^T \mathbf{H}^T \mathbf{H} \boldsymbol{v}_p - 2\beta {\boldsymbol{v}_p}^T \mathbf{H}^T (\mathbf{s_w} - \hat{\mathbf{s}}_0) + |\mathbf{s_w} - \hat{\mathbf{s}}_0|^2, \end{aligned} \tag{3.6}$$

where the last term is constant during the search. The optimal gain for a particular codebook entry can be determined by minimizing ϵ_0 with respect to β. This is done by differentiating ϵ_0 with respect to β, setting the resulting equation to zero and finally solving for β to give:

$$\beta_{opt} = \frac{{\boldsymbol{v}_p}^T \mathbf{H}^T (\mathbf{s_w} - \hat{\mathbf{s}}_0)}{{\boldsymbol{v}_p}^T \mathbf{H}^T \mathbf{H} \boldsymbol{v}_p}. \tag{3.7}$$

Substituting the optimal value of β in Eq. (3.6) and omitting the constant term, since it does not depend on any of the parameters to be optimized, the error criterion is reduced to

$$\epsilon_1 = -\frac{\left[{\boldsymbol{v}_p}^T \mathbf{H}^T (\mathbf{s_w} - \hat{\mathbf{s}}_0)\right]^2}{{\boldsymbol{v}_p}^T \mathbf{H}^T \mathbf{H} \boldsymbol{v}_p}. \tag{3.8}$$

The vector $\mathbf{s_w} - \hat{\mathbf{s}}_0$ is usually referred to as the *target* vector. The value of ϵ_1 is computed for each codebook entry. At this point the reader is reminded that an adaptive codebook entry here implies a certain delay value. Thus, the delay that results in the smallest value of ϵ_1

is selected. The fixed codebook is searched in an identical manner. The only difference is that the contribution of the adaptive codebook vector has to be taken in to account. This is done by replacing $\hat{\mathbf{s}}_0$ in Eqs. (3.6), (3.7) and (3.8) by $\hat{\mathbf{s}}_p$ of Eq. (3.5).

Equations (3.7) and (3.8) can be re-written in sample domain as follows. Let $\mathbf{s}_t$ denote the target vector. Then the numerator of Eq. (3.7) can as well be written as $\boldsymbol{v}_p{}^T\mathbf{H}^T\mathbf{s}_t$ or as $(\mathbf{H}\boldsymbol{v}_p)^T\mathbf{s}_t$. But $\mathbf{H}\boldsymbol{v}_p$ is the the synthesized weighted speech vector for adaptive codebook entry p (the filtered adaptive codebook vector), denoted as $\hat{\mathbf{s}}_p$. Thus, we have

$$\beta_{opt} = \frac{(\mathbf{H}\boldsymbol{v}_p)^T\mathbf{s}_t}{(\mathbf{H}\boldsymbol{v}_p)^T\mathbf{H}\boldsymbol{v}_p} = \frac{\sum\limits_{n=0}^{N-1} s_t[n]\hat{s}_p[n]}{\sum\limits_{n=0}^{N-1} \hat{s}_p[n]\hat{s}_p[n]}, \tag{3.9}$$

and

$$\epsilon_1 = -\frac{\left[(\mathbf{H}\boldsymbol{v}_p)^T\mathbf{s}_t\right]^2}{(\mathbf{H}\boldsymbol{v}_p)^T\mathbf{H}\boldsymbol{v}_p} = -\frac{\left[\sum\limits_{n=0}^{N-1} s_t[n]\hat{s}_p[n]\right]^2}{\sum\limits_{n=0}^{N-1} \hat{s}_p[n]\hat{s}_p[n]}. \tag{3.10}$$

3.1.3 Use of fractional delays in pitch prediction

So far, the pitch delay (period) in adaptive codebook search was restricted to integer multiples of the sampling interval. This restriction affects the performance of the coder especially in high pitched sounds. Higher prediction gain can be achieved by using fractional values for the delay [34, 35] (additional information on fractional delays and their use in other applications can be found in [36]).

Trying to generalize the equation of the adaptive codebook contribution to the resulting excitation, expressed as

$$\hat{r}[n] = \beta\hat{r}[n-D], \qquad n = 0, \ldots, N-1, \tag{3.11}$$

the plain substitution of the integer delay D by a real number is not possible. This is because the discrete-time signal $\hat{r}[n]$ is not defined for non-integer values of the argument. This difficulty is overcomed by proceeding as follows [34] .

A fractional delay can be expressed as an integer delay T at sampling rate f_s and a fraction $t/I, \ t = 0, 1, \ldots, I-1$, where I is the resolution of the fraction, specified as a multiple of the original sampling rate f_s of the input signal $\hat{r}[n]$. A non-integer delay t/I at the original sampling rate f_s corresponds to an integer delay t at a rate If_s. Thus, a fractional delay of t/I samples can be implemented by increasing the sampling rate of $\hat{r}[n]$ by a factor of I (by inserting $I-1$ zero-valued samples between each sample of $\hat{r}[n]$) and then lowpass filter it to obtain an interpolated version. The interpolated signal is delayed by t samples and the delayed output is down-sampled by a factor of I (by selecting every I-th sample). The resulting signal, whose sampling frequency is again f_s, is the original signal delayed by the non-integer delay t/I. It should be noted however that a constant integer delay is introduced due to the delay of the lowpass interpolation filter.

The interpolation filter $b[n], \ n = 0, 1, \ldots, M-1$ is chosen to be an FIR filter with linear phase. It is designed to attenuate the aliasing components due to the down-sampling process. The number of coefficients is chosen such that its delay at the high sampling rate, $(M-1)/2$, is an integer multiple of I, or equivalently

$$M = 2\tau I + 1, \tag{3.12}$$

where τ is the integer delay introduced by the interpolation filter at the lower sampling rate.

Fractional delays can be implemented efficiently with a polyphase structure [37, 38]. The polyphase subfilters $p_t[n]$ are obtained from the coefficients of the lowpass filter $b[n]$ according to

$$\begin{aligned} p_t[n] = b[nI + t], \qquad & \text{for } t = 0, 1, \ldots, I-1 \\ & \text{and } n = 0, 1, \ldots, K-1 \ , \end{aligned} \tag{3.13}$$

and such that $b[m] = 0$ for $m > M-1$. The parameter K is the number of coefficients of the polyphase subfilter and is given by

$$K = \left\lceil \frac{M}{I} \right\rceil. \tag{3.14}$$

Conceptually, the t-th subfilter $p_t[n]$ delays the input signal by t/I samples, for $t = 1, \ldots, I-1$ and the zero-th subfilter $p_0[n]$ generates no delay. Therefore, the output signal

$\hat{r}[n]$ for a fractional delay of t/I samples is given as

$$\hat{r}[n] = \beta \sum_{m=0}^{K-1} p_t[m]\hat{r}[n-m], \qquad n = 0, \ldots, N-1. \tag{3.15}$$

When the overall delay consists of an integer part T and a fraction t/I, and taking into account the delay τ of the lowpass filter, the adaptive codebook contribution at delay T and fraction t is expressed as

$$\hat{r}[n] = \beta \sum_{m=0}^{K-1} p_t[m]\hat{r}[n-T+\tau-m]. \tag{3.16}$$

The above formulation of the interpolation operation implies a non-causal filtering operation where τ future samples must be known. This raises a problem in the closed-loop procedure for the determination of the optimum delay, because knowledge of the input in the present subframe may be necessary for short delays. The same problem arises for the case of an integer delay. This can be seen in Eq. (3.2) where for delays less than the subframe length, $\boldsymbol{v}_p$ is not defined for $n = D$ or $T, \ldots, N-1$. This difficulty is overcomed by either repeating the excitation as stated earlier in Section 3.1.1 or by extending the excitation with the LP residual of the current subframe. The latter method cannot be used at the decoder since the LP residual is not available. Thus, this method can only used in the search stage of the encoder to simplify the closed-loop search [5].

3.1.4 Practical approaches to the adaptive codebook search

The closed-loop analysis for determining the best fractional delay requires evaluation of every vector $\boldsymbol{v}_p$ in Eq. (3.10) for all fractional delays in the specified range, filtering it with the weighted synthesis filter (matrix $\mathbf{H}$) and computing the resulting error ϵ_1. This procedure forms a large part of the overall complexity of the coder and hence a variety of techniques have been proposed to reduce its complexity.

The most common technique is based on reducing the number of delays that have to be examined. This is done by making the search a two-stage procedure. First an open-loop estimate of the pitch period, using integer values only, is used to narrow down the range of possible delay values on which a fine search will be done. This limited range of delay values is then used by a closed-loop procedure to find the high-resolution delay using fractions [5].

In the following subsections, we briefly describe the two-stage procedure used by the ITU-T G.729 standard, to estimate the pitch period.

Open-loop pitch analysis

The complexity of the closed-loop search for the adaptive codebook delay is reduced by limiting the search range around a candidate delay T_{op}, obtained from an open loop pitch analysis. The open-loop pitch estimation uses the weighted speech signal $s_w[n]$ and is done once per frame (10 ms or 80 samples for 8 kHz sampled speech). First, 3 maxima of the correlation

$$R[k] = \sum_{n=0}^{79} s_w[n]s_w[n-k] \tag{3.17}$$

are found in the following three ranges of delay values:

$$\begin{aligned} i = 1: &\qquad 80, \ldots, 143, \\ i = 2: &\qquad 40, \ldots, 79, \\ i = 3: &\qquad 20, \ldots, 39. \end{aligned}$$

The retained maxima $R[d_i],\;\; i = 1, \ldots, 3$, are normalized through

$$R'[d_i] = \frac{R[d_i]}{\sqrt{\sum_n {s_w}^2[n-d_i]}} \qquad\qquad i = 1, \ldots, 3 \tag{3.18}$$

where d_i is the integer delay in the i-th range that maximizes $R[k]$ of Eq. (3.17). The expression on the right-hand side of this equation is similar to the last term of Eq. (2.19), which is the term to be maximized.

The normalized correlations are then weighted with the most weight given to the normalized correlation corresponding to the shortest delays, i.e., for $i = 3$, and least weight to that corresponding to the longest delays, i.e., for $i = 1$. The winner among the three weighted normalized correlations is used to select the best open-loop delay T_{op}. This procedure of dividing the delay range into three sections and favoring the smaller values is used

Table 3.1 G.729: Range(s) and resolution of fractional delay at each subframe.

Subframe	Fractional Delay	Range(s)	Resolution
1	T_1	$[19\frac{1}{3},\ 84\frac{2}{3}]$ $[85,\ 143]$	1/3 integers only
2	T_2	$[\lfloor T_1 \rfloor - 5\frac{2}{3},\ \lfloor T_1 \rfloor + 4\frac{2}{3}]$	1/3

to avoid pitch doubling.

Closed-loop Search

The approach used to implement the pitch filter for delays smaller than the subframe length is somewhat different than any of the techniques mentioned so far. In the G.729 approach, the excitation is interpolated by a filter which can be viewed as a cascade of an ideal interpolation filter and a lowpass filter. An ideal interpolation filter would reproduce the original samples, but the filter used by the G.729 coder will alter them because of its additional lowpass effect. Finally, the excitation vector is formed by recursively calculating its entries; that is, each sample is estimated by weighting the sample at delay T and its nearby samples. In order to estimate sample $(N - \tau - D^{(0)})$, it is necessary to use samples from the current subframe. In this case the already estimated samples are used instead (see Eqs. (3.22) and (3.23)).

In the search stage, the excitation is extended by the LP residual to simplify the search. Each frame is divided into two subframes of 5 ms (or $N = 40$ samples) long and the adaptive codebook search is done every subframe. Table 3.1 summarizes the range(s) and corresponding resolution of the fractional delay for each subframe.

In the first subframe, the delay T_1 is found by searching a small range of 6 integer delay values around the open-loop delay T_0 with a resolution that depends on which of the two specified ranges T_0 falls into. For the second subframe, closed-loop pitch analysis is done around the delay value T_1, selected for the first subframe. In each case, search boundaries d_{min} and d_{max} are defined, where d_{min} and d_{max} are integers.

The expression to be maximized during the search is given as

$$R[k] = \frac{\sum_{n=0}^{N-1} s_t[n]\hat{s}_k[n]}{\sqrt{\sum_{n=0}^{N-1} \hat{s}_k[n]\hat{s}_k[n]}}, \tag{3.19}$$

where $s_t[n]$ is the target signal and $\hat{s}_k[n]$ is the filtered past excitation at delay k (past excitation at delay k convolved with $h[n]$). Note that Eq. (3.19) is equivalent to Eq. (3.10).

The convolution $\hat{s}_k[n]$ is computed for the delay d_{min}. To reduce the amount of computation required to evaluate the convolution, for the other integer delays in the search range $d_{min}+1,\ldots,d_{max}$, the convolution is updated using the recursive relation

$$\hat{s}_k[n] = \hat{s}_{k-1}[n] + \hat{r}[n]h[n], \qquad n = N-1,\ldots,0, \tag{3.20}$$

where $\hat{r}[n],\ n = -143,\ldots,N-1$ is the excitation buffer, and $\hat{s}_{k-1}[-1] = 0$. Note that in the search stage, the samples $\hat{r}[n],\ n = 0,\ldots,N-1$, are not known and they are needed for delays less than $N = 40$. Thus, to simplify the search and make the relation in Eq. (3.20) always valid, the LP residual is copied to $\hat{r}[n]$, for $n = 0,\ldots,N-1$.

For the determination of T_2 and T_1 if the optimum integer closed-loop delay T_0 is less than 85, fractional pitch search has to be carried out around the specified ranges. The fractional pitch search is done by interpolating the normalized correlation in Eq. (3.19) by a factor of 3 ($I = 3$) and searching for its maximum. The interpolation is done using a FIR filter b_1 based on a Hamming windowed $\sin(x)/x$ function, truncated at ± 11 and padded with zeros at ± 12 ($b_1[12] = 0$). The filter has its cut-off frequency at 3600 Hz in the oversampled domain. The interpolated values of $R[k]$ for the fractions $-\frac{2}{3}$, $-\frac{1}{3}$, 0, $\frac{1}{3}$, and $\frac{2}{3}$ are obtained using the interpolation formula

$$R[k]_t = \sum_{i=0}^{8} b_1[3i+t]\, R[k+4-i], \qquad t = 0,1,2, \tag{3.21}$$

where $t = 0,1,2$ corresponds to the fractions $0, \frac{1}{3}$, and $\frac{2}{3}$, respectively.

The non-causal filtering operation implies that the center of the lowpass FIR filter, $b_1[n]$, is aligned with the current sample to be predicted. This requires that τ samples before and after the current sample are known. In this case the length of the filter is 25 and hence $\tau = 4$. Therefore, to allow proper interpolation, it is necessary to compute the correlation terms in Eq. (3.19) using the range $d_{min} - 4,\ d_{max} + 4$.

Generation of the optimum adaptive codebook vector

Once the fractional pitch delay has been determined, the optimum adaptive codebook vector $\upsilon_{opt}[n],\ n = 0, \ldots, N-1$ is computed by interpolating the excitation signal $\hat{r}[n]$ at the optimal integer delay k and fraction t using a longer interpolation filter, as

$$\hat{r}[n] = \sum_{i=0}^{20} b_2[3i+t]\, \hat{r}[n-k+10-i], \qquad n = 0, \ldots, N-1 \quad t = 0, 1, 2, \tag{3.22}$$

and

$$\upsilon_{opt}[n] = \hat{r}[n], \qquad n = 0, \ldots, N-1. \tag{3.23}$$

The new interpolation filter b_2 is based on a Hamming windowed $\sin(x)/x$ function truncated at ±29 and padded with zeros at ±30 ($b_2[30] = 0$). The filter has a cut-off frequency at 3600 Hz in the oversampled domain.

Computation of the adaptive codebook gain

Once the adaptive codebook vector is determined, the optimal adaptive codebook gain β_{opt} is computed as

$$\beta_{opt} = \frac{\sum_{n=0}^{N-1} s_t[n]y[n]}{\sum_{n=0}^{N-1} y[n]y[n]}, \qquad \text{bounded by} \ \ 0 \leq \beta_{opt} \leq 1.2, \tag{3.24}$$

where $y[n]$ is the filtered optimal adaptive codebook vector (zero-state response of the weighted synthesis filter $h[n]$ to $\upsilon_{opt}[n]$) obtained by

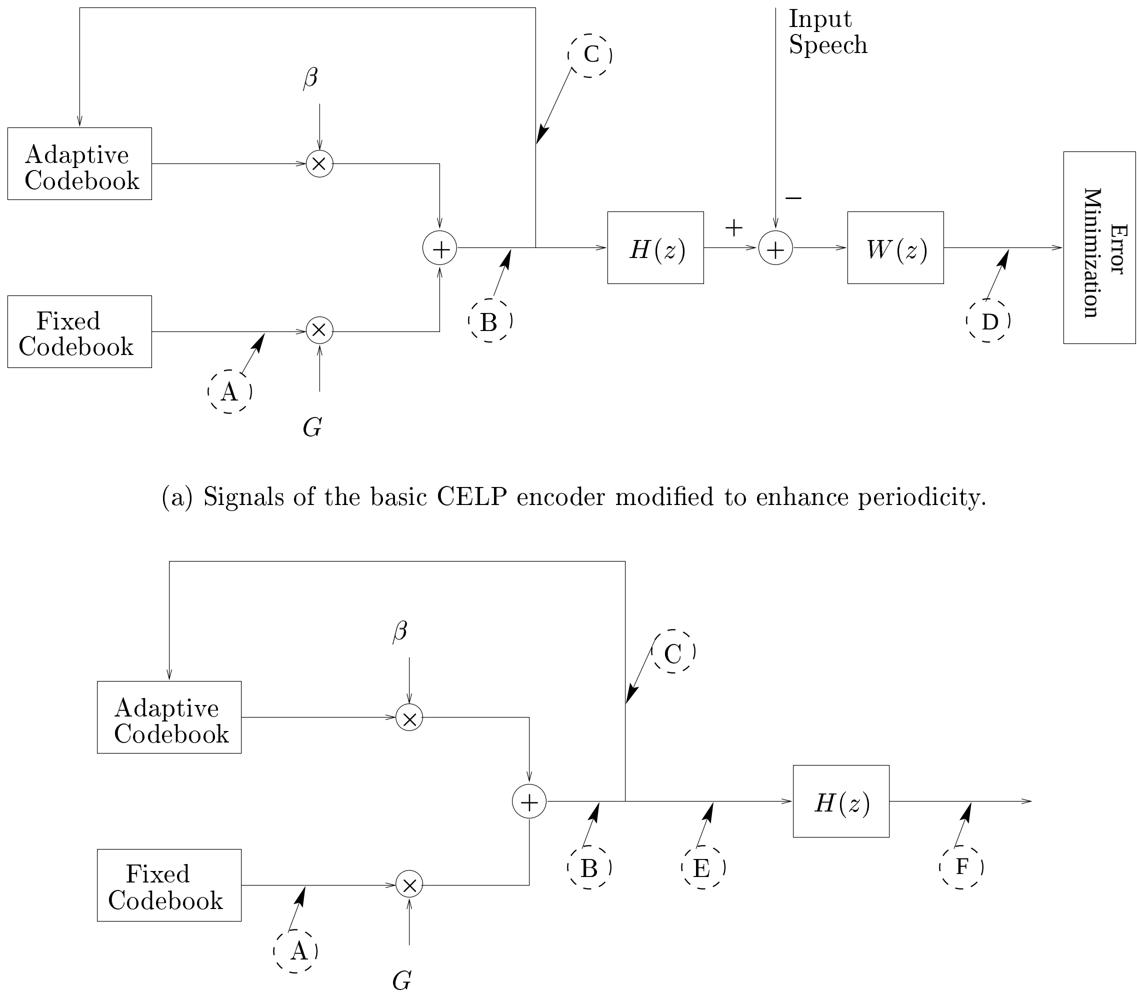
$$y[n] = \sum_{n=0}^{N-1} v_{opt}[i]h[n-i], \qquad n = 0, \ldots, N-1. \tag{3.25}$$

3.2 Improved Modeling of Periodicity

The introduction of an adaptive codebook with sub-sample resolution has significantly improved the quality of reconstructed speech. However, the need for lower bit-rates reduces the number of bits available to model the excitation, thus restricting the fixed codebook to smaller sizes. This results in less accurate waveform matching whose effect is perceived as "noisy" voiced segments of speech.

During steady state voiced speech, the adaptive codebook contributes a large fraction to the resulting excitation. The purpose of the fixed excitation is to provide the missing part. Unfortunately, the fewer number of bits are available to the fixed codebook, the less likely it will be to find a vector that matches well the missing part of the excitation. The problem becomes more acute during high pitched sounds where the pitch period is shorter than the subframe length. In this case, more than one pitch pulse (epoch) is present within a subframe. During onsets of such voiced segments, the adaptive codebook is unlikely to contain a good representation of the residual and therefore its contribution is low. Therefore a higher contribution is expected from the part of the fixed codebook. Even though structured codebooks have been designed to reduce computational complexity and improve periodicity in such cases by optimizing the positions of the samples in the codevector, the small number of non-zero samples available (typically 4 to 5 pulses) can not model well both pitch pulses. It is possible that the selected vector will disturb the periodicity of the resulting excitation signal, or alternatively increase the quantization noise. After the selection of the two contributions, the "noisy" resulting optimal excitation is fed back to the adaptive codebook. As a result, the adaptive codebook is populated with a "noisy" residual and no longer provides the intended purely periodic signal.

To counter this effect, several techniques have been proposed to reduce the presence of noise between the harmonics and enhance periodicity. To aid their description, these techniques are presented in chronological order. In addition, Fig. 3.2 will help indicate where on the basic CELP configuration the modifications have been made. Same indices on the encoder and decoder imply that the same modification is made at both the encoder



(a) Signals of the basic CELP encoder modified to enhance periodicity.

(b) Signals of the basic CELP decoder modified to enhance periodicity.

Fig. 3.5 Improved modeling of periodicity in CELP.

and the decoder.

Adaptive postfiltering

The postfilter aims to enhance the reconstructed speech signal at the decoder. This is done by cascading a long-term with a short-term filter

$$H(z) = H_l H_s = \frac{1}{1-\varepsilon\beta z^{-D}} \frac{A(z/\alpha)}{A(z/\gamma)}, \tag{3.26}$$

where $0 \leq \alpha, \gamma \leq 1$ and $0 \leq \varepsilon \leq 1$.

The short-term postfilter emphasizes the formants and deemphasizes the valleys of the speech signal, thus reducing the audible noise at spectral valleys and increasing it at the formants where the high energy signal renders it inaudible. The frequency response of the long-term filter is that of a comb filter which attenuates the frequency regions between the pitch harmonics. Here again it is assumed that the masking threshold [16] is higher at the harmonics and lower at the frequency regions between them.

Postfiltering was originally applied at position “F” in Fig. 3.5(b) as described in [8], but later in [10] the long-term filter was placed before the synthesis filter, i.e., in position “E” to reduce artifacts in the reconstructed speech introduced by the former configuration.

In general, the distortion introduced by the postfilter on the reconstructed signal reduces the objective Signal-to-Noise ratio, yet it increases its subjective quality.

Comb filtering

Wang and Gersho [11] have proposed comb filtering the resulting excitation. Its purpose is to enhance the pitch harmonics and attenuate the frequency regions between them, with the assumption that any energy between the harmonics is noise and should be discarded. The transfer function of the filter is given as,

$$H(z) = (1 - \eta z^{-1}) \frac{1 + \gamma z^{-D}}{1 - \lambda z^{-D}}, \tag{3.27}$$

where $\gamma = 0.6$, $\lambda = 0.001F_0$ and $\eta = 0.2$, and was placed at position “B” as shown in Fig. 3.2. Both the adaptive and stochastic codebook parameters are determined using the comb filter at the position indicated.

The comb filtering technique has an indirect effect on the way the adaptive codebook is populated. This can be seen by analyzing the way the excitation is chosen and how the adaptive codebook is populated. Basically, the best vector during both the adaptive and stochastic codebook search, is chosen to minimize the weighted difference between a comb filtered synthesized excitation and the input speech. The resulting optimum excitation is finally comb filtered and fed back to update the adaptive codebook.

Constrained excitation

In this approach due to Shoham [12], the gain G of the fixed codebook contribution is constraint to values below its estimated optimal value, based on the performance of the adaptive codebook and the sensitivity of the ear to noisy components between harmonics. The constrained fixed codebook gain is estimated during the fixed codebook closed-loop search for the best gain and index. The estimated reduced gain is the one used to scale the optimum stochastic codevector and construct the resulting optimum excitation, and finally transmitted to the decoder.

Decreasing the amount of the noisy stochastic contribution not only enhances the periodicity of the reconstructed speech but the adaptive codebook is updated with a less noisy excitation as well.

Pitch sharpening

In pitch sharpening [13], only the adaptive codebook update procedure is modified. The objective is to "clean up" the resulting optimal excitation from the noise before is fed back to update the adaptive codebook, i.e., the signal at position "C" as indicated in Fig. 3.2. This is done in two different ways.

First by scaling down the fixed codebook contribution by a factor that equals to the energy of the optimum scaled fixed codebook vector normalized to the energy of the resulting optimal excitation. The resulting reduced gain fixed codebook vector added to the scaled optimum adaptive codebook vector is fed back to update the adaptive codebook.

The second way is by center-clipping the resulting optimal excitation at position "C" before is fed back.

Harmonic noise weighting

This technique, proposed by Gerson and Jasiuk [14], uses a multi-tap pitch analysis filter (or single-tap with fractional delay) in cascade with the original weighting filter to improve the weighting of the error. The filter is placed at position "D" as indicated in Fig. 3.5(a).

The purpose of this filter is to steer the closed-loop selection mechanism to better match the frequency regions between the harmonics of the input signal. This is done by deemphasizing the error at the harmonics and emphasizing it at the frequency regions between them.

As it was found that the performance of the coder is not affected if the adaptive codebook search is done utilizing only the original spectral weighting filter, the harmonic noise weighting filter is only utilized during the fixed codebook search.

Pulse codebook

In this technique, Asakawa *et al.* [15] divided the fixed codebook to a pulse-train codebook and a random codebook. The pulse codebook consists of a set of pulse trains with equal amplitude pulses and spaced by the pitch period. Each vector in this codebook is specified by the position of its first pulse which is the parameter to be optimized during the pulse codebook search.

The coder utilizes a voiced/unvoiced classification mechanism which decides which of the two codebooks will be used for the fixed codebook search. When voiced, the periodicity of the resulting excitation is enhanced by searching the pulse codebook. If the speech is unvoiced, the random codebook is used which models such signals best.

Pitch synchronous innovation

In pitch synchronous innovation [7] the stochastic codebook vector is modified during the closed-loop search at position “A” as indicated in Fig. 3.2. The modification is done during voicing and for delays less than the subframe length. In that case, the first D samples of the vector are repeated using the same procedure as in the adaptive codebook and the stochastic codebook parameters are estimated using the modified vector. This repetition introduces periodicity to the stochastic codevector, thus enhancing the periodicity of the resulting excitation.

G.729 approach

The G.729 approach [5] is similar to the pitch synchronous innovation technique. Here, an adaptive pitch synthesis filter is placed immediately after the fixed codebook at position “A” in Fig. 3.2.

For delays less than the subframe length, the filter modifies the fixed codebook vector according to:

$$c[n] = \begin{cases} c[n] & \text{for } n = 0, \ldots, D-1, \\ c[n] + \beta' c[n-D] & \text{for } n = D, \ldots, N-1, \end{cases} \tag{3.28}$$

with

$$\beta' = \begin{cases} 0.8 & \text{if } \hat{\beta}_{opt}^{(m-1)} > 0.8, \\ \hat{\beta}_{opt}^{(m-1)} & \text{if } 0.2 \leq \hat{\beta}_{opt}^{(m-1)} \leq 0.8, \\ 0.2 & \text{if } \hat{\beta}_{opt}^{(m-1)} < 0.2. \end{cases} \tag{3.29}$$

The parameter $\hat{\beta}_{opt}^{(m-1)}$ is the quantized optimal adaptive codebook gain from the previous subframe.

The fixed codebook search is again done by using the filtered codevectors. Note that the difference between the G.729 and pitch synchronous innovation approach is the scaling factor applied to the repeated samples. In G.729 the repeated samples are scaled whereas in pitch synchronous innovation no scaling is applied. Both techniques aim to enhance the speech during high pitched sounds which as reported are most troublesome.

3.3 Motivation for Proposed Technique

Our approach is focused on the adaptive codebook contribution. The objective is to modify the way the optimal adaptive codebook vector is selected, such that the contribution of the codebook resembles the intended noise-free purely periodic part of the LP residual.

This idea is motivated by the following observations, on which the principles of Waveform Interpolation [39] and Pitch Pulse Evolution model reported in [40] are also based. During steady state voiced speech the pitch period is nearly constant; while occasional doubling and halving of the fundamental frequency might also be observed during voicing onsets. Starting at any arbitrary time instant of such segments of speech, one can easily identify a sequence of pitch pulse waveforms. Observing the evolution of these waveforms, one can see that their general shape evolves slowly with time, often obscured by noisy components that tend to vary form waveform to waveform. Separating the slowly evolving waveform from the rapidly changing noisy component, we can safely assume that the latter is the component supplied by the fixed codebook, while the former is the intended adaptive codebook contribution.

These observations led us to the conclusion that in steady state voicing, the adaptive codebook should supply a pitch waveform, whose shape changes slowly from pulse to pulse. Thus the abrupt changes that the resulting optimal excitation contains have to be removed in order to populate the adaptive codebook with a more "smooth", noise-free signal. This can also be achieved by allowing the adaptive codebook to be populated the usual way, and then remove the noise from the current optimal codevector based on a relatively long history of pitch pulses. The latter technique is the one that our proposed method adopts. The PPA technique is presented in the next section.

3.4 The Pitch Pulse Averaging Technique

As stated in the previous section, the adaptive codebook is populated with the usual approach, i.e., by shifting backwards the excitation buffer by one subframe length, after the resulting optimal excitation has been copied to the current subframe. In Section 3.1.1 it was assumed that the excitation buffer has a length of $D_{max} + N$ samples, of which the first D_{max} samples constitute the past excitation. Setting the length of the past excitation to the longest possible pitch period D_{max}, implies that the past excitation contains at least one pitch pulse waveform. In our approach, the past excitation is extended to contain a number of pitch pulse waveforms, even for the case of having a signal with pitch period as long as D_{max} .

The PPA technique can be divided into two steps.

1. The evolution of the current pitch pulse waveform is extracted from the relatively long excitation history.

2. The noisy component is removed from the current waveform by averaging its evolution.

These two steps are introduced and described in detail in the following sections.

3.4.1 Extraction of pitch pulses

The evolution of the pitch pulses can be extracted if one could identify those waveforms from the past excitation. In brief, this could be done by identifying and extracting the best match to the current waveform, then in the same way find the match to the first match and so on. Here, a match to an arbitrary time instant is defined as the sample from the past that was selected to minimize the weighted mean-squared error of the synthesized speech at that time instant. This sample is identified by the optimal delay estimated for that time instant.

For example, the simplest case would be to identify the best match to the current subframe, i.e. at time instants $n = 0, \ldots, N-1$, when the delay is greater than the subframe length. All the time instants in this period are assigned the same delay $D^{(0)}$, thus their matched samples are found $D^{(0)}$ samples back. Thus, the best match for time instant $n = q,\ q = 0, \ldots, N-1$ is sample $\hat{r}[q - D^{(0)}]$. This can also be written as

$$S_0[n] = \hat{r}[n - D^{(0)}], \qquad \text{for } n = 0, \ldots, N-1, \tag{3.30}$$

where $D^{(0)}$ is the optimal integer delay found for the current subframe, after a closed loop search of the adaptive codebook.

In order to find the second waveform, the match to the time instants that constitute the first match, i.e. time instants $n = q - D^{(0)},\ q = 0, \ldots, N-1$, have to be identified. Here, there might be a case where different time instants are assigned with different delays because they belong into different subframes. Once the subframe(s) that those time instants belong to are identified, their matches can be found by using their assigned delays. The second waveform can thus be constructed as follows:

$$S_1[n] = \begin{cases} \hat{r}[n - D^{(0)} - D^{(k+1)}] & \text{for } n = 0, \ldots, D^{(0)} - kN - 1 \\ \hat{r}[n - D^{(0)} - D^{(k)}] & \text{and } n = D^{(0)} - kN, \ldots, N-1 \end{cases}, \tag{3.31}$$

where

$$k = \left\lfloor \frac{D^{(0)}}{N} \right\rfloor. \tag{3.32}$$

This procedure is shown graphically in Fig. 3.4.1. The relative values assigned to the delays are typical for a voiced segment of speech. In general, the delays of consecutive subframes are different, with large differences observed during unvoiced segments of speech or even during voicing onsets where pitch doubling might occur.

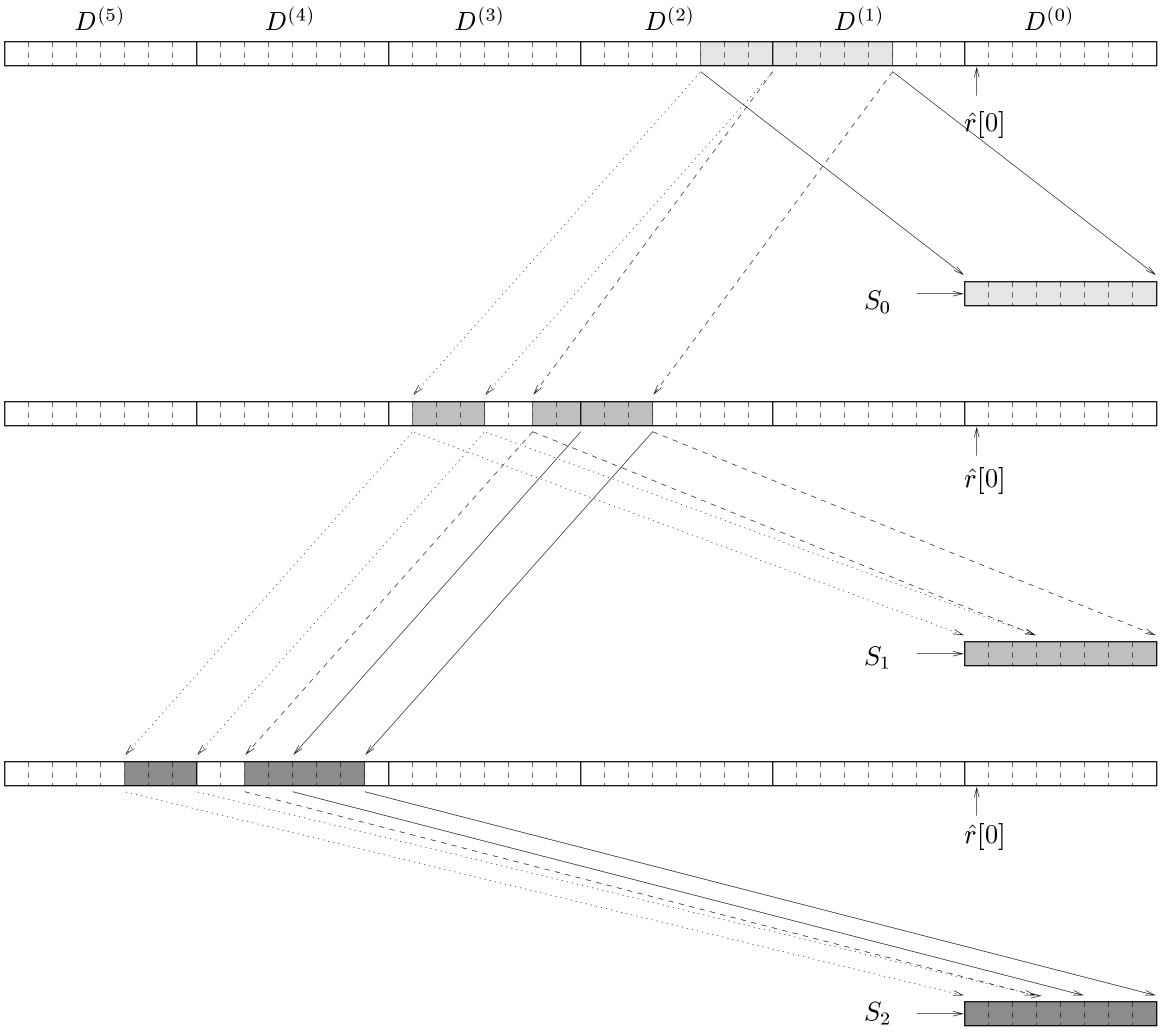


Fig. 3.6 Pitch pulse waveform extraction using a *breadth-first* search procedure.

Assumed values: $N = 8$, $D^{(k)} = 11, 10, 12, 12$, for $k = 0, 1, 2$ and 3, respectively.

Depth-First search procedure using integer delay values

The approach for extracting the pitch pulse waveforms introduced above uses a *breadth-first* search procedure, where all the elements of one waveform are extracted before moving to the next waveform. This makes this approach rather complex to implement. An alternative approach is to use a *depth-first* search procedure, where all the elements having the same index n in each waveform i are extracted before advancing the index to the next element.

The pitch pulse waveforms $S_i[n]$ are now given as:

$$S_i[n] = \hat{r}[n - P(i)], \qquad \begin{array}{l} \text{for } i = 0, \ldots, L, \\ \text{and } n = 0, \ldots, N-1, \end{array} \tag{3.33}$$

where

$$P(i) = P(i-1) + D^{(k)}, \qquad \text{with } P(0) = D^{(0)}, \tag{3.34}$$

and

$$k = \begin{cases} 0 & \text{if } (-P(i-1)+n) > -1, \\ \left\lfloor \dfrac{P(i-1)-n-1}{N} \right\rfloor + 1 & \text{otherwise.} \end{cases} \tag{3.35}$$

In Eq. (3.34), $D^{(k)}$ denotes the optimal delay found for the k-th subframe in the past, as illustrated in Fig. 3.4.1. The integer L denotes the total number of waveforms extracted. Its value depends and varies according to the following factors:

- The past excitation length n_{max}, i.e., how far in the past we want to consider the evolution of the current subframe. This value is fixed and determined empirically.
- The values of the delays encountered in the previous subframes. The longer the delays, the smaller the total number of waveforms extracted and vice-versa.

This approach is demonstrated in Fig. 3.4.1 for the first and last elements of the first three waveforms in the previous example.

Practical considerations for delays less that the subframe length

So far we have assumed that the delay at the current subframe is greater than the subframe length and thus all the examples given follow this assumption.

When this delay is smaller than the subframe length, $N - D^{(0)}$ samples need to be read from the current subframe in order to completely define the first pitch pulse waveform $\mathbf{S}_0$

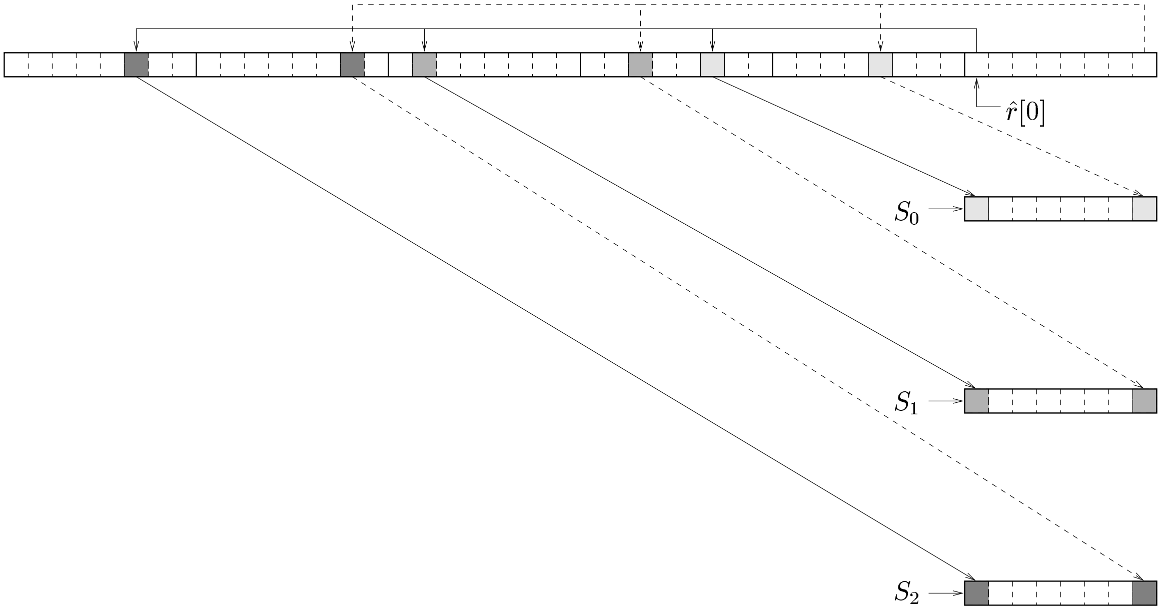


Fig. 3.7 Pitch pulse waveform extraction using a *depth-first* search procedure.

Assumed values: $N = 8$, $D^{(k)} = 11, 10, 12, 12$, for $k = 0, 1, 2$ and 3, respectively.

(see Eq. (3.34)). Unfortunately, the excitation at the current subframe is still unknown and thus those samples are not defined.

This problem can be solved by either of the following two methods:

1. Use the adaptive codebook approach to repeat the extracted waveform, or

2. Use the G.729 approach to form the optimal adaptive codebook vector and then copy this vector to the current subframe.

The second approach would have been exactly the same as the first one if the filter used by the G.729 coder for interpolating the excitation, was an ideal interpolation filter. In that case, the samples would have been reproduced and thus the last $N - D^{(0)}$ samples would have been repeated. Note that the purpose of the interpolation filter was to introduce fractional delays. Nevertheless, it is used for integer delays as well and in this case its effect is to lowpass the excitation. When fractional delays are introduced, the excitation can not be repeated without using samples from the current subframe to calculate the first $D^{(0)}$ samples. This is due to the fact that the subfilter coefficients corresponding to future samples are no longer zero, even in the case of a proper interpolation filter. This implies that the same procedure employed by G.729 to construct the adaptive codebook vector (see Eqs. (3.22) and (3.23)) has to be followed to calculate the last τ samples before repetition begins.

Assuming that the lowpass effect of the filter does not change significantly the resulting vector, similar results as with the first method can be obtained by just coping the vector formed by the G.729 approach to the current subframe. This choice is also motivated by the following facts:

- The PPA algorithm is tested on the G.729 coder and we are trying to minimize the changes to the existing code.
- It was reported that both techniques give similar results.
- As it was already explained, when fractional delays are used the first method is not possible to implement without using a similar approach as the second method.

Depth-First search procedure using fractional delay values

The *depth-first* search procedure described earlier, employs integer delays to extract the pitch pulses. As noted in Section 3.1.3, better matching can be achieved if delays of higher resolution are used. Thus, the procedure described above has been modified to operate with fractional delays.

The introduction of fractional delays can be done in two ways as follows:

1. *Using polyphase filters.*

   This method is rather cumbersome to implement and requires a large amount of computation for the following reasons:

   - The delayed past excitation will have to be filtered with the appropriate subfilter for each sample of every waveform to be extracted. This implies that the amount of computation will increase substantially.
   - The computation of the overall integer delay and fraction for each sample to be extracted is rather complicated.

These characteristics makes this method undesirable, therefore the method described next is the one that we adopted.

2. *Interpolating the excitation buffer.*

   As described in Section 3.1.3, fractional delays of resolution I can be implemented by interpolating the excitation buffer by a factor I. Thus, a fractional delay expressed as an integer delay T and a fraction t/I in the original excitation, is equivalent to an integer delay of $IT+t$ samples in the interpolated excitation. Following this principle, the pitch pulse waveforms are now given as

$$S_i[n] = \hat{r}_{int}[-P(i) + nI], \qquad \begin{array}{l} \text{for } i = 0, \ldots, L, \\ \text{and } n = 0, \ldots, N-1, \end{array} \tag{3.36}$$

   where

$$P(i) = P(i-1) + (IT^{(k)} + t^{(k)}), \qquad \text{with } P(0) = IT^{(0)} + t^{(0)}, \tag{3.37}$$

   and

$$k = \begin{cases} 0 & \text{if } (-P(i-1) + In) \geq 0, \\ \left\lfloor \dfrac{P(i-1) - In - 1}{IN} \right\rfloor + 1 & \text{otherwise.} \end{cases} \tag{3.38}$$

   In Eq. (3.37), $\hat{r}_{int}$ denotes a pointer to the beginning of the interpolated current subframe in the interpolated excitation; that is, to the sample point in the interpolated excitation that corresponds to the first sample of the current subframe in the original excitation. The integer and fractional part of the delay in the k-th subframe are denoted as $T^{(k)}$ and $t^{(k)}$ respectively.

Defining the number of waveforms extracted

Using the *depth-first* search procedure to extract the pitch pulse waveforms, there might be cases where some of the extracted waveforms are not complete, i.e., some of their sample entries were matched to entries beyond the end of the existing past excitation. These waveforms are thus rejected and only the ones completed are used to define the total number of waveforms extracted, N_w. For example, if the pitch period of a voiced segment is approximately 70 samples, then the maximum number of extracted waveforms is 10.

3.4.2 Averaging of pitch pulses

After the pitch pulse waveforms have been extracted, the noisy component can be removed from the intended adaptive codebook vector by averaging these waveforms. The adaptive codebook vector found for the current subframe, resembles very closely the first pitch pulse waveform extracted. This is the most recent waveform in the evolution of the pitch pulses and thus should be emphasized most. As the waveforms age in time, i.e., those extracted from samples further in the past of the excitation, their relevance decreases and therefore they should be given less emphasis.

Since it is very important that we are be able to control the number of waveforms that are emphasized most, the weighting function is required to have a varying shape. This requirement led us to the choice of a Kaiser window. In continuous time, a Kaiser window is specified by the following equation:

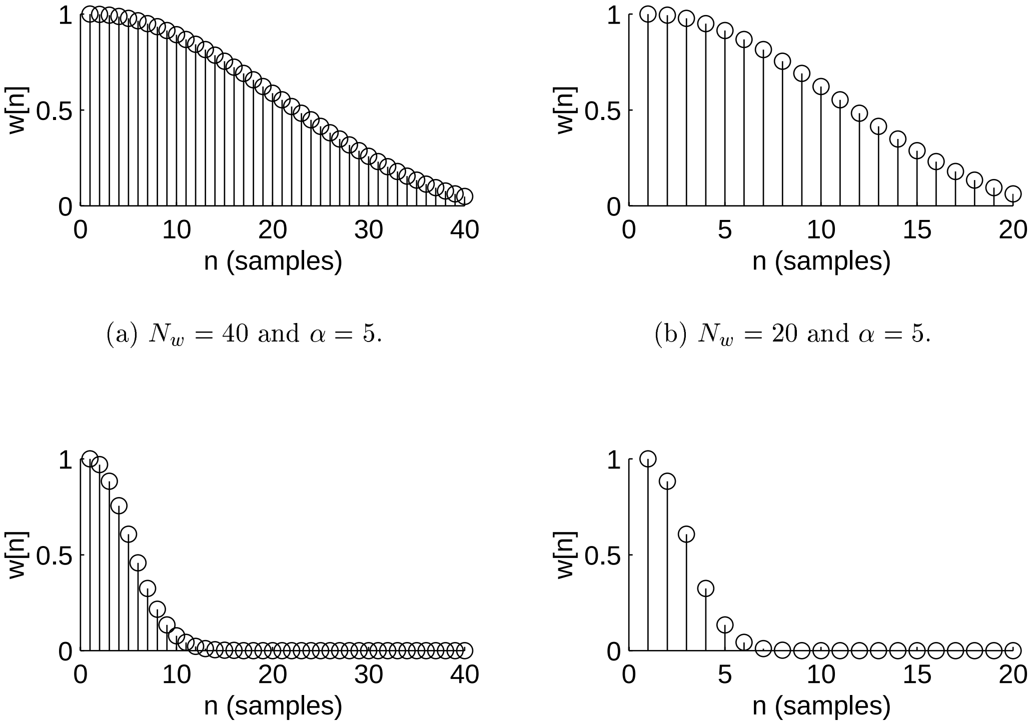
$$w(t) = \begin{cases} \dfrac{I_0(\alpha\sqrt{(1-t^2)})}{I_0(\alpha)} & \text{for } -1 \le t \le 1, \\ 0 & \text{otherwise,} \end{cases} \tag{3.39}$$

where $I_0(\cdot)$ is the zeroth order modified Bessel function of the first kind.

The discrete-time one-sided window of length N_w is obtained by setting

$$t = \frac{n}{(N_w - 1)}, \qquad \text{for } n = 0, \ldots, N_w - 1, \tag{3.40}$$

which provides the weights for the N_w extracted waveforms. The independent window parameter α determines the shape of the window. Increasing α, the width of the window decreases and thus only the most recent waveforms are emphasized. For $\alpha = 0$, the window is a rectangular and thus all waveforms are weighted equally. Its value is also estimated empirically. Figure 3.8 shows examples of the weighting function for different values of N_w



(a) $N_w = 40$ and $\alpha = 5$.

(b) $N_w = 20$ and $\alpha = 5$.

(c) $N_w = 40$ and $\alpha = 100$.

(d) $N_w = 20$ and $\alpha = 100$.

Fig. 3.8 Examples of the weighting function $w[n]$.

and α.

The pitch pulse waveforms $\mathbf{S}_i$ are now weighted with their corresponding weight $w[i]$ and added together to form the averaged waveform $\boldsymbol{v}_{av}$, given as

$$\boldsymbol{v}_{av} = \sum_{i=0}^{N_w-1} w[i]\mathbf{S}_i. \tag{3.41}$$

Before forming the new adaptive codebook vector for the current subframe, the gain difference between the averaged waveform $\boldsymbol{v}_{av}$ and the one intended to be supplied by the original coder, $\boldsymbol{v}_{opt}$, has to be compensated. Note, for the purposes of this thesis, the term “original coder” refers to the unmodified coder. Since the extracted waveforms are not orthogonal, normalization of the energy of the weights does not solve the problem. A simple way to compensate for this difference is to multiply the averaged waveform with a gain-scaling factor given as

$$g = \sqrt{\frac{\sum_{i=0}^{N-1} \upsilon_{opt}^2[n]}{\sum_{i=0}^{N-1} \upsilon_{av}^2[n]}}. \tag{3.42}$$

The gain-scaled averaged waveform $\tilde{\boldsymbol{\upsilon}}_{av}$ is given by

$$\tilde{\boldsymbol{\upsilon}}_{av} = g\boldsymbol{\upsilon}_{av}. \tag{3.43}$$

In the next chapter we will verify that this adaptive gain-control technique works. The vector $\tilde{\boldsymbol{\upsilon}}_{av}$ replaces the originally estimated optimal adaptive codebook vector $\boldsymbol{\upsilon}_{opt}$ and is subsequently used to calculate the gain β_{opt} and form the adaptive codebook contribution for the current subframe.

Chapter 4

Simulation of the PPA Algorithm and Results

This chapter presents the simulation and experimental results of the PPA algorithm, described in the previous chapter. The platform chosen to simulate and test the algorithm is the floating-point C simulation of the ITU Rec. G.729 8 kbit/s CS-ACELP codec. This choice was motivated by the fact that it represents the latest technology of speech coders.

The G.729 coder operates on speech frames of 10 ms, which corresponds to 80 samples at a sampling rate of 8000 samples per second. A frame is subdivided into 2 subframes of 5 ms (40 samples) each. The input speech signal is analyzed every 10 ms frame to extract the linear-prediction filter coefficients, whereas the excitation parameters (pitch delay, fixed codebook indices and gains) are estimated on a subframe basis. The fractional pitch delay uses a 1/3 resolution.

For the purpose of this thesis, a brief description of the steps followed by the coder to construct the adaptive codebook contribution and how the codebook is finally populated, is provided below. It should be noted though that the steps described here, are the ones that relate to our work.

1. Find the fractional pitch delay for the current subframe. At this stage, the current subframe contains the LP residual.
2. Compute the adaptive codebook vector using the fractional delay found above. Here, the filtering approach is used as described in Section 3.1.4. The new samples overwrite the LP residual in the current subframe.

3. Calculate the pitch gain, β_{opt}, using Eq. (3.24).
4. Compute the fixed codebook vector and gain.
5. Construct the total excitation by multiplying each codebook vector with its corresponding gain and adding the two resulting contributions. The result overwrites the adaptive codebook vector in the current subframe.
6. Repeat all the above steps for the second subframe.
7. Update the excitation buffer for the next frame by shifting it to the left by one frame length (80 samples).

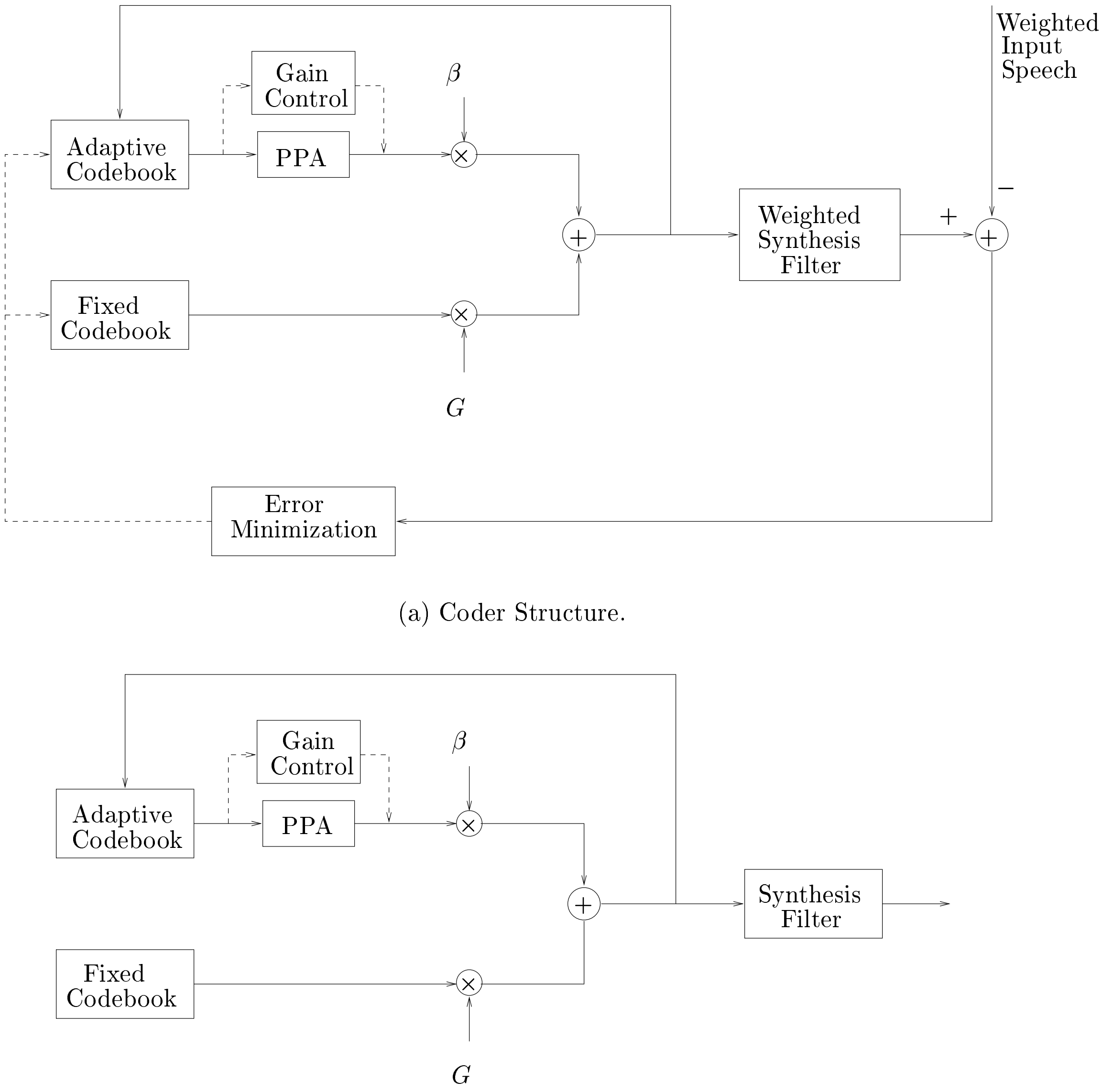
4.1 The New Coder Structure

The basic coder structure has been altered to accommodate the PPA algorithm. The block diagram of Fig. 4.1 indicates the place on the coder and decoder where the PPA algorithm has been added.

Most of the alterations made were additions. The only substantive modification done to the existing code was the increase of the past excitation length. Originally, the length of the past excitation (not including the current frame which is not consider to be past) was equal to the maximum allowable pitch, plus the length of the interpolation filter memory; that is $(143+10)$ samples. The length of the past excitation was increased to $(5 \times 143+10)$. This implies that the past excitation buffer at any time instant contains a minimum of 5 pitch pulses and a maximum of 35. These results correspond to the case where the pitch period takes the maximum (143 samples) and minimum (20 samples) allowable value, respectively.

The PPA algorithm has now been embedded to the procedure described in the previous section, as follows:

1. (same as before).
2. (same as before).
   Note that this operation is important since the samples in the current subframe are no longer undefined.
   (a) Extract the pitch pulse waveforms using fractional delays.
   (b) Weight the extracted waveforms.
   (c) Gain-scale the averaged waveform.



(a) Coder Structure.

(b) Decoder Structure.

Fig. 4.1 Modified Codec Structure.

(d) Copy the resulting weighted waveform to the current subframe.

3. (same as before).
4. (same as before).
5. (same as before).
6. (same as before).
7. (same as before).

At the decoder, after the fractional pitch delay has been decoded, steps (2) to (2d) are repeated and then the decoded fixed codevector and gains are used to construct the total excitation.

4.2 Simulation

In an attempt to validate the correctness of the algorithm, several speech files of different pitch have been used as the input to the coder. Here we only present the results obtained during the simulation with a high pitch and an average pitch speech file. This choice enabled us to study the effectiveness of the algorithm during not only unvoiced and voiced segments of speech, but also segments where a subframe contains more than one pitch pulse. For the latter case, a pitch period of less than 40 samples, most preferably around 25 samples, is required. Because such high pitch speech files were initially not available, we artificially increased the pitch of a female voice speech file to create the file "xtest1.au". For the following results, the algorithm was set to search up to a maximum of 715 samples (approximately 89 msec) from the past, i.e. n_{max} was set to 715.

4.2.1 Extraction of pitch pulse waveforms

A voiced segment of the LP residual and the excitation signal constructed by the original coder is shown in Figures 4.2 and 4.3, respectively. This signal corresponds to an utterance spoken by a male.

The extracted waveforms during the onset, steady state and end of voicing are shown in Figures 4.4(a), 4.4(b) and 4.4(c) respectively. In the figures shown, the zeroth waveform is the most recent. The extracted waveforms in the above three cases correspond to the subframes in Fig. 4.3 (in shaded regions) denoted as "A", "B" and "C". Similarly, Figures

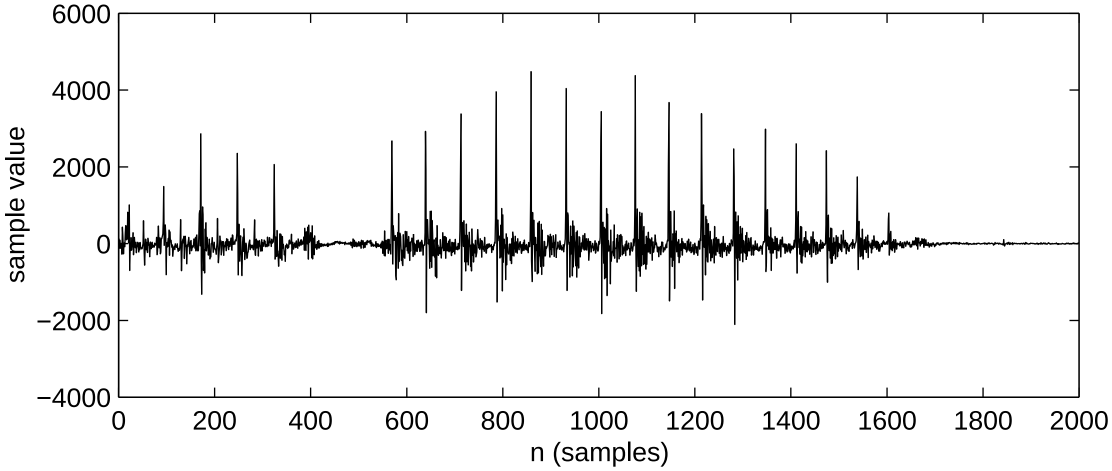


Fig. 4.2 Segment of the LP residual from the male speech file "pb1m1.au".

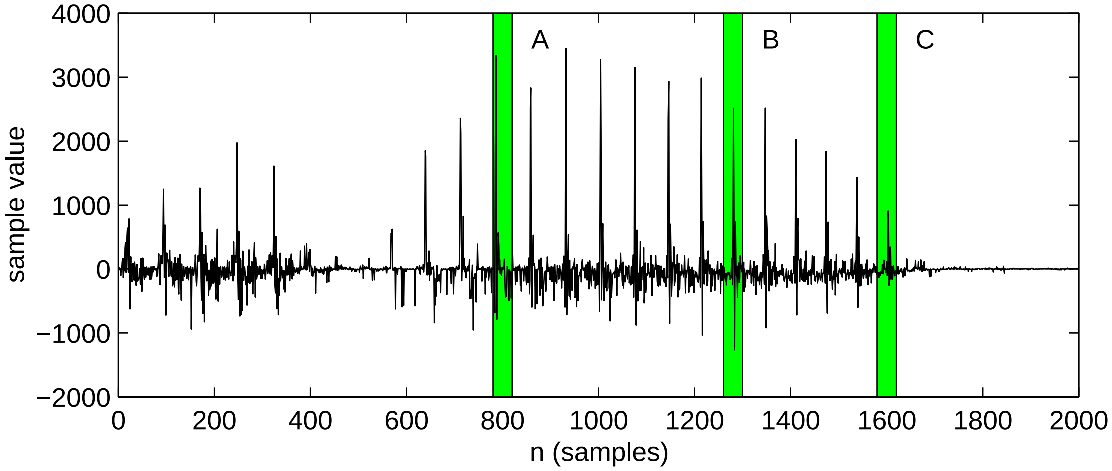
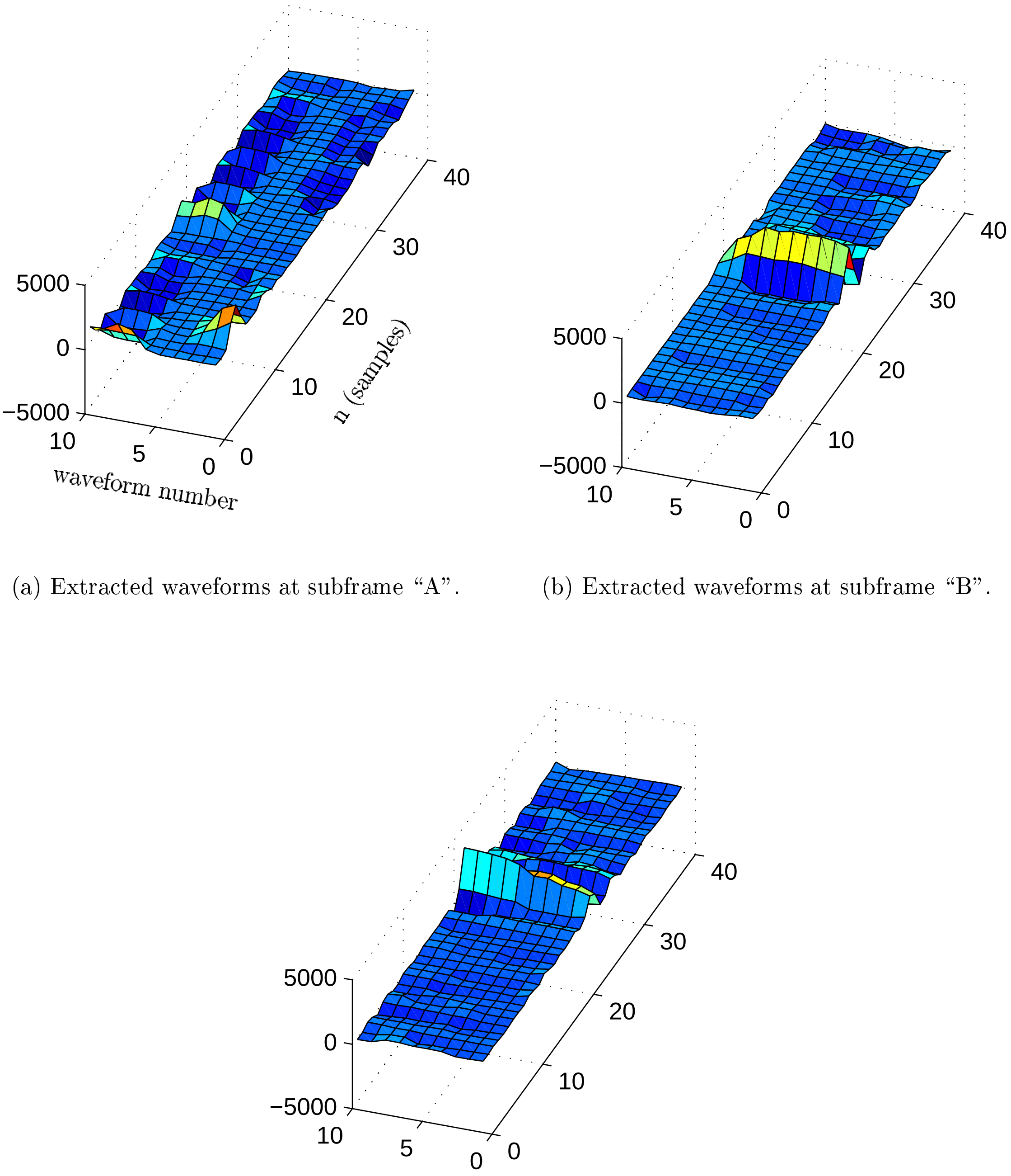


Fig. 4.3 Excitation segment from "pb1m1.au"constructed by the original coder.



(a) Extracted waveforms at subframe "A".

(b) Extracted waveforms at subframe "B".

(c) Extracted waveforms at subframe "C".

Fig. 4.4 Extracted waveforms from the excitation of "pb1m1.au".

4.5(a) and 4.5(b) show a voiced segment of the excitation corresponding to a female spoken utterance (from file “xtest1.au”) and the extracted waveforms at the indicated subframe.

During unvoiced segments of speech, the extracted waveforms are more noise-like since pitch periodicity is absent. This case is demonstrated in Fig. 4.6. The segment of unvoiced excitation shown was taken from the same male voice speech file.

All the above waveforms have been extracted by omitting steps (2b), (2c) and (2d) in the procedure described in Section 4.1. Thus the waveform extraction process itself did not alter the excitation in any way and the resulting excitation is the one constructed by the original coder.

4.2.2 Waveform weighting

The weight applied to each extracted waveform is basically controlled by the independent parameter α of Eq. (3.39). The value of α affects the following:

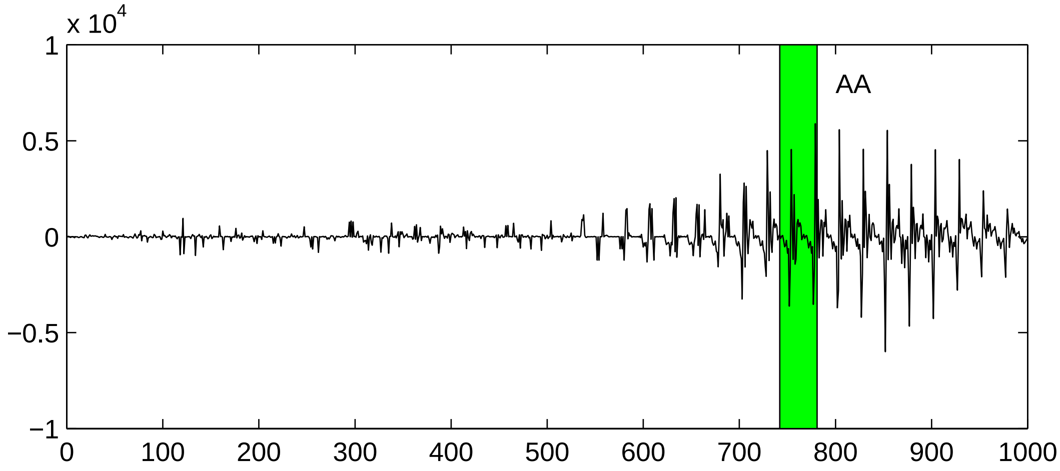
- The averaged adaptive codebook vector,
- The stochastic codebook gain, and
- The population of the adaptive codebook.

Thus if waveform weighting is employed, the extracted waveforms at the subframes indicated in Fig. 4.3 will not be exactly the same as the ones shown in Fig. 4.4. The same applies for the resulting excitation. This situation is illustrated in the following example where the same male voice speech file was processed with $\alpha = 5$. Note that this is an extreme case since waveforms far in the past are still given a considerable weight.

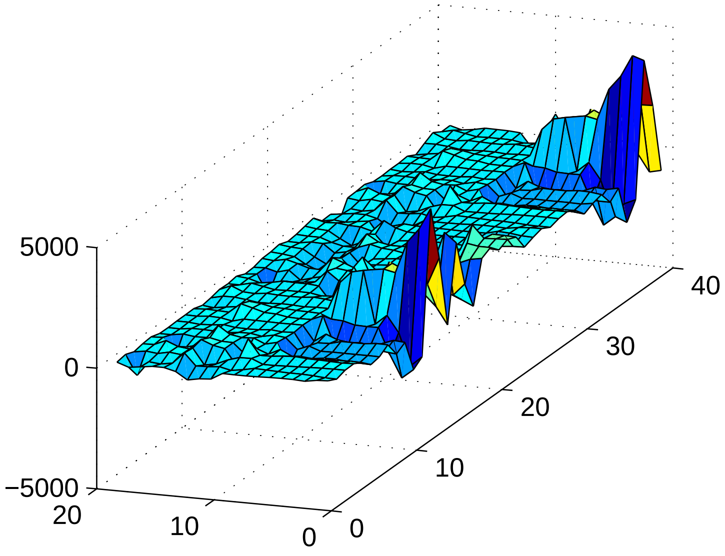
The constructed excitation for this value of α and the associated extracted waveforms are shown in Figs. 4.7 and 4.8. As expected, the envelope of the pitch pulses is more smooth and therefore distorted.

4.2.3 Comments

During voiced segments the pitch pulse waveforms extracted follow closely the evolution of the pitch pulse. A major contribution to the successful extraction of the pitch pulses by the algorithm is the characteristic of the coder to favor delays in the lower range (see Section 3.1.4) during the open-loop pitch analysis. In doing so, pitch doubling is avoided

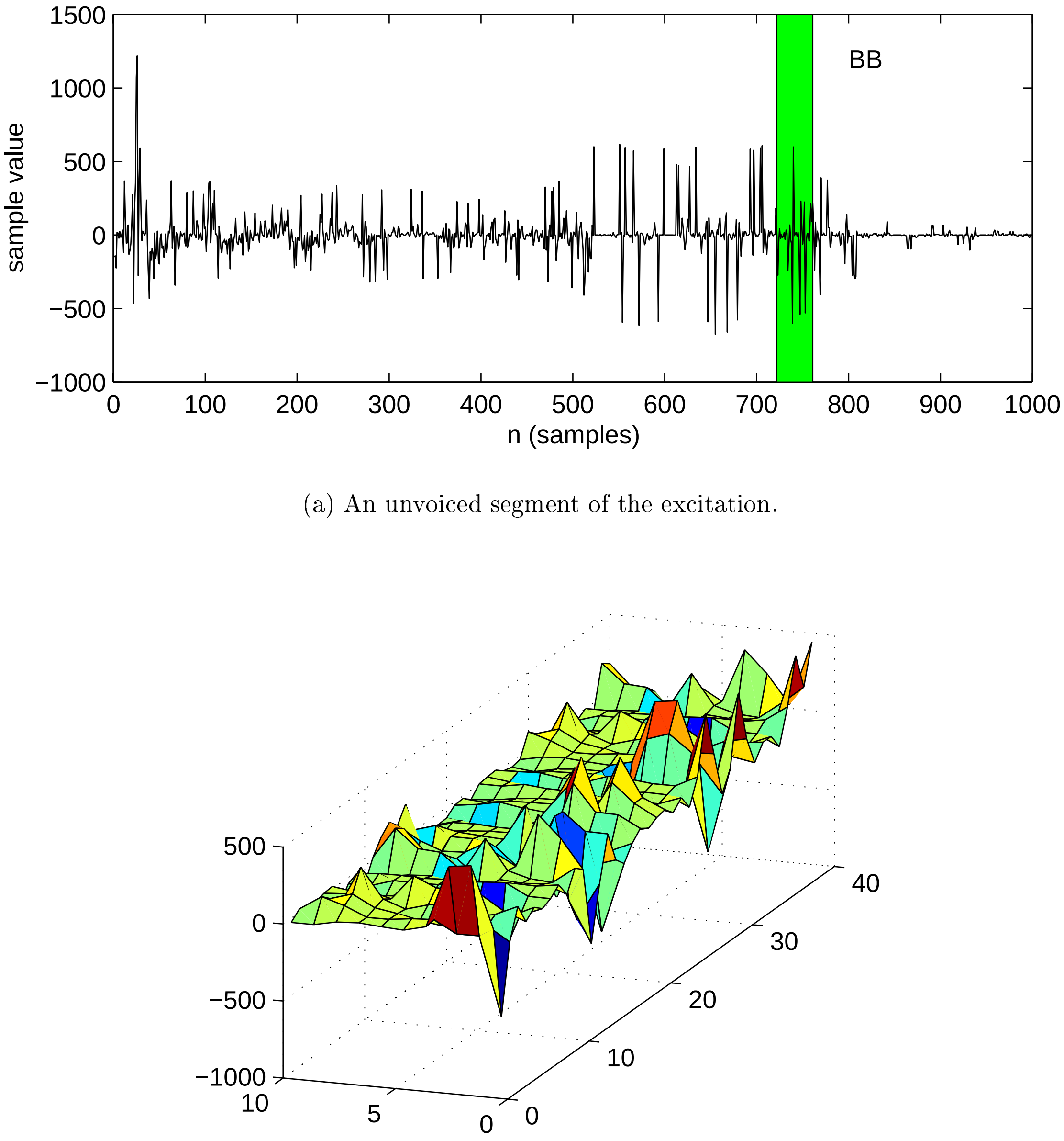


(a) Segment of the excitation.



(b) Extracted waveforms at subframe “AA”.

Fig. 4.5 Extracted waveforms from the excitation of “xtest1.au”.



(a) An unvoiced segment of the excitation.

(b) Extracted waveforms at subframe "BB".

Fig. 4.6 Extracted waveforms during an unvoiced segment from the LP residual of "pb1m1.au".

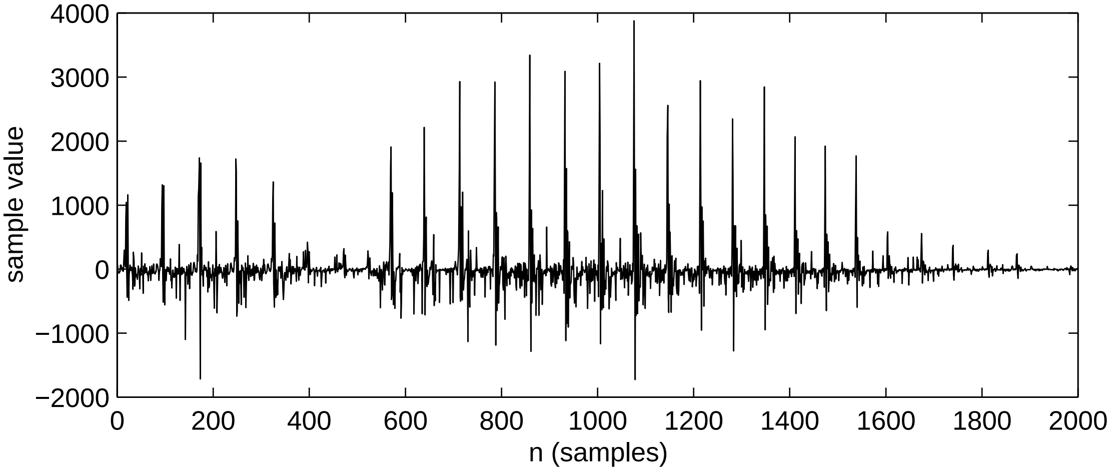


Fig. 4.7 Constructed excitation segment for $\alpha = 5$ in “pb1m1.au”.

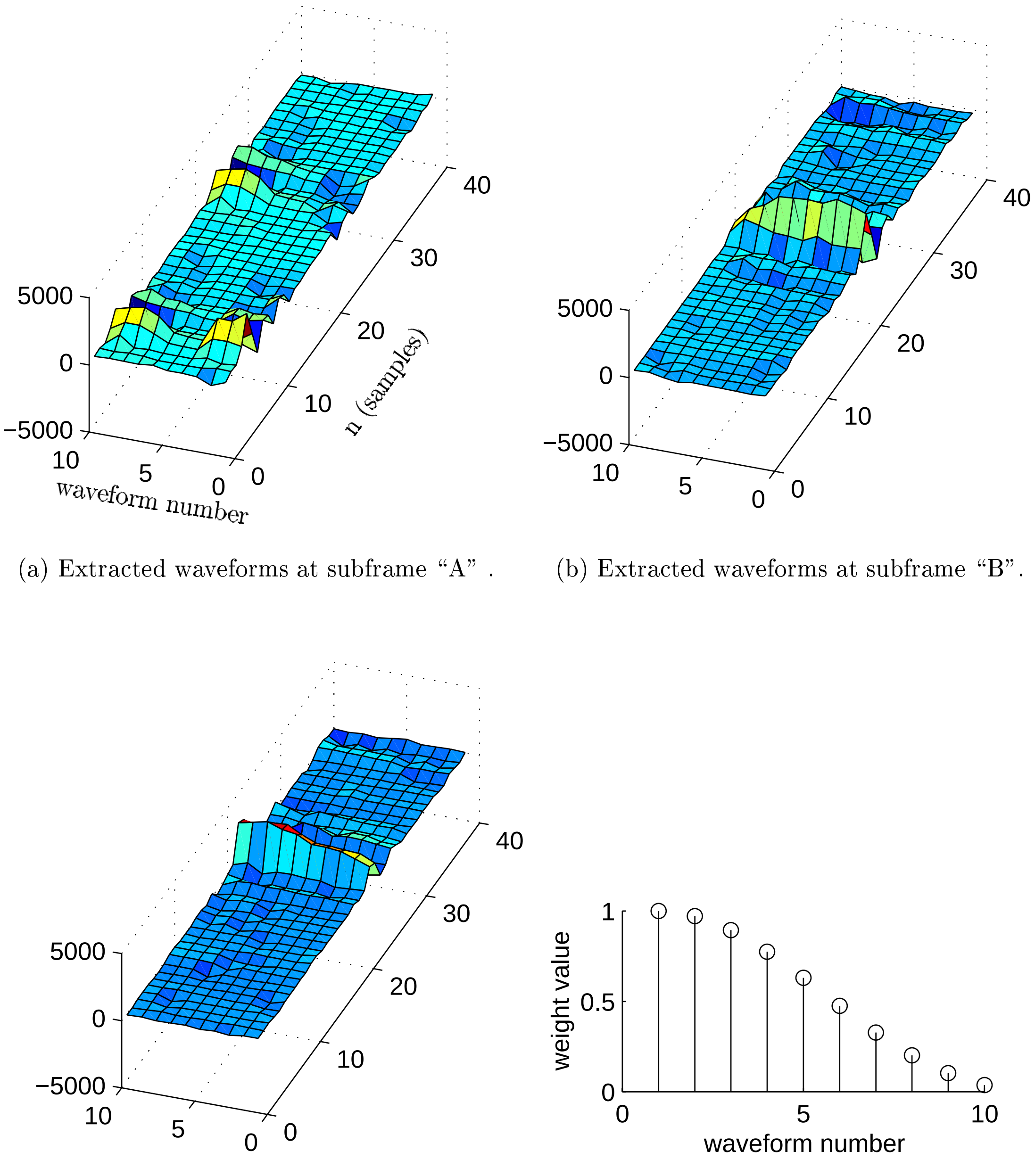
and the algorithm successfully extracts the pitch pulses in order of precedence. Thus the pitch pulses can be averaged effectively and produce the desired result.

This is not the case for unvoiced segments of speech. The pitch period during such segments varies substantially from subframe to subframe. Thus it is possible that samples of, for example, the first extracted waveform are older (arise from further in the past) than the ones comprising the second extracted waveform. In conclusion, the waveforms extracted by the algorithm during unvoiced segments are meaningless and random, thus any further processing should not contribute much to the resulting waveform. Nevertheless, it is preferable that during such segments, most emphasis is given to the first extracted waveform.

4.3 Results

This section presents the results of tests carried out to measure the performance of the algorithm. Along this process, weaknesses of the basic algorithm are identified and the appropriate modifications are presented.

The performance of the algorithm is measured by comparing the modified coder with the original coder using objective and subjective tests. The objective tests enabled us to identify the weak points of the algorithm, make the appropriate modifications where possible and calibrate the independent parameter α. Therefore these tests were carried out



(a) Extracted waveforms at subframe “A” .

(b) Extracted waveforms at subframe “B”.

(c) Extracted waveforms at subframe “C”.

(d) Applied weights for $\alpha = 5$.

Fig. 4.8 The extracted waveforms at indicated subframes in Fig. 4.3 for $\alpha = 5$.

first and at the end, the algorithm is tested through some informal listening tests.

4.3.1 Objective tests

Recall that our goal is to reduce the noise during voiced segments of speech, and that a great source of this noise is the stochastic contribution to the resulting excitation. From this observation we can safely deduce that the quality of the synthesized speech would improve if during such segments the adaptive codebook contribution is increased and the corresponding stochastic codebook contribution is reduced.

This requirement can be quantified in the following three objective measures:

1. The cross-correlation coefficient $\rho(0)$ between the filtered adaptive codebook vector $y[n]$ (zero-state response of the weighted synthesis filter $h[n]$ to $v_{opt}[n]$) and target vector $s_t[n]$ given by

$$\rho(0) = \frac{\sum_{n=0}^{N-1} s_t[n]y[n]}{\sqrt{\sum_{n=0}^{N-1} {s_t}^2[n] \sum_{n=0}^{N-1} y^2[n]}}. \tag{4.1}$$

2. The quantized gain of the fixed codebook vector, G, as calculated by the coder itself.

3. A more general measure of performance is the mean-squared error, MSE, between the weighted synthesized and weighted input speech vectors, denoted as $\boldsymbol{s}_w$ and $\hat{\boldsymbol{s}}$ respectively. This is calculated as,

$$MSE = \frac{1}{N} \sum_{n=0}^{N-1} (s_w[n] - \hat{s}[n])^2. \tag{4.2}$$

   As it will be evident from the results that follow, the first two objective tests are not very well correlated, i.e., an increase in the correlation coefficient does not always result in a decrease in the fixed codebook gain. Thus, more emphasis is placed in minimizing the mean-squared error as defined above.

Note that for the modified coder, $y[n]$ in Eq. (4.1) is replaced with $y_{av}[n]$, and $v_{opt}[n]$ with $v_{av}[n]$.

The first tests were carried out using the female speech file of Section 4.2.1. The correlation coefficient and fixed codebook gain, calculated on a subframe basis, and LP residual of the speech segment under study, are shown in Fig. 4.9.

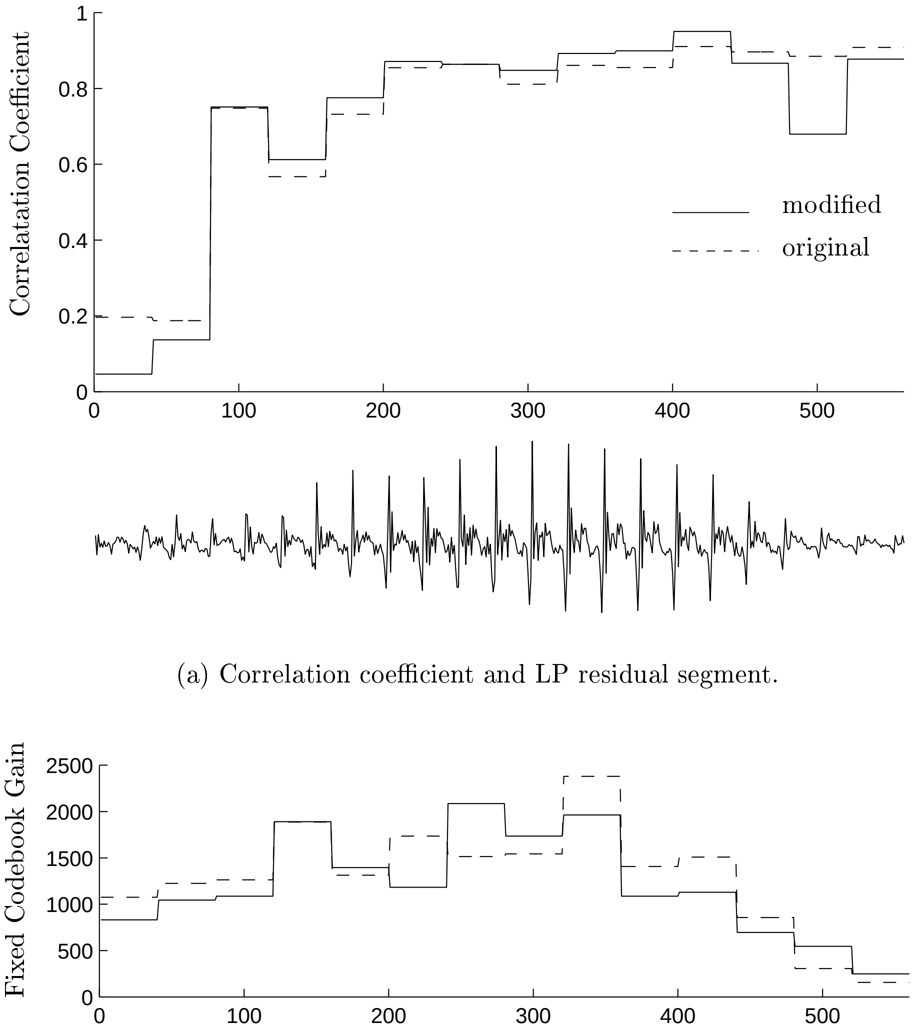
The correlation coefficient is generally higher during the onsets and steady state voicing than originally, but it becomes worse during the end of the voiced segment. This action can be explained as follows:

- During the onset, only the first few extracted waveforms are a good representation of the desired one and these are the ones with the greatest energy. The rest of the waveforms, originating from further in the past, are mainly low energy noise. The weighting function naturally considers only the first few waveforms and a very small weight is given on the rest.
- During steady state, more waveforms of high energy and good representations of the desired one are present and therefore again the algorithm is generally doing well. In the example shown, the amplitudes of the pitch pulses vary rather smoothly from subframe to subframe. In cases where this is not true, the algorithm does not perform that well because the waveforms extracted have varying amplitudes.
- During the end of the voiced segment, the most recent extracted waveforms have considerably smaller energy compared to the older ones (for example, see Fig. 4.4(c)). This greatly reduces the effectives of the weighting function which attempts to deemphasize the older waveforms. The effect of this is to extend the pitch pulses beyond the end of the voiced segment, or in other words, provide a waveform that does not match well the predicted one. This situation is demonstrated in an exaggerated form in Fig. 4.7.

The fixed codebook gain is generally lower than the original during the entire voiced segment.

4.3.2 Refinements

In an attempt to improve the performance of the algorithm by increasing the correlation coefficient during the end of voiced segments as well as during "anomalous" voicing conditions (voiced segments of considerable variations in the pitch pulse amplitudes), we tried



(a) Correlation coefficient and LP residual segment.

(b) Fixed codebook gain.

Fig. 4.9 Results for the modified coder with $\alpha = 140$.

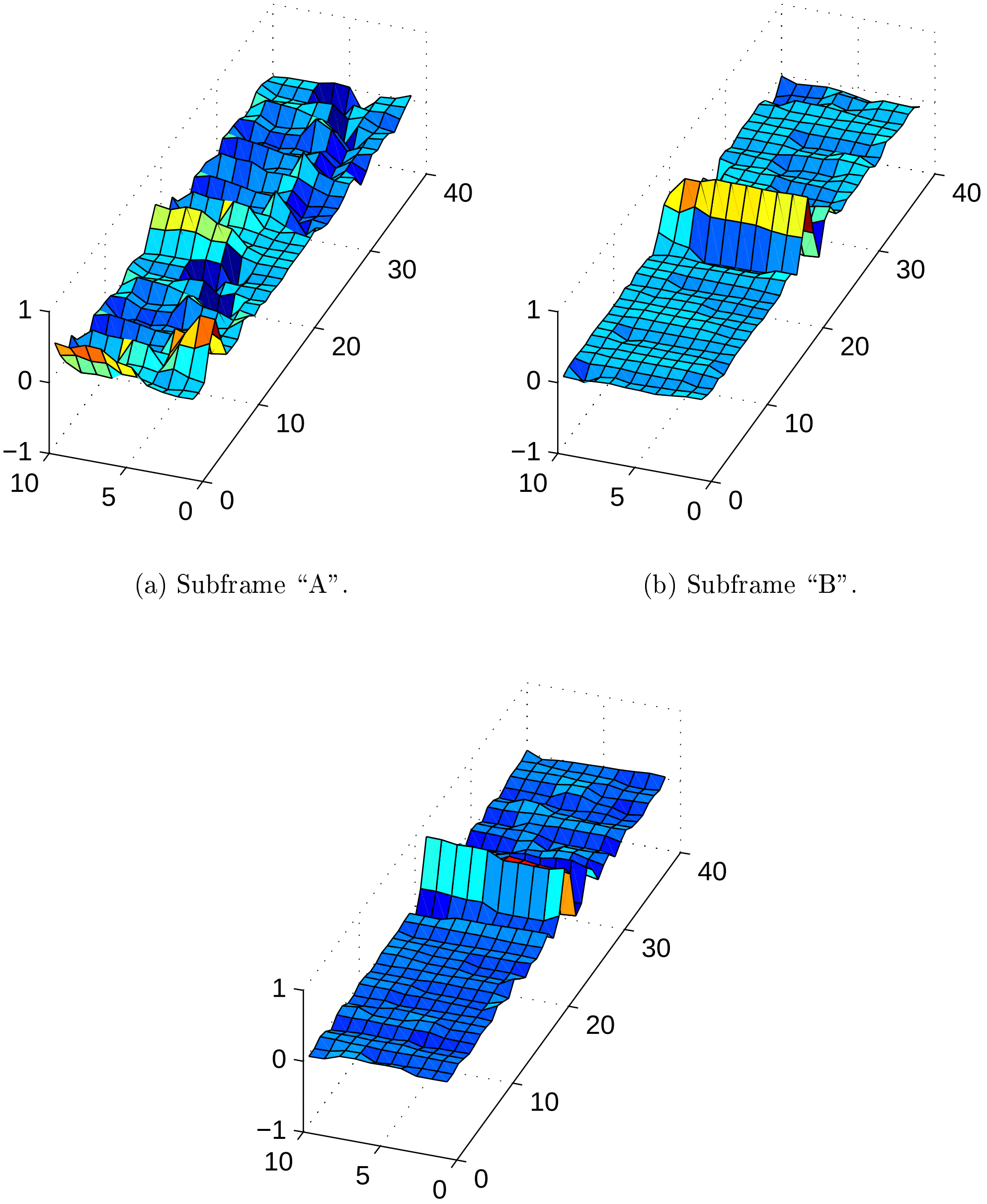
to normalize the energy of the extracted waveforms before the weighting takes place. This way, the pitch pulses in the extracted waveforms, will have more uniform amplitudes and thus the weighting will be more effective. This of course will affect the performance of the weighting function during the onsets because the noisy waveforms extracted from far in the past will also be amplified. To illustrate this effect, the extracted waveforms in Fig. 4.4 are normalized in energy and the result is shown in Fig. 4.10.

With this modification to the existing algorithm, the same file was processed and the results for $\alpha = 140$ are shown in Fig. 4.11. After the extracted waveforms have been normalized, the correlation coefficient has generally remained the same during steady state (compared to what it was before normalization), and improved during the end of voicing, as expected. At the onset, the average value of the correlation coefficient has slightly decreased.

The mean-squared error is plotted for the original and modified with normalized waveforms coder and shown in Fig. 4.12. The average MSE of the segment shown is lower than the original. During steady state its value is lower in the majority of subframes. Nevertheless, at the subframe indicated in a shaded region, the mean-squared error is significantly higher than originally. Even though the correlation coefficient is slightly lower than the original, the fixed codebook gain is significantly higher (see Fig. 4.11). But the increase or decrease in the fixed codebook gain does not provide any useful information on whether it results in a lower or higher mean-squared error than the original; i.e, there are cases where an increase in the fixed codebook gain resulted in a higher MSE and in other cases resulted in lower MSE.

To help investigate the cause for the increase in the mean-squared error in that subframe, the LP residual and constructed residuals for both the original and the modified with normalized waveforms coders are plotted in Fig. 4.13.

Letting the shaded subframe to be the current one, the original coder repeats the pitch pulse preceding the current subframe as the pitch period for those subframes is around 25 samples. This pitch pulse is indicated in Fig. 4.13(b) with the letter “I”. The modified coder with normalized waveforms extracts a number of waveforms from the past and averages them using a Kaiser window with $\alpha = 140$. The first extracted waveform is basically a repetition of the pitch pulse preceding the current subframe, indicated in Fig. 4.13(c) with the letter “II”. Since this waveform is emphasized most, it is expected that the averaged waveform will be similar to it. These observations are justified by looking at Fig. 4.14

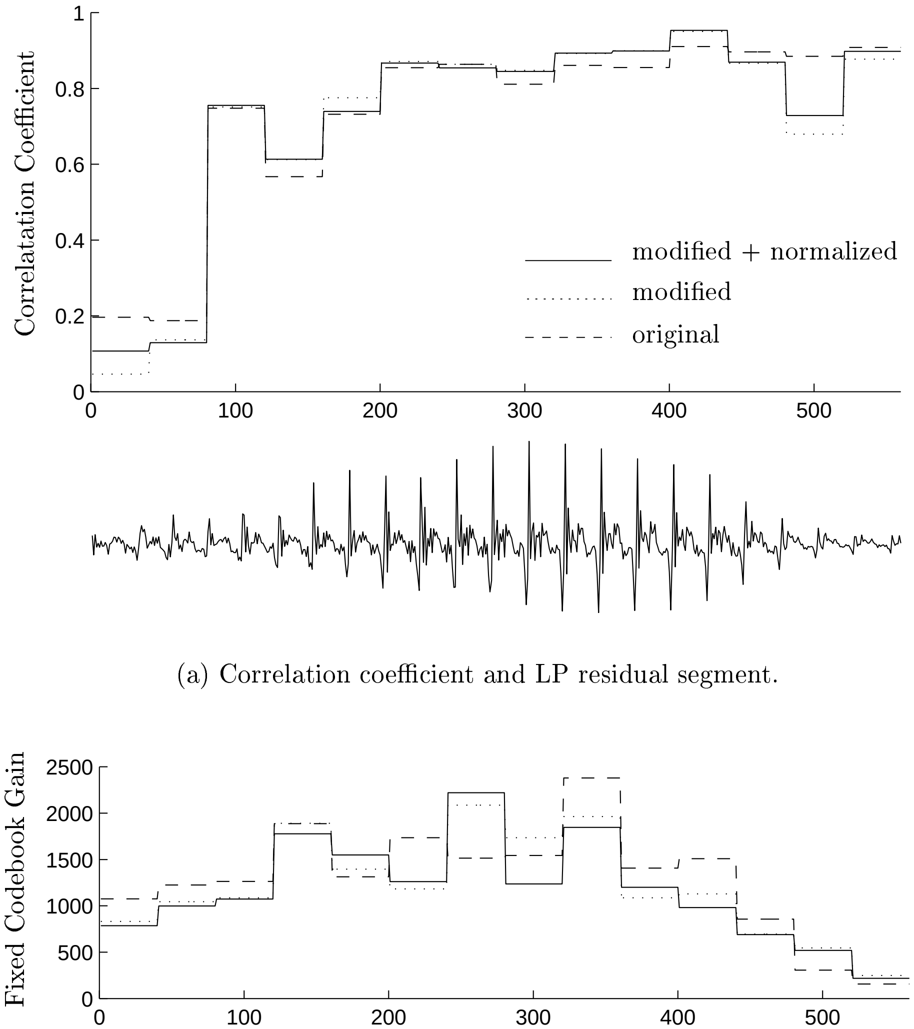


(a) Subframe “A”.

(b) Subframe “B”.

(c) Subframe “C”.

Fig. 4.10 Extracted waveforms normalized in energy.



(a) Correlation coefficient and LP residual segment.

(b) Fixed codebook gain.

Fig. 4.11 Results after the extracted waveforms have been normalized.

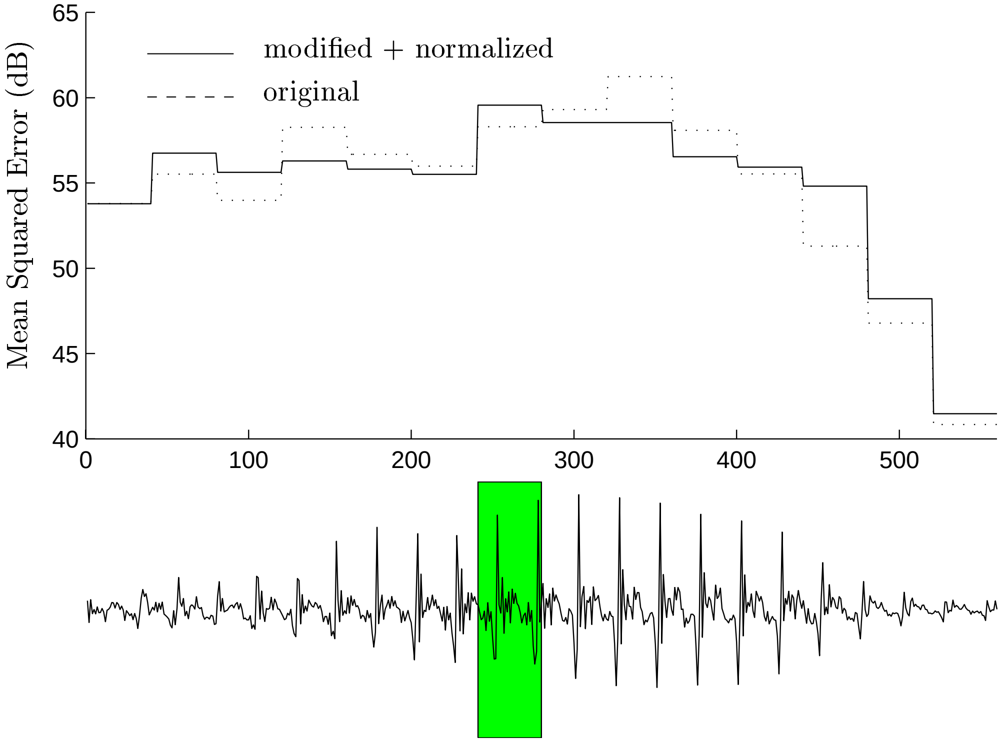
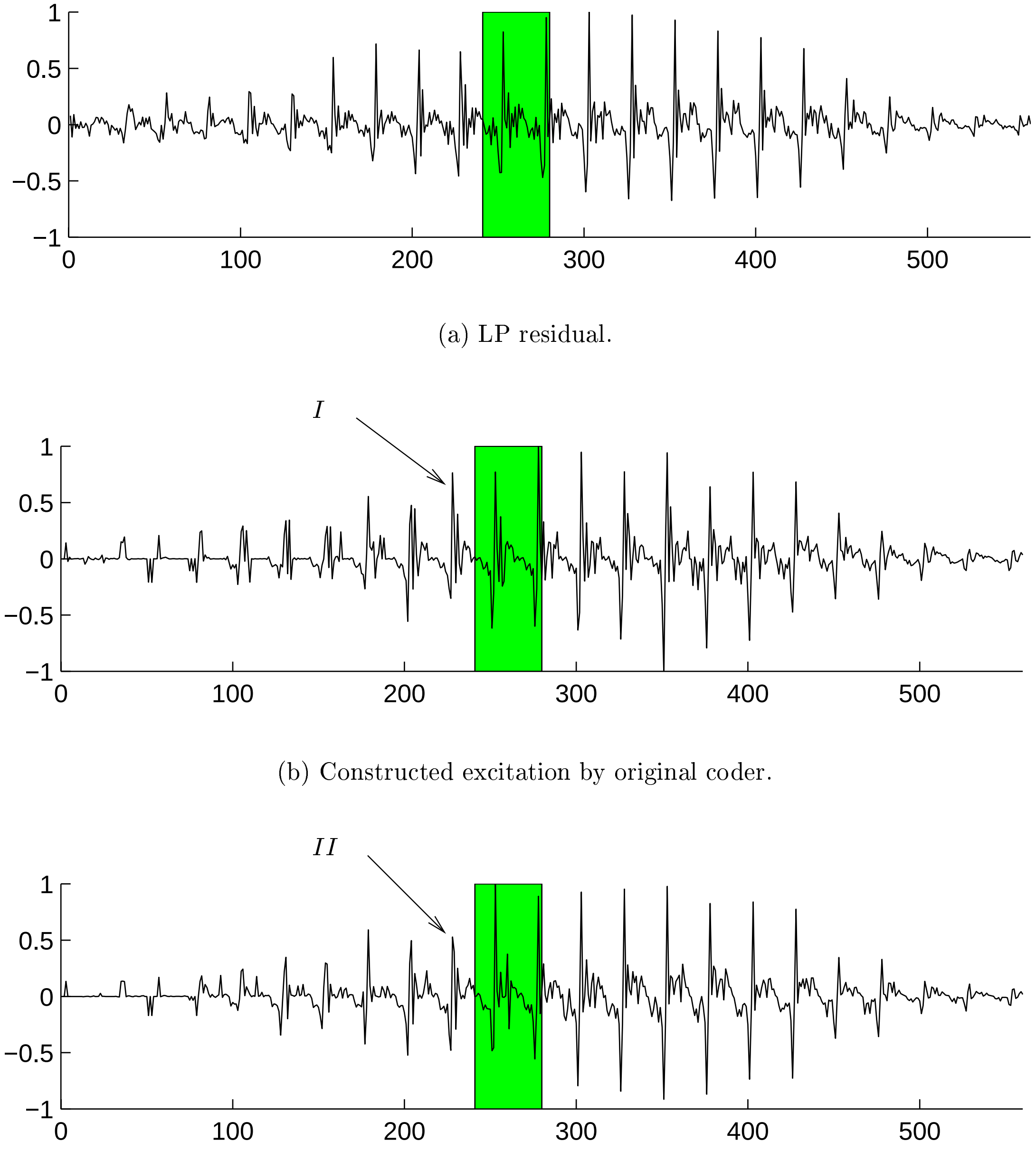


Fig. 4.12 Mean-Squared Error plot.

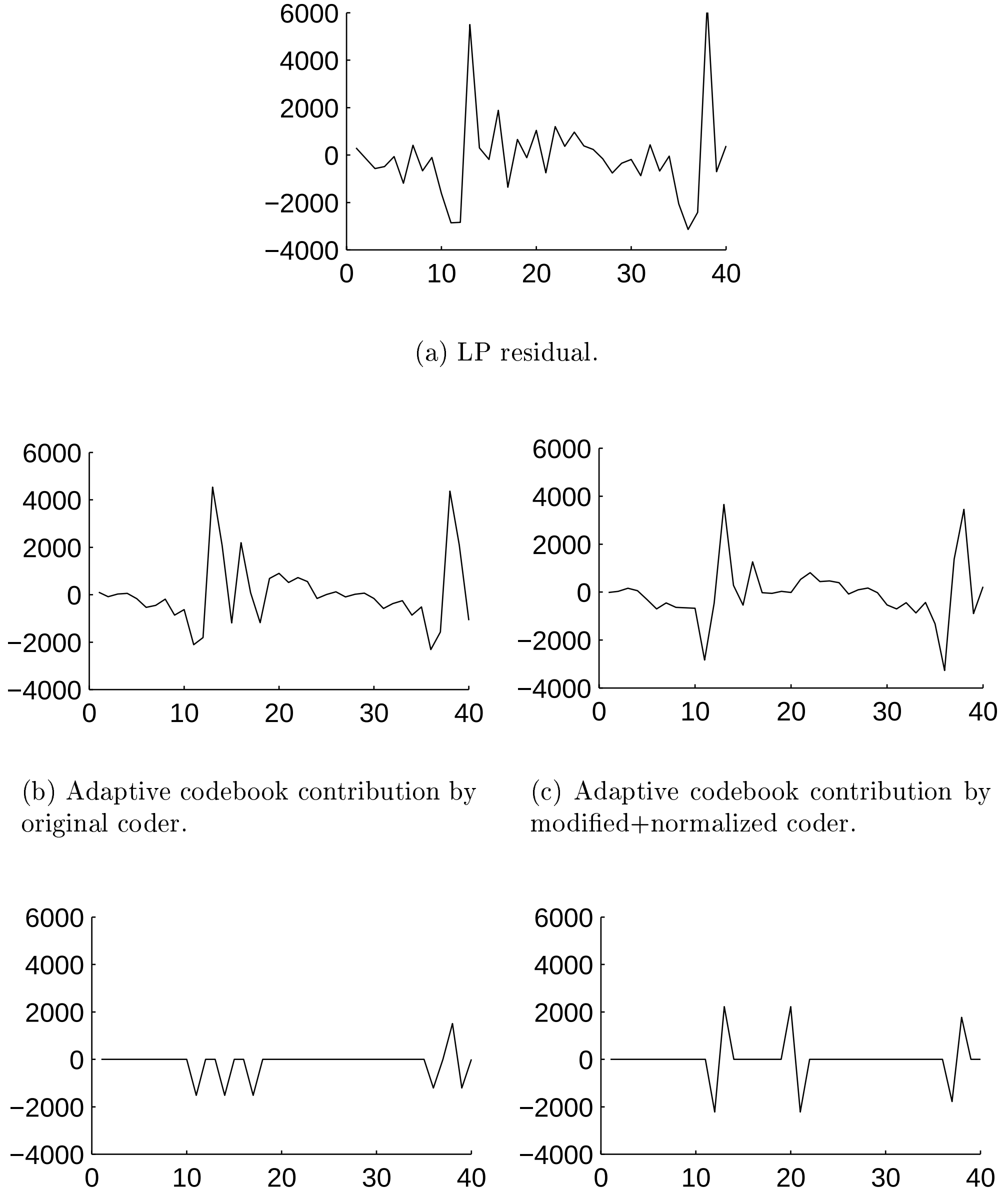


(a) LP residual.

(b) Constructed excitation by original coder.

(c) Constructed excitation by modified+normalized coder ($\alpha = 140$).

Fig. 4.13 Excitation signals.



(a) LP residual.

(b) Adaptive codebook contribution by original coder.

(c) Adaptive codebook contribution by modified+normalized coder.

(d) Fixed codebook contribution by original coder.

(e) Fixed codebook contribution by modified+normalized coder.

Fig. 4.14 Excitation contributions at subframe indicated in Fig. 4.13.

where the adaptive and fixed codebook contributions and LP residual, corresponding to this subframe, are plotted for both the original and the modified with normalized waveforms coders. The pitch pulses in the contribution by the original coder (Fig. 4.14(b)) are stronger that the ones in Fig. 4.14(c). Noting that the LP residual has stronger pitch pulses than both contributions, the original coder requires lower amplitude pulses in the fixed codebook contribution (shown in Fig. 4.14(d)) than the modified coder. This resulted in higher fixed codebook gain for the modified coder. The positions of the fixed codebook pulses shown in Figures 4.14(d) and 4.14(e) are chosen such that the weighted MSE of synthesized speech is minimized. However, it should be noted that the pitch synthesis filter placed after the fixed codebook (see Section 3.2) is incorporated in the search and the positions of the pulses are optimized accordingly. The result of using the pitch synthesis filter is to repeat some or all of the four available pulses. These additional pulses are seen in Figures 4.14(d) and 4.14(e) with lower amplitudes, due to the scaling factor (see Section 3.2). In both cases, the positions of most of the pulses are chosen so as to increase the negative and positive transitions of the pitch pulses. According to the LP residual, the second pitch pulse should have a higher positive amplitude than the first. In the case of the original coder, both pitch pulses have a relative large positive amplitude and therefore a positive pulse (in the fixed codebook vector) is only placed at the position of the second pitch pulse. This was not the case for the modified coder. As both pitch pulses had low amplitude, positive pulses had to placed at both positions. In order to minimize the error, a positive pulse was placed at the first pitch pulse position, but the second was left with one of the lower amplitude repeated pulses. The result was to increase the amplitude of the first pitch pulse more than the second, as shown in Fig. 4.13(c), and boost the MSE. This was not the case for the original coder whose resulting waveform is more similar to the LP residual (see Figures 4.13(a) and 4.13(b) respectively).

In conclusion, the MSE was higher for the modified coder because the pitch pulse in the previous subframe was not a good representation of the current. The algorithm should not be blamed for this though, as the correlation coefficient and MSE for the previous subframe were better than those of the original coder. This implies that the waveform constructed by the modified coder in the previous subframe was “better” (more similar to the LP residual) than that constructed by the original coder. This can as well be justified by comparing the two pitch pulses preceding the shaded subframe in Figs. 4.13(b) and 4.13(c) with those in Fig. 4.13(a). Therefore, the error that the original coder introduced in the previous subframe, accidentally ended up as an advantage to the next (shaded) subframe.

In order to make sure that the modified coder does not perform any worse for low pitch frequency speech, where the original coder in general performs well, similar tests have been carried out for an utterance spoken by a male speaker whose average pitch period is approximately 75 samples. For this purpose, the male voice speech file “pipm8.au” has been used as input to the coder. The results are shown in Figs. 4.15 and 4.16.

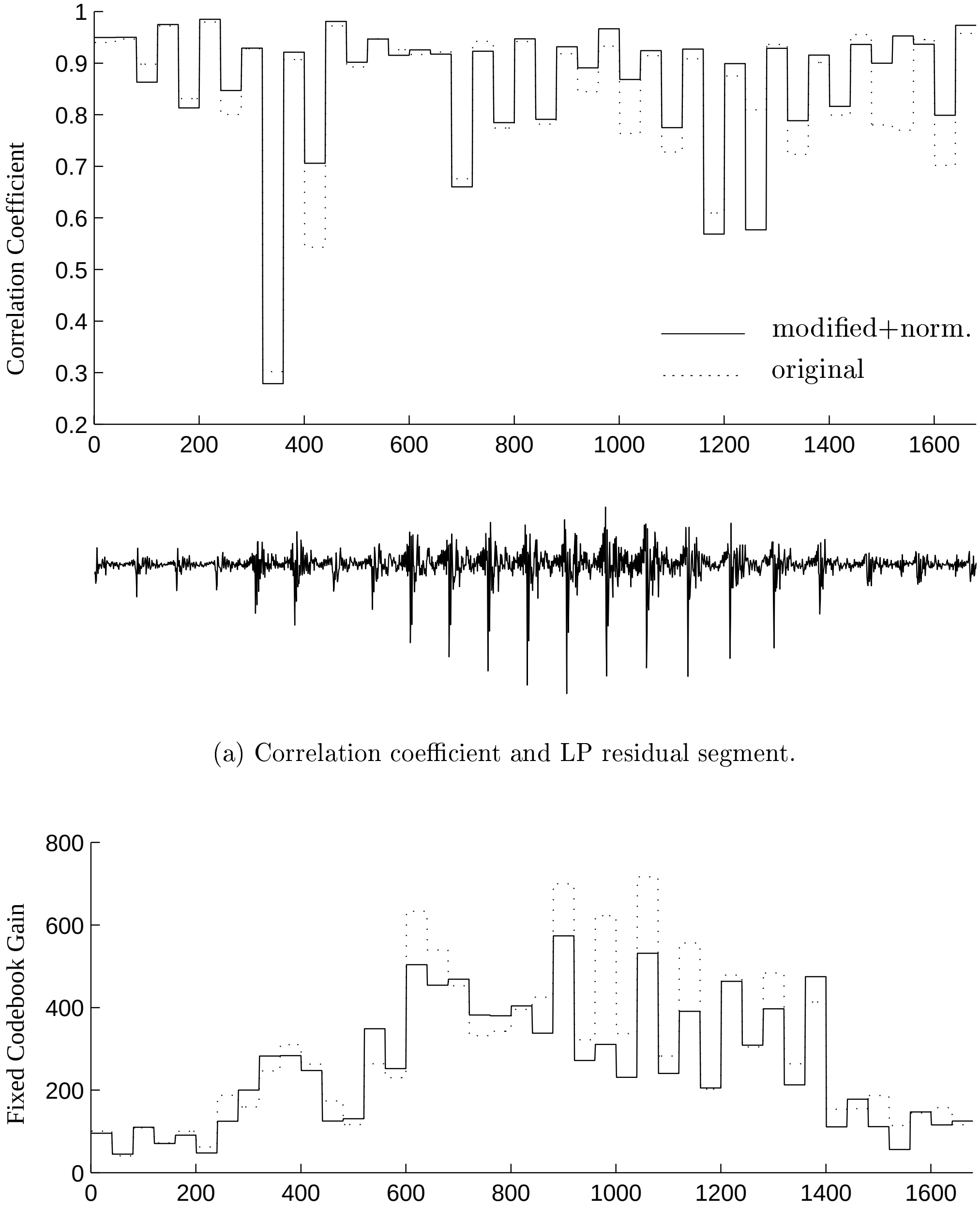
For the indicated segment, the correlation coefficient during steady state is higher than that of the original; the fixed codebook contribution and mean-squared error are generally smaller. This indicates that the adaptive codebook is making a larger contribution and the fixed codebook is making a smaller contribution. During onset and end of voicing, all the above parameters take values above and below the original values. Considering the values for the entire segment shown, in average, all the parameters have been improved.

The results for the speech segments shown, are tabulated in Table 4.1. Here, the values of the three parameters are averaged over the indicated subframes.

Table 4.1 Results for the presented voiced segments.

File	Range	Coder	$\overline{\rho(0)}$	$\overline{G}$	$\overline{MSE}$	$\overline{MSE}$
	(subframes)			(quantized)		improvement
xtest1.au	211–224	original	0.706	1298	459347	
		modified ($\alpha = 140$)	0.710	1168	414151	9.8%
pipm8.au	356–397	original	0.846	292	82572	
		modified ($\alpha = 140$)	0.862	258	77159	6.6%

So far, results have been provided for specific voiced segments of the aforementioned speech files. A more general view of the performance of the modified coder, can be obtained by calculating the MSE for all the voiced segments of those speech files. This will include different conditions at all voicing regions, that is, onsets, steady state and end of voicing. For comparison purposes, an estimate of the upper bound of the performance of the algorithm will be given. This can be obtained by switching between the modified and original coder based on which is best at each subframe. For best performance, the switching decision should be based on the usual MSE estimate for each case, but this is very costly in terms of computational complexity. A simpler way of implementing such a mechanism, is to make a decision based on which case produces the best adaptive codevector for the current subframe. The best adaptive codevector would be the one with the highest cross-



(a) Correlation coefficient and LP residual segment.

(b) Fixed codebook gain.

Fig. 4.15 Results for “pipm8.au” ($\alpha = 140$).

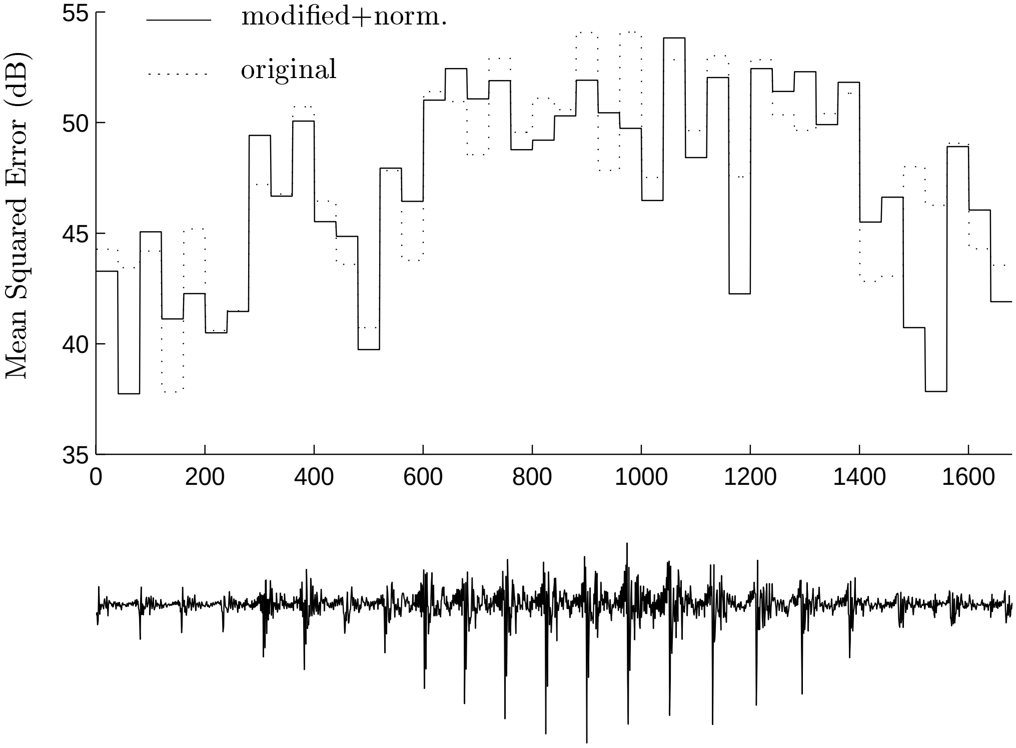


Fig. 4.16 Mean-Squared Error for “pipm8.au” ($\alpha = 140$).

correlation coefficient, between the filtered adaptive codevector and the target vector. Of course this decision mechanism cannot be implemented at the decoder, unless an extra bit is being transmitted to indicate which case should be employed at each subframe. This "best" adaptive codevector does not always result to the smallest weighted MSE, but it is still a very good assumption. This condition has been implemented and the results are summarized in Table 4.2. The parameter α has been tuned to minimize the MSE averaged either over all voiced subframes or over steady state subframes.

Table 4.2 Upper bound performance results.

File	Segments considered	α	$\overline{\rho(0)}$ (quantized)	$\overline{G}$	$\overline{MSE}$	$\overline{MSE}$ improvement
pipm8.au	voiced	100	0.823	448	201434	5.9%
	steady state	60	0.876	621	279794	8.2%

Before proceeding, recall the observation that the algorithm is doing better during steady state than during onsets and end of voicing. In addition to the explanation already provided, one more reason that the algorithm is doing worse at the onsets is the following. During onsets there is a rapid build-up of pitch pulses along with a comparatively smaller noisy segment (ringing) in between them. Since most of the energy is concentrated at the pitch pulses and the adaptive codebook contribution is very low, the fixed codebook is expected to concentrate most of its energy at those positions. The very limited number of 4 pulses available in G.729 for the fixed codevector and the fact that all four must have the same amplitude, renders the fixed codebook unable to rapidly compensate for the high amplitude pitch pulses and the ringing at the same time. Thus the noisy segment is being slowly compensated from subframe to subframe. When the algorithm is used to average those waveforms, the small rapid changes of the ringing are smoothed out; the degree of smoothing depending on the value of parameter α. Therefore, the attempt of the coder to slowly build up the noisy part is made even harder. This verifies the observation that, for best results, the best value of α is higher at the the onset and end of voicing than during steady state. Another way of viewing this condition is the following. During onsets, as well as unvoiced segments, the adaptive codebook acts as a second stochastic codebook since in both cases its contents do not have any form of periodicity. Any attempt to smooth out its contents during such segments, in general will weaken the ability of the coder to quickly compensate for the rapid changes in the excitation. The following experiment verifies the above observation. Using the file "pipm8.au" as input, the voiced subframes (includes onsets, steady state and end of voicing) and steady state subframes have been identified. Then the following scenarios have been tried:

(A) The algorithm was activated at all subframes.

(B) The algorithm was activated only during voiced subframes.

(C) The algorithm was activated only during steady state voiced subframes.

The results are summarized in Table 4.3.

Table 4.3 Results for all voiced segments.

Scenario	Segments considered	Coder	$\overline{\rho(0)}$ (quantized)	$\overline{G}$	$\overline{MSE}$	$\overline{MSE}$ improvement
	voiced	original	0.813	461	214169	
	steady state	original	0.867	638	304847	
A	voiced	modified ($\alpha = 150$)	0.811	459	219145	-2.3%
	steady state	modified ($\alpha = 150$)	0.859	648	311870	-2.3%
B	voiced	modified ($\alpha = 180$)	0.815	454	210534	1.7%
	steady state	modified ($\alpha = 180$)	0.871	636	296403	2.8%
C	voiced	modified ($\alpha = 110$)	0.813	452	205964	3.8%
	steady state	modified ($\alpha = 110$)	0.860	637	304189	0.2%

The results show that when the algorithm is activated during voiced subframes, the best value for α is higher than the case where the algorithm is activated only during steady state. This is due to the fact that at onsets and ends of voicing as well as during unvoiced segments the original coder is in action, which as already shown, performs better than the modified. Of course, as verified by experimental results, during those segments where the original coder is active, the MSE is not exactly the same as it would be if the original coder was active for the whole speech file. Once the modified coder is activated, the past residual signal now contains segments arising from averaged waveforms. Thus due to the recursive nature of the adaptive codebook, any consequent waveforms constructed by the original coder will not be identical as it would be if the original coder was active for the whole speech file.

In conclusion, the algorithm should be activated only during steady state voiced segments. Since our goal is not to alter the basic coding format, the method of switching between the modified and original coder (which delivers an upper bound on the performance of the technique) cannot be used since an extra bit will be required to be transmitted at each subframe. Therefore, the adaptation device should use resources that are available at both the encoder and decoder. A very simple way of predicting the voiced segments is by looking at the relative change in the pitch period of the current subframe to that of a number of past subframes, since it is nearly constant during voicing. The voicing onsets could be skipped by considering a number of past pitch lags. The larger this number is, the more pitch pulses are skipped during onsets. The voicing decision mechanism that was tried is given as follows:

$$Voiced = \begin{cases} 1 & \text{if } (|D^{(0)} - D^{(1)}| < \delta \wedge \cdots \wedge |D^{(0)} - D^{(M)}| < \delta) \\ 0 & \text{otherwise} \end{cases} , \tag{4.3}$$

where M is the number of past pitch lags to be considered and δ is an integer.

Since the coder does not allow any change in the pitch period larger than $4\frac{2}{3}$ or smaller than $5\frac{2}{3}$ within a frame, the value of δ is confined within these limits. Both M and δ have to be optimized for best results. Table 4.4 summarizes the results obtained for the file “pipm8.au”.

Table 4.4 Results for all voiced segments with adaptive voicing decision.

M	δ	α	Segments considered	$\overline{\rho(0)}$ (quantized)	$\overline{G}$	$\overline{MSE}$	$\overline{MSE}$ improvement
5	4	200	voiced	0.817	449	211364	1.3%
			steady state	0.875	614	282598	7.3%
3	3	90	voiced	0.808	474	209410	2.2%
			steady state	0.862	682	286926	5.9%

Compared to the results in scenario “C” of Table 4.3, the average MSE for all the voiced segments has slightly increased, but dropped considerably during the steady state. This result once again verifies the expectation that the algorithm performs better during steady state voicing. This is also verified from the upper bound performance results of Table 4.2, where the average MSE improvement is higher during steady state. The average

MSE has dropped for the overall voiced segments because even though the voicing decision mechanism accounts for voicing onsets, ends of voiced segments cannot be detected. Since the algorithm performs worse than the original during those segments, the overall average MSE has dropped.

4.3.3 Subjective tests

The ultimate goal of this work is to perceptually enhance the performance of the original coder. As the original coder is doing well for speech files with average pitch, we tried to identify segments of high pitch speech files where the synthesized speech was most degraded. Using the settings at the first row of Table 4.4, some informal listening tests were carried out on synthesized speech using different values of α. Unfortunately it was hard to identify any differences because there were no audible harshness during vowels.

Thus we artificially reduced the performance of the original coder to introduce more noise to the synthesized speech. This was achieved by reducing the number of pulses in the fixed codebook from 4 to 3. Indeed, the synthesized speech was then greatly degraded with an audible noisy character. After running the algorithm under this setting, no significant audible improvement has been observed in most high pitch sentences, except during a small segment of a file which contained a high pitch prolonged vowel, where the noise had been masked.

Chapter 5

Conclusions

In this thesis we presented the design and simulation of the Pitch Pulse Averaging (PPA) technique whose goal is to reduce the noise in the pitch pulse waveforms supplied by the adaptive codebook during steady state voiced speech segments. The algorithm was simulated and tested on a floating point version of the G.729 coder.

The simulations verified that the algorithm successfully extracts the pitch pulse waveforms from the past excitation and aligns them in order of precedence. This process is most useful when the pitch period is nearly constant for successive subframes. The extracted waveforms are averaged to reduce any large differences between them, thus effectively reducing the amount of noise in the final waveform. The amount of weighting is controlled by the independent parameter α. The smaller its value, the larger the weight given to the waveforms extracted from further in the past and thus more smooth the resulting waveform will be. Its value was tuned for best results, that is to reduce the weighted mean-squared error between the input and synthesized speech. The results showed that its value should be in the range of (100–200). In this range of values, only the first few extracted waveforms are weighted significantly. This implies that the first extracted waveform is very close to the best it can be under the current setting and there is very little space for improvement.

The algorithm does best during steady state voiced speech because the similarity between successive waveforms is higher during such segments. Any attempt by the fixed codebook to disturb the periodicity of such segments by inserting pulses of unwanted amplitudes to unnecessary positions, was alleviated by the averaging procedure. The best value of α during these segments is low, which implies that the fixed codebook is introducing unwanted noise to the final excitation waveform and there is more need to remove it by the averaging procedure. The situation is reversed during the onsets. In this case the algorithm slows down the attempt of the coder to compensate for the increasing amplitude pitch pulses and ringing at the same time, while having only 4 pulses available for the fixed codevector. Similarly, during unvoiced segments the adaptive codebook essentially acts as a second stochastic codebook, helping the fixed codebook compensate for the highly noisy character of such segments. The averaging of such segments, reduces the noisy character of the adaptive codebook, thus leaving more work to the fixed codebook. Thus for such segments the value of α is high and the overall performance of the algorithm drops below that of the original coder.

The problem of poorer performance during the onsets and unvoiced segments has been reduced by introducing a simple mechanism to activate the algorithm only during voiced segments, skipping the first few pitch pulses at the onsets. The problem at the ends of voiced segments has not been solved though, since there was no simple and efficient way to detect such segments. Nevertheless the overall performance of the algorithm has improved.

Best results could be obtained by switching between the PPA technique and the original coder depending on which of the two performs best at each subframe. Using a simple decision mechanism, the cost in terms of complexity would be nearly the same as if the algorithm was continuously active. But an extra bit per subframe should be transmitted to inform the decoder which case should be used at each subframe to be synthesized. For a 5 ms subframe and an 8 kHz input sampled speech, such a scheme would increase the coding rate by 200 bps.

Informal listening tests showed that the current configuration of the G.729 coder has been highly optimized and leaves not much space for speech quality improvement in the synthesized speech. The small number of pulses provided for the fixed codebook allows only a minor smoothing to be performed and thus a small improvement in the weighted mean-squared error was achieved which is not sufficient to induce a commentary audible enhancement. Nevertheless, a minor quality enhancement has been achieved when the rate of the coder has been artificially reduced by decreasing the number of pulses in the fixed codebook from 4 to 3. The next generation low bit-rate coders will operate at even lower bit-rates, that is at 4 kbps and below. At such rates it will be harder to achieve high quality. Our PPA technique along with a jointly optimized fixed codebook may be the key to providing the desired speech quality.

Bibliography

[1] B. S. Atal and M. R. Schroeder, “Stochastic coding of speech signals at very low bit rates,” *Proc. IEEE Int. Conf. Commun.* (Amsterdam), pp. 1610–1613, May 1984.

[2] M. R. Schroeder and B. S. Atal, “Code-excited linear predictive (CELP): High quality speech at very low bit rates,” *Proc. IEEE Int. Conf. on Acoustics, Speech, Signal Processing* (Tampa), pp. 937–940, Mar. 1985.

[3] J. P. Campbell, Jr., V. C. Welch, and T. E. Tremain, “An expandable error-protected 4800 bps CELP coder (U.S. Federal Standard 4800 bps voice coder),” *Proc. IEEE Int. Conf. on Acoustics, Speech, Signal Processing* (Glasgow), pp. 735–738, May 1989.

[4] J.-H. Chen, R. V. Cox, Y.-C. Lin, N. Jayant, and M. J. Melchner, “A low delay CELP coder for the CCITT 16 kb/s speech coding standard,” *IEEE J. Selected Areas Commun.*, vol. 10, pp. 830–849, June 1992.

[5] R. Salami, C. Laflamme, J.-P. Adoul, A. Kataoka, S. Hayashi, C. Lamblin, D. Massaloux, S. Proust, P. Kroon, and Y. Shoham, “Description of the proposed ITU-T 8-kb/s speech coding standard,” *Proc. IEEE Workshop on Speech Coding for Telecom.* (Annapolis), pp. 3–4, Sept. 1995.

[6] A. DeJaco, W. Gardner, P. Jacobs, and C. Lee, “QCELP: The North American CDMA digital cellular variable rate speech coding standard,” *Proc. IEEE Workshop on Speech Coding for Telecom.* (Sainte-Adèle, Québec), pp. 5–6, Oct. 1993.

[7] K. Mano, T. Moriya, S. Miki, H. Ohmuro, K. Ikeda, and J. Ikedo, “Design of a pitch synchronous innovation CELP coder for mobile communications,” *IEEE J. Selected Areas Commun.*, vol. 13, pp. 31–40, Jan. 1995.

[8] P. Kroon and F. Deprettere, “A class of analysis-by-synthesis predictive coders for high quality speech coding at rates between 4.8 and 16 kbits/s,” *IEEE J. Selected Areas Commun.*, vol. 6, pp. 353–363, Feb. 1988.

[9] P. Kroon and B. S. Atal, “Strategies for improving the performance of CELP coders at low bit rates,” *Proc. IEEE Int. Conf. on Acoustics, Speech, Signal Processing* (New York), pp. 152–154, Apr. 1988.

[10] I. A. Gerson and M. A. Jasiuk, "Vector sum excited linear prediction (VSELP) speech coding at 8 kbps," *Proc. IEEE Int. Conf. on Acoustics, Speech, Signal Processing* (Albuquerque), pp. 461–464, Apr. 1990.

[11] S. Wang and A. Gersho, "Improved excitation for phonetically-segmented VXC speech coding below 4 kb/s," *Proc. IEEE Globecom Conf.* (San Diego), pp. 946–950, Dec. 1990.

[12] Y. Shoham, "Constrained-stochastic excitation coding of speech at 4.8 kb/s," in *Advances in Speech Coding* (B. S. Atal, V. Cuperman, and A. Gersho, eds.), pp. 339–348, Kluwer Academic Publishers, 1991.

[13] T. Taniguchi, M. Johnston, and Y. Ohta, "Pitch-sharpening for perceptually improved CELP and the sparse-delta codebook for reduced computation," *Proc. IEEE Int. Conf. on Acoustics, Speech, Signal Processing* (Toronto), pp. 241–244, May 1991.

[14] I. A. Gerson and M. A. Jasiuk, "Techniques for improving the performance of CELP type speech coders," *Proc. IEEE Int. Conf. on Acoustics, Speech, Signal Processing* (Toronto), pp. 205–208, May 1991.

[15] Y. Asakawa, H. Sekine, M. Takashima, N. Ishikawa, T. Matsuda, T. Okamoto, and R. Muramatsu, "A 5.6 kb/s speech codec using a pulse codebook and improved viterbi decoding," *IEEE Int. Conf. on Acoustics, Speech, Signal Processing*, vol. 1, pp. I–277–I–280, 1994.

[16] D. O'Shaughnessy, *Speech Communication: Human and Machine.* Addison-Wesley Publishing Company, 1987.

[17] J. R. Deller Jr., J. G. Proakis, and J. H. L. Hansen, *Discrete-Time Processing of Speech Signal.* Macmillan, 1993.

[18] J. Makhoul, "Linear prediction: A tutorial review," *Proc. IEEE*, vol. 63, pp. 561–580, Apr. 1975.

[19] B. S. Atal and S. L. Hanauer, "Speech analysis and synthesis by linear prediction of the speech wave," *J. Acoustical Society of America*, vol. 50, pp. 637–655, Aug. 1971.

[20] L. R. Rabiner and R. W. Schafer, *Digital Processing of Speech Signals.* Prentice-Hall, 1978.

[21] B. S. Atal, "Predictive coding of speech at low bit rates," *IEEE Trans. Commun.*, vol. COM-30, pp. 600–614, Apr. 1982.

[22] R. P. Ramachandran and P. Kabal, “Stability and performance analysis of pitch filters in speech coders,” *IEEE Trans. Acoustics, Speech, Signal Processing*, vol. ASSP-35, pp. 937–946, July 1987.

[23] S. W. Lang and J. H. McClellan, “A simple proof of stability for all-pole linear prediction models,” *Proc. IEEE*, vol. 67, pp. 860–861, May 1979.

[24] R. P. Ramachandran and P. Kabal, “Pitch prediction filters in speech coding,” *IEEE Trans. Acoustics, Speech, Signal Processing*, vol. 37, pp. 467–477, Apr. 1989.

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

[26] P. Kroon and W. B. Kleijn, “Linear-prediction based analysis-by-synthesis coding,” in *Speech Coding and Synthesis* (W. B. Kleijn and K. K. Paliwal, eds.), pp. 79–115, Elsevier, 1995.

[27] B. S. Atal and J. R. Remde, “A new model of LPC excitation for producing natural-sounding speech at low bit rates,” *Proc. IEEE Int. Conf. on Acoustics, Speech, Signal Processing* (Paris), pp. 614–617, May 1982.

[28] P. Kroon, E. F. Deprettere, and R. J. Sluyter, “Regular-Pulse Excitation: A novel approach to effective and efficient multi-pulse coding of speech,” *IEEE Trans. Acoustics, Speech, Signal Processing*, vol. ASSP-34, pp. 1054–1063, Oct. 1986.

[29] A. Gersho and R. M. Gray, *Vector Quantization and Signal Compression.* Kluwer Academic Publishers, 1992.

[30] P. Hedelin, P. Knagenhjelm, and M. Skoglund, “Vector quantization for speech transmission,” in *Speech Coding and Synthesis* (W. B. Kleijn and K. K. Paliwal, eds.), pp. 311–346, Elsevier, 1995.

[31] J.-L. Moncet and P. Kabal, “Codeword selection for CELP coders,” tech. rep., INRS-Telecommunications, Montreal, Canada, July 1987.

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

[33] W. B. Kleijn, D. J. Krasinski, and R. H. Ketchum, “Improved speech quality and efficient vector quantization in SELP,” *Proc. IEEE Int. Conf. on Acoustics, Speech, Signal Processing* (New York), pp. 155–158, Apr. 1988.

[34] P. Kroon and B. S. Atal, "Pitch predictors with high temporal resolution," *Proc. IEEE Int. Conf. on Acoustics, Speech, Signal Processing* (Albuquerque), pp. 661–664, Apr. 1990.

[35] J. S. Marques, I. M. Trancoso, J. M. Tribolet, and L. B. Almeida, "Improved pitch prediction with fractional delays in CELP coding," *Proc. IEEE Int. Conf. on Acoustics, Speech, Signal Processing* (Albuquerque), pp. 665–668, Apr. 1990.

[36] T. I. Laasko, V. Valimaki, M. Karjalainen, and U. K. Laine, "Splitting the unit delay," *IEEE Signal Processing Magazine*, pp. 30–60, Jan. 1996.

[37] R. Crochiere and L. Rabiner, *Multirate digital signal processing.* Prentice-Hall, 1983.

[38] J. G. Proakis and D. G. Manolakis, *Digital Signal Processing: principles, algorithms and applications.* Macmillan Publishing Company, 1992.

[39] W. B. Kleijn and J. Haagen, "Waveform interpolation for coding and synthesis," in *Speech Coding and Synthesis* (W. B. Kleijn and K. K. Paliwal, eds.), pp. 175–208, Elsevier, 1995.

[40] J. Stachurski, *A Pitch Pulse Evolution Model for Linear Predictive Coding of Speech.* PhD thesis, McGill University, Montreal, Canada, May 1996.