TIME-DOMAIN CODING OF (NEAR) TOLL QUALITY SPEECH AT RATES BELOW 16 KB/S

Peter Kroon

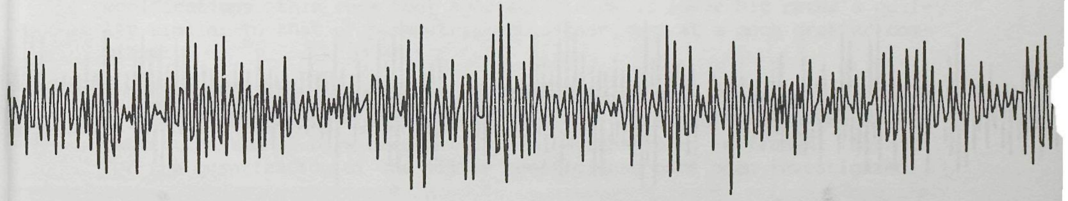
TIME-DOMAIN CODING OF (NEAR) TOLL QUALITY SPEECH AT RATES BELOW 16 KB/S

Proefschrift

Ter verkrijging van de graad van doctor in de technische wetenschappen aan de
Technische Hogeschool Delft, op gezag van de
Rector Magnificus, prof.dr. J.M. Dirken,
in het openbaar te verdedigen ten overstaan van het College van Dekanen op
dinsdag 21 mei 1985 te 14.00 uur

door

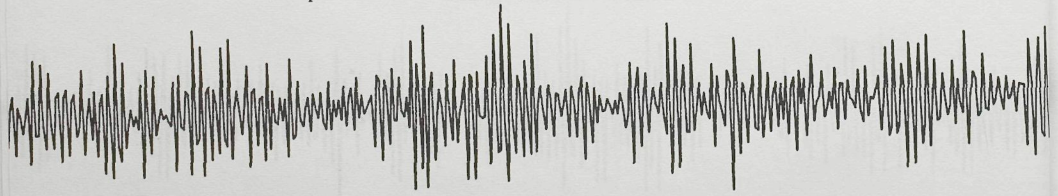
PETER KROON
geboren te Vlaardingen
elektrotechnisch ingenieur



Dit proefschrift is goedgekeurd door de promotor:
prof.dr.ir. P. Dewilde

Dr.ir. Ed. F. Deprettere heeft in hoge mate bijgedragen aan de begeleiding bij het totstandkomen van dit proefschrift en is als zodanig door het College van Dekanen aangewezen.

This work has been partially supported by
Philips Research Laboratories, Eindhoven, The Netherlands.



TIME-DOMAIN CODING OF (NEAR) TOLL QUALITY SPEECH AT RATES BELOW 16 KB/S

Peter Kroon

Delft University of Technology,
Mekelweg 4, 2628 CD Delft,
The Netherlands

ABSTRACT

The use of digital techniques for organizing communication networks and for representing signals results in systems which are superior to their analog counterparts in terms of quality and reliability. During the last decade many interesting speech coding algorithms have been proposed, but only the advent of fast and economical VLSI components has made it practical to implement speech coding techniques in real-time systems.

In this thesis a study of efficient time domain algorithms for the encoding of speech signals with a (near) telephone quality speech at bit rates below 16 kb/s is presented. A Delayed Decision Coding (DDC) system using adaptive predictive techniques, which removes the correlation in the speech signal, is used as the basic structure. Procedures for determining the excitation sequence within such a DDC structure are investigated.

One such procedure, called Multi-Pulse Excitation (MPE) coding [Atal and Remde, 1982], is studied in detail and techniques for improving the performance of this coder are described.

In the course of the work a new coding concept called Regular-Pulse Excitation (RPE) coding was developed and it is demonstrated that this technique produces speech with a quality comparable to existing methods such as the MPE coder, but with a much lower complexity.

To provide a reference to the parametric approaches (MPE and RPE), we describe the results of simulations with a code book approach for finding the optimal excitation sequences. We demonstrate that with some modifications this code book approach yields at lower bit rates a quality similar to that of parametric approaches, but at a much greater complexity.

Quantization procedures for the coder parameters are described, and efficient procedures for encoding the pulse positions of the multi-pulse excitation signal have been developed. Furthermore, different methods for the quantization of the filter coefficients have been investigated.

An investigation of the RPE and MPE coders demonstrated that both coders provide (near) toll quality speech at 10 kb/s. Coder transparency can be obtained at 16 kb/s, while with the use of vector quantization techniques an adequate performance is obtained at 6 kb/s. Further, it is demonstrated that both coders can be successfully applied for the encoding of wide-band speech signals at rates below 32 kb/s.

An efficient realization of the proposed RPE coding scheme is obtained by mapping the algorithm onto silicon with the use of CORDIC processor elements as basic building blocks.

Finally, to provide a suitable environment for the investigation of speech coding algorithms, we describe a convenient interactive software package, which we have developed in the course of the work.

1. PREFACE

Thesis Objectives

Although speech research has been a continuing effort since the early nineteen forties, it was not until the beginning of the nineteen eighties that new coding concepts that enable the encoding of (near) toll-quality speech at rates below 16 kb/s were introduced. An interesting concept called Multi-Pulse Excitation (MPE) coding was proposed in [Atal and Remde, 1982]. This introductory paper contained a good description of the coding principles but lacked many implementation details. Our first objective was to investigate the method and evaluate its performance. Impressed by the initial results we investigated modifications of the coder to improve the performance and to increase the efficiency. The use of a pitch predictor provided the most dramatic improvement [Kroon and Deprettere, 1984]. To reduce the complexity of the coder several alternatives were investigated, such as reduction of the search frame size [Kroon and Deprettere, 1983]. Despite the encouraging results, we felt that the complexity of the MPE coder was too high, and that the encoding algorithm was not very suitable for VLSI implementation. This has led to the development of a new coding concept called Regular-Pulse Excitation (RPE) coding [Deprettere and Kroon, 1985], which provides a performance equal to that of an MPE coder but has a structure more amenable to VLSI implementations. In [Deprettere and Jainandunsing, 1985] the encoding procedure was mapped as a regular VLSI structure using CORDIC modules. The two coders investigated have the same underlying structure, and we recognized the fact that the stochastic coder [Atal and Schroeder, 1984] has a similar structure and differs only in the definition of the excitation function and the corresponding search procedure. Since one of our objectives was to compare the different approaches, we investigate to a certain extent this coding concept as well. Another objective of this thesis was to provide procedures for quantization and encoding the coder parameters and their effect on the coder performance. Finally, as an important side effect, we established a software environment for the development and the evaluation of the coders [Kroon, 1983] [Kroon, 1983].

Thesis Outline

Chapter 1 introduces the reader to the most important speech coding techniques, and serves as a general introduction to the other chapters of this thesis.

In Chapter 2 we define the basic structure of the investigated coders. The effect of the short-time and long-time predictors on the coder performance is evaluated and the error weighting procedures are introduced.

Chapter 3 discusses a new coding concept called Regular-Pulse Excitation. This concept is based on a parametric approach to finding the optimum excitation sequence, and provides a computationally efficient procedure. The quality of the synthetic speech is close to toll quality at bit rates around 10 kb/s.

The Multi-Pulse Excitation coder, as proposed by Atal and Remde, is thoroughly investigated in Chapter 4. We describe modifications to the basic MPE analysis procedures and their resulting impact on the synthetic speech quality and the complexity of the coder.

Instead of using a parametric approach for the definition of the excitation sequence we can use a search procedure to find the excitation sequence directly. This procedure is evaluated in Chapter 5 and is intended as a reference for the performance of the coders described in the previous chapters.

The parametric approach requires a separate quantization and encoding stage, and in Chapter 6 different procedures for encoding and quantization will be described.

A comparison of the different coding concepts and the performance at various bit rates is discussed in Chapter 7. Application of the RPE coder to the encoding of wide band speech is also discussed. Finally, we describe some experiments with degraded speech signals, and non-speech signals.

Efficient structures for the RPE coder will be described in Chapter 8 and we discuss the possibility of mapping the coder onto silicon.

The software environment used for the development and the evaluation of the coders is described in Chapter 9.

References

Atal, B.S. and J.R. Remde, "A new model of LPC excitation for producing natural-sounding speech at low bit rates," *Proc. IEEE Int. Conf. Acoust., Speech and Signal Processing*, pp. 614-617 (April 1982).

Atal, B.S. and M.R. Schroeder, "Stochastic coding of speech signals at very low bit rates," *Proc. IEEE Int. Conf. Communications*, p. 48.1 (May 1984).

Deprettere, Ed. F. and P. Kroon, "Regular excitation reduction for effective and efficient LP-coding of speech," *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing*, pp. 25.8.1-25.8.4 (March 1985).

Deprettere, Ed.F. and K. Jainandunsing, "Design and VLSI implementation of a concurrent solver for N-coupled least-squares fitting problems," *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing*, pp. 6.3.1-6.3.4 (March 1985).

Kroon, P. and E.F. Deprettere, "On the design of LPC-vocoders with multi-pulse excitation," *Proc. European Conf. Circuit Theory and Design*, pp. 390-394 (Sep. 1983).

Kroon, P., "SPPACK User's Guide," Report NT-37, Dept. of EE, Delft University of Technology, Delft, The Netherlands (April 1983).

Kroon, P., "SPPACK Programmer's Guide," Report NT-38, Dept. of EE, Delft University of Technology, Delft, The Netherlands (April 1983).

Kroon, P. and E.F. Deprettere, "Experimental evaluation of different approaches to the multi-pulse coder," *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing*, pp. 10.4.1-10.4.4 (March 1984).

CONTENTS

1. OVERVIEW OF SPEECH CODING

1.1 Introduction

For many years it has been well known that communication systems could benefit from utilization of digital techniques. Usage of these techniques for network organization as well as signal representation results in systems that are superior to their analog counterparts, in terms of quality and reliability. A digital representation of signals offers many advantages such as low susceptibility to interference, applicability of digital encryption, and solid state storage facilities. Furthermore, digitization of transmission functions allows an integration of data, voice and graphics in a common communication concept. Despite all these advantages of the digital approach, wide-scale utilizations had to wait for the advent of fast and economical digital hardware. With recent developments in the design of integrated circuits and the appearance of signal processing chips, it has become practical to implement many digital signal processing algorithms in real-time systems.

The importance of speech signals in communications and the underlying goals of transmitting speech with the highest possible quality, over the least possible channel capacity, and with the least cost, are responsible for the increasing interest in speech coding research.

In the remainder of this section, we review the principles of speech coding and the most important techniques.

1.2 Basic Properties Of Speech

The mechanism of speech production [Flanagan, 1972] is as follows: sound is generated either by vibration of the vocal cords or by creation of turbulent air flow at a constriction. This excitation is then spectrally shaped by the vocal tract which consists of the pharynx and the oral cavity. Sometimes, the nasal tract is also involved. This process can be modeled as a slowly time-varying linear system that is excited by either a train of (glottal) pulses or random noise (Fig. 1.1). Note that this model assumes that source features are separable from the vocal tract features.

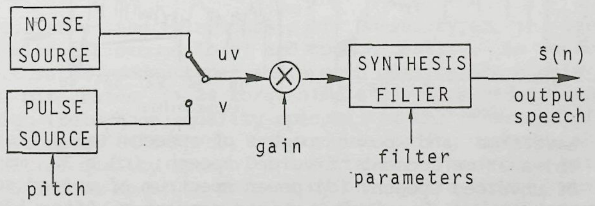


Figure 1.1. Model of speech production.

Speech sounds can be divided into three classes according to their mode of excitation:

1. Voiced sounds (vowels and voiced consonants) are produced by a quasi- periodic vibration of the vocal cords. The frequency of these vibrations determines the pitch of the sound.

2. Fricatives or unvoiced sounds (e.g. ss,sh,f) are generated by a turbulent flow of air at a constriction at some point in the vocal tract. Sounds that are produced by both a constriction in the vocal tract and a vibration of the vocal folds are called voiced fricatives (e.g. z,zh,v).

3. Plosives which are a result of making a closure of the vocal tract, building up the pressure behind the closure and releasing it abruptly. A plosive can be either voiced (e.g. b,d,g) or unvoiced (e.g. p,t,k).

An important property of speech waveforms is that they are band limited (to approximately 8 kHz) due to their specific production process. A further bandwidth reduction (e.g. 3400 Hz in telephony systems) is possible without too much degradation. The finite bandwidth implies that the speech waveform can be sampled at a finite rate (e.g 8000 Hz in telephony systems).

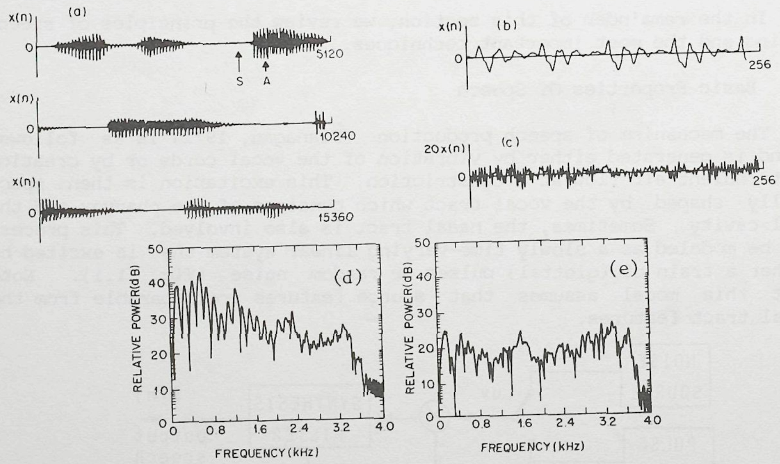


Figure 1.2. Waveforms and power spectra of speech; (a) a 2 s segment; (b) a 32 ms segment of voiced speech; (c) a 32 ms segment of unvoiced speech; (d) power spectrum of voiced segment b; (e) power spectrum of unvoiced segment c. After [Jayant and Noll, 1984].

When viewed over a long period of time (e.g. 1 s) speech is seen to be a highly non-stationary process due to the variation of amplitude, voiced/unvoiced, silence behavior, and the time varying vocal tract (Fig. 1.2). However, over short periods of time (10-40 ms) speech is locally stationary and has a well-defined short-time spectrum. The envelope of the spectrum is determined primarily by the frequency response of the vocal tract filter. The resonances can be seen as peaks in the spectral envelope and are referred to as formants. Other properties of the speech waveform are revealed by the short-time statistics such as amplitude distribution and auto-correlation function. The probability density function of speech amplitudes is estimated by determining a histogram for a large number of samples. A good approximation to the measured amplitude distribution was found [Paez and Glisson, 1972] [Noll, 1974] to be a Laplace distribution or a gamma distribution. A simple Gaussian distribution is usually adequate for a short-time (20 ms) distribution. The correlations that exist among the amplitude samples of a speech waveform are measured by an auto-correlation function. The correlation between adjacent samples (8 kHz) is usually high. For voiced segments we could also observe a correlation between samples spaced a pitch period apart due to the quasi-periodic excitation.

Concluding, we could say that speech waveforms exhibit a considerable redundancy that could be used to advantage for designing digital coding algorithms.

1.3 Classes Of Speech Coders

Designing a speech coding system involves a tradeoff among factors such as signal quality, transmission bit rate and robustness against environmental impairments and coder complexity (cost). These factors are mutually dependent. Signal quality improves for increasing bit rates, provided the optimum coding method for that particular bit rate is chosen. To maintain, as much as possible, signal quality at decreasing bit rates, coder complexity will increase.

Speech coding methods may be classified into different types depending on the speech properties and models that are utilized to obtain coding efficiency. We distinguish three types: waveform coding, vocoding (or source coding) and hybrid coding.

Fig. 1.3 shows the relationship of these types in terms of the number of bits per second (b/s) and speech quality. As we see from this figure, bit rates, coder types and speech quality are highly correlated. For high bit rates (> 24 kb/s) simple waveform coders can be used to produce high (telephone) quality speech. At the other end of the scale, the low bit range (< 5 kb/s), only source coding methods, which produce low (synthetic) quality speech, are suitable. The medium bit rate range (between 5 and 24 kb/s) is covered by the hybrid coders, which can produce medium (communication) to high (telephone) quality speech. A short description of each type is given below, with a more detailed discussion in the rest of this chapter.

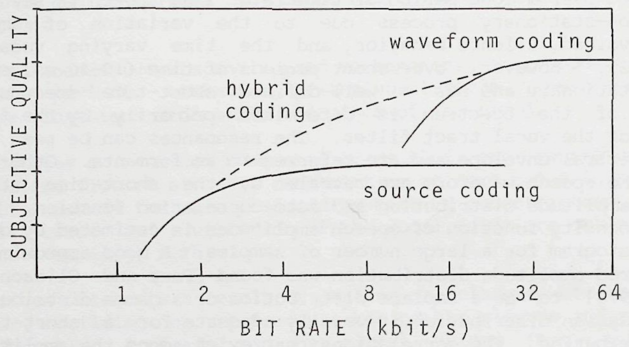


Figure 1.3. Subjective quality versus bit rate for speech coding.

Source Coders (Vocoders). Source coders use a priori knowledge about how the signal was generated at the source. This knowledge is applied to model the source while signal coding is accomplished by coding the model parameters. Source coders for speech, referred to as vocoders, use the previously described speech production model represented by excitation parameters and vocal tract parameters. Due to the simplicity of this model, especially of the excitation part, vocoders produce a synthetic quality that cannot be improved by simply increasing the bit rate.

Waveform coders. Waveform coders do not use a priori knowledge about the signal generation process. The objective of waveform coding is to make the reconstructed signal be as close to the original signal as possible. In principle, they are designed to be signal independent; hence they can code equally well a variety of signals. They also tend to be robust for a wide range of speaker characteristics and for noisy environments. By exploiting more signal specific characteristics, such as amplitude nonstationarity and static and dynamic spectral characteristics a greater coding efficiency can be obtained.

Hybrid coders. Hybrid coders form the link between waveform coders and source coders. At the intended bit rates it is no longer possible to have the reconstructed signal match the input signal on a sample-by-sample basis as in waveform coding. On the other hand, the maximum attainable quality of source coders is too low. Hybrid coders form a class in which the design criterion is neither waveform preservation nor signal modeling. Rather, we require that the output speech have a quality similar to the input signal on a subjective basis. Therefore, it is necessary to take advantage of certain properties of human speech perception. Moreover, waveform coding techniques and source coding techniques can be combined.

For telephone (toll) quality speech coding at rates below 16 kb/s, hybrid coders are going to play an important role in the coming years. In the following we describe current examples of the three categories with an emphasis on waveform and hybrid coders.

1.4 Source Coding (Vocoders)

Vocoders depend on a parameterization of the speech signal in accordance with the linear, quasi-stationary model of speech production as described earlier. The sound source is either a sequence of pulses separated by a pitch period for voiced sounds or a white noise source for unvoiced sounds. The vocal tract is modeled by a time-varying linear filter. The analysis procedure has to estimate the excitation and the vocal tract parameters. The parameters obtained are quantized and transmitted. At the receiver, the pitch, gain and voicing parameters are used to generate an excitation signal which excites the synthesis filter. In our model the source and system (vocal tract) are independent and could be estimated separately.

1.4.1 *Excitation Estimation*

Two main problems exist in the process of excitation estimation: first, the discrimination between voiced and unvoiced segments [Atal and Rabiner, 1976] and, second, the determination of the fundamental frequency for voiced segments. Two basic approaches to tackling these problems are time domain methods [Gold and Rabiner, 1969], which operate directly on the waveform, and frequency domain [Sluyter et al., 1980] [Sluyter, 1982] methods, which are based on spectral analysis.

1.4.2 *Filter (Vocal Tract) Estimation*

The vocal tract which is responsible for the spectral envelope of the speech segments can be represented in many ways. Similar to the excitation estimation, we could use time domain approaches and frequency domain approaches. Examples of time domain approaches are the autocorrelation vocoder [Huggins, 1954], where samples of the autocorrelation function of the speech segment are used, or the Linear Predictive Coder (LPC) [Atal and Hanauer, 1971] [Markel and Gray, 1976], which uses linear prediction coefficients that describe the spectral envelope. Frequency domain approaches are the formant vocoder [Klatt, 1980], which uses the frequency values of major spectral resonances and the channel vocoder [Gold and Rader, 1967] and its digital counterpart, the DFT-vocoder [Bosscha and Sluyter, 1982], which uses the values of the short-time amplitude spectrum of the speech signal evaluated at specific frequencies.

1.4.3 *Very-Low Bit Rate Vocoding Techniques*

The number of bits required for transmission of the vocoder parameters depends on different factors, such as the update rate of the filter coefficients and the method used for parameter quantization. For example, the U.S. government standard LPC vocoder (LPC-10) [Tremain, 1982] operates at a 2400 b/s transmission rate. The bit rate can be further decreased by transmitting parameters only when they have changed sufficiently from the previous transmitted parameters. At the receiver, the parameters are interpolated linearly and the synthesis filter coefficients are updated regularly. With this variable frame rate (VFR) technique [Viswanathan et al., 1982], the transmission rate can be reduced from 2400 b/s to about 1600 b/s with no loss in speech quality. Further bit rate reduction can be obtained by using more sophisticated quantization techniques such as vector quantization [Buzo et al., 1980]. In that method, all the linear prediction coefficients that represent an input speech spectrum are considered as a vector and quantized as a single unit. Using vector quantization, one can reduce the bit rate to 800 b/s [Wong et al., 1982]. By considering a number of consecutive frames as a vector, we obtain a segment vector quantizer [Roucos et al., 1982], which can transmit intelligible speech at 220 b/s when optimized for a single speaker.

1.5 Waveform Coding (Time Domain Methods)

1.5.1 *Pulse Code Modulation*

A simple form of waveform coding is Pulse Code Modulation (PCM). In this method the analog signal is sampled and uniformly quantized into one of $N = 2^R$ levels (R bits/sample). If a signal bandwidth of W Hz is assumed, the total bit rate is then $2WR$ b/s. Two common characteristics for a uniform quantizer with step size Δ and input range $V = x_N - x_0$ are shown in Fig. 1.4.

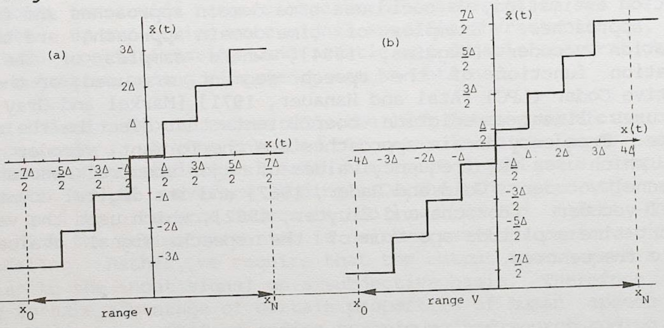


Figure 1.4. Two common uniform 3-bit quantizer characteristics (a) mid-tread, (b) mid-riser.

The mid-tread or rounding characteristic produces zero output for input samples that are in the neighborhood of zero. The mid-riser or truncation characteristic has a decision level located at zero, and maps zero valued input samples into non-zero output samples. When the input sample x falls outside the decision levels x_0 and x_N the quantizer is said to be overloaded. To avoid significant overload or saturation noise the range V is chosen to be a suitable multiple of the rms signal level σ. This multiple is called the loading factor. A large range, however, increases granular noise, while a small range reduces granular noise at the expense of saturation noise. For an optimal performance of the uniform quantizer the step size Δ and the range V have to be aligned to the input signal characteristics. In practice, alignment of the quantizer input is realized by one of two techniques: non-uniform or adaptive quantization.

1.5.1.1 *Non-Uniform Quantization*

A non-uniform quantizer has a small step size for small signal values and a large step size for large signal values. Given the probability density function (pdf) of the input signal, a non-uniform quantizer that minimizes the quantization noise for a given number of bits R can be designed. The design procedure for such a quantizer was independently proposed in [Lloyd, 1957] and in [Max, 1960], and this optimum quantizer is commonly referred to as a Lloyd-Max quantizer. Max tabulated the optimum uniform and non-uniform quantizer characteristics for Gaussian distribution functions. In [Paez and Glisson, 1972] [Adams and Giesler, 1978] the Lloyd-Max design procedure was used to generate similar characteristics for Laplace and gamma distributions.

A disadvantage of such an pdf optimized quantizer is its sensitivity to a mismatch between actual pdf and design pdf [Mauersberger, 1979]. Logarithmic quantizers are more robust than pdf optimized quantizers against a mismatch between signal pdf and quantizer pdf. Since non-uniform quantization is equivalent to uniform quantization of a non-linearly compressed signal, we can specify a non-uniform quantizer by an appropriate companding (compression and expanding) law. The amplitude compression characteristics used in log quantizers follow either the so-called μ-law or the A-law [Gersho, 1978]. The A-law characteristic is used in European PCM telephone systems, while the PCM systems in the United States, Canada, and Japan use μ-law compression. Even though the maximum signal to noise ratio (SNR) of a logarithmic quantizer is less than the maximum SNR of a uniform quantizer, the former is more robust against varying input signal levels. For a wide range of power levels, a high SNR of the logarithmic quantizer is maintained while the SNR of the uniform quantizer drops rapidly with diminishing power levels. To achieve the same quality over a significant dynamic range a 12 bit uniform quantizer or an 8 bit logarithmic quantizer can be used. Thus a saving of 4 bits per sample is achieved by using non-uniform quantization.

1.5.1.2 *Adaptive Quantization* Adaptive quantization utilizes a quantizer (uniform or non-uniform) whose step size is adapted to the short-term amplitude of the signal. In an Adaptive Pulse Code Modulation (APCM) system, the quantizer step size is adjusted at a rate that approximates the syllabic rate (10-30 ms) of running speech. Fig. 1.5 shows a schematic diagram of an APCM system.

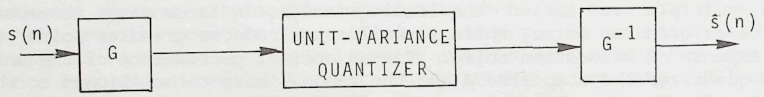


Figure 1.5. Adaptive PCM (APCM) system.

The quantizer is assumed to be fixed and the quantizer scale factor G is varied to be inversely proportional to the short term root-mean square (rms) value of the signal. Two general methods for sending G to the receiver are forward estimation and backward estimation [Noll, 1975]. In forward estimation G is computed block wise (10-30 ms) and transmitted to the receiver for every block as side information. In backward estimation G is updated for every sample and computed from previously quantized samples and need not be transmitted. In general, forward adaptation is more optimal and more robust than backward adaptation. At a given bit rate, the performance (in terms of SNR) of adaptive quantizers is better than that of logarithmic quantizers [Noll, 1974].

1.5.2 *Predictive Coding*

Thus far, we have only taken into account the amplitude characteristics of the speech signal. A more efficient coder could be designed if we are also to take into account the correlations among the speech samples. Two types of correlations can be recognized: the correlation between successive speech samples (causing a non-flat spectral envelope), and the correlation between adjacent pitch periods (causing the spectral fine structure). If a dependence between successive samples exists, we could assume that a speech sample s(n) could be approximately predicted as a linear combination of a number of immediately preceding samples [Atal and Hanauer, 1971] [Makhoul, 1975] [Markel and Gray, 1976]

$$\hat{s}(n) = -\sum_{k=1}^{p} a_k s(n-k) \qquad (1.1)$$

where a_k, $1 \le k \le p$, are called predictor coefficients and p is the order of the predictor. The error r(n) between the actual value and the predicted value is called the residual and is given by

$$r(n) = s(n) - \hat{s}(n) = s(n) + \sum_{k=1}^{p} a_k s(n-k) \qquad (1.2)$$

The problem is to find the predictor coefficients $\{a_k\}$, that minimize

the error r(n) in an appropriate sense. Using the mean-squared error criterion, and leaving the range of summation currently undefined, we have to minimize

$$\sum_n r^2(n) = \sum_n [s(n) + \sum_{k=1}^{p} a_k s(n-k)]^2 \qquad (1.3)$$

By setting the partial derivatives with respect to each a_k to zero, we obtain the normal equations

$$\sum_{k=1}^{p} a_k \sum_n s(n-k)s(n-i) = -\sum_n s(n)s(n-i) \ , \quad 1 \le i \le p \qquad (1.4)$$

The choice of the summation range will affect the method of solution and the properties of the resulting optimum predictor. Popular methods are the auto-correlation method [Markel and Gray, 1973], and the covariance method [Atal and Hanauer, 1971]. In the auto-correlation method the signal is appropriately windowed, such that it is identically zero outside the interval $1 \le n \le L$. As a result the normal equations will become a Toeplitz set and can be solved very efficiently. In the covariance method we use the data in the interval $1 \le n \le L$, thereby minimizing the squared error over the interval $p+1 \le n \le L$.

By denoting the z-transforms of signals by the corresponding capital letters, we obtain for the predictor residual

$$R(z) = A(z)S(z) \qquad (1.5)$$

where

$$A(z) = 1 + \sum_{k=1}^{p} a_k z^{-k} \qquad (1.6)$$

From Eq. (1.5) we see that R(z) can be obtained by filtering S(z) through an all-zero filter with transfer function A(z). We also see from Eq. (1.5) that if R(z) has a flat spectrum, then the signal spectrum can be modeled by the spectrum of the all-pole filter 1/A(z). The filter A(z) is also known as the "inverse filter" since it is the inverse of the all-pole model of the signal spectrum.

1.5.2.1 *Adaptive Predictive Coding* To match the time-varying spectral characteristics of the input signal, the predictor coefficients can be adapted periodically (typically every 10-30 ms). The concept of an Adaptive Predictive Coder (APC) is shown in Fig. 1.6.

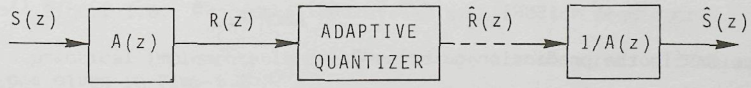


Figure 1.6. Conceptual Adaptive Predictive Coding (APC) system.

In this figure the adaptive quantizer represents the APCM system of Fig. 1.5. The filters $A(z)$ and $1/A(z)$ are sometimes called the analysis and the synthesis filter, respectively. $R(z)$ is the residual which is quantized as $\hat{R}(z)$ and transmitted together with the side information (i.e. filter coefficients and quantizer scale factor). For an infinite level quantizer, $\hat{r}(n)$ is equal to $r(n)$ and the synthetic signal $\hat{s}(n)$ will be equal to the original $s(n)$. If we use a quantizer with a finite number of levels the $A(z)$ filter is operating on the non-degraded input signal, while the $1/A(z)$ filter uses a quantized version of the input. Due to the feedback in the synthesis filter, any quantization noise present can be emphasized. To prevent this situation, the quantizer is placed in a feedback loop, as shown in Fig. 1.7.

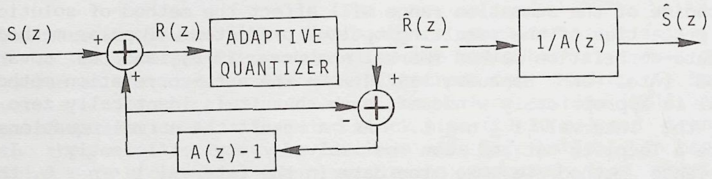


Figure 1.7. Practical Adaptive Predictive Coding (APC) system.

From Fig. 1.7 we can write

$$R(z) = A(z)S(z) + [A(z)-1]\,Q(z) \tag{1.7}$$

where

$$Q(z) = \hat{R}(z) - R(z) \tag{1.8}$$

represents the quantization noise. The synthetic signal $\hat{S}(z)$ is given by

$$\hat{S}(z) = S(z) + Q(z) \tag{1.9}$$

Therefore, the error $S(z)-\hat{S}(z)$ in quantizing the signal $S(z)$ is equal to the residual quantization error $Q(z)$, which depends on the variance of the residual signal. The ratio between the variance of the speech signal and that of of the residual signal, describes the effect of the predictor and is called prediction gain. The actual value of the prediction gain depends on the shape of the input signal spectrum and will be high for voiced segments, and low for unvoiced segments. In [Makhoul and Berouti, 1979] it was shown that

$$(SNR)_{APC} = S/R + (SNR)_{APCM} \quad \text{dB} \tag{1.10}$$

where S/R is the prediction gain in dB.

1.5.2.2 *Differential Coding* If we choose for $A(z)$ a filter with fixed coefficients, we call the predictive system a Differential PCM (DPCM) system. The coefficients are computed by taking the long-term average

of the signal spectrum and computing the optimal predictor coefficients. Note that the effect of increasing the order p tends to saturate, in general, around a predictor order p of about 2 or 3 due to the changing spectral characteristics of the input signal. If we adapt the predictor to the input signal we get an Adaptive Differential PCM (ADPCM) system [Jayant, 1974].

1.5.2.3 *Delta Modulation* Delta Modulation (DM) is basically a DPCM system, which uses a 1-bit (2-level) quantizer and a sampling rate much larger than the Nyquist rate. The over-sampling of the waveform increases the adjacent sampling correlation, which justifies the use of a 2-level quantizer. However, *rapidly varying signals cause slope overload* distortion and steady state signals introduce granular noise. The use of adaptive quantizers reduces these effects. Again, different strategies are possible. Adaptive Delta Modulation (ADM) uses a variable step size, based on previous quantizer outputs. When the step size is adapted very smoothly in time, we obtain a continuously variable slope delta modulation coder (CVSDM) [Jayant, 1974]. Another adaptation strategy is Digitally Controlled Delta Modulation (DCDM) [Greefkens et al., 1970].

1.5.2.4 *Pitch Prediction* The adaptive predictor tends to remove the correlation between adjacent samples. Such prediction, however, ignores the correlation between samples spaced a pitch period apart, due to the quasi-periodic nature of voiced speech. One method that exploits this correlation is the usage of an inverse pitch filter $P(z)$ [Atal and Schroeder, 1970], defined by the general form

$$P(z) = 1 + \beta_1 z^{-(M-1)} + \beta_2 z^{-M} + \beta_3 z^{-(M+1)} \tag{1.11}$$

where M is the pitch period in samples, and β_1, β_2 ,and β_3 are the three pitch predictor coefficients. The delay M is chosen so that the correlation between samples delayed M samples apart is the highest. The β-parameters are then chosen to minimize the error signal. If β_1 and β_3 of the three-tap filter are set to zero, we obtain a one-tap pitch predictor which also yields good results but is less complicated than the three-tap predictor. The concept of an APC coder with pitch prediction is drawn in Fig. 1.8.

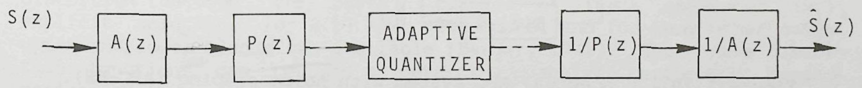


Figure 1.8. Conceptual APC system with a pitch predictor.

For a practical implementation we have to use a configuration similar to the one given in Fig. 1.7.

1.5.2.5 *Noise Spectral Shaping* The quantization noise in APC generally has a flat spectrum. To ensure that the distortion in the speech signal is perceptually small, it is necessary to consider the spectrum of the quantization noise and its relation to the speech spectrum. An example of the noise spectrum in APC is shown in Fig. 1.9. If the short-term spectrum has a large dynamic range, as in voiced speech (dashed line), it is possible for the noise level to be above the signal level in some frequency ranges (solid line). In noise spectral shaping, one shapes the quantization noise spectrum such that it remains below the signal spectrum as much as possible (dash-dotted line).

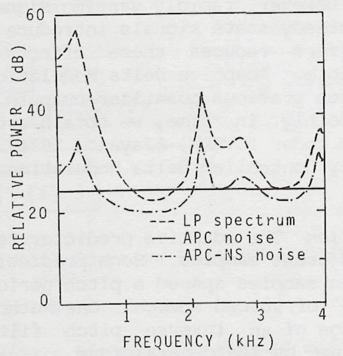


Figure 1.9. Comparison between the noise spectrum in APC and in APC-NS.

Fig. 1.10 shows a conceptual APC system with noise shaping (APC-NS) [Makhoul and Berouti, 1979] [Atal and Schroeder, 1979].

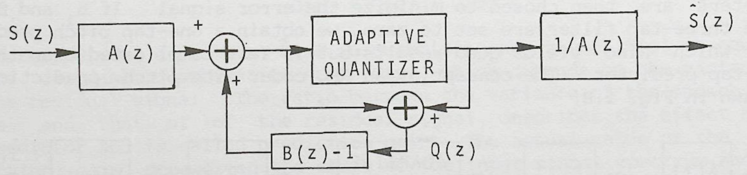


Figure 1.10. Conceptual APC system with noise shaping (APC-NS).

The output signal at the receiver is given by

$$S(z) = S(z) + B(z)/A(z)\ Q(z) \tag{1.12}$$

By an appropriate choice of the filter B(z), the noise spectrum can be given any shape. A filter that has proven to be a suitable choice [Makhoul and Berouti, 1979] [Atal and Schroeder, 1979] is an all-zero filter of the form

$$B(z) = A(z/\gamma) = 1 + \sum_{k=1}^{p} a_k \gamma^k z^{-k}, \quad 0 \le \gamma \le 1 \tag{1.13}$$

where γ is a constant (typically γ=0.85). Note that for a value of γ equal to one, the APC-NS coder is equivalent to the APC coder without noise shaping. The SNR for the APC-NS system is always less than the SNR of an APC coder without noise shaping [Makhoul and Berouti, 1979]. However, even if the SNR drops, the perceptual effect can be a reduced noise level and higher speech quality. An adaptive predictive coding system exploiting pitch prediction and noise shaping techniques can produce toll-quality speech at or around 16 kb/s.

1.5.3 *Vector Quantization*

A generalization of PCM is vector quantization (VQ) [Buzo et al., 1980] [Abut et al., 1982] [Gray, 1984]. In VQ a block of L consecutive samples of the input waveform (or an appropriate transform thereof), forms an L-dimensional vector x that is treated as one entity. In the encoding process, each input vector x is assigned to the nearest representative vector y_i in a set of N stored reference vectors known as the "code book". The best matching pattern in the code book is selected by the encoding process according to a suitable fidelity measure (e.g. mean squared error) and a binary word is used to identify this pattern in the code book. Even if the input samples are statistically independent, an advantage can be gained by quantizing a block at a time rather than one sample at a time. This is a fundamental result of Shannon's rate distortion theory [Shannon, 1948] [Shannon, 1959]. The distortion of the block quantizers is defined as

$$D = \frac{1}{L} \sum_{i=1}^{L} \bar{e}_i^2 \tag{1.14}$$

where $\bar{e}_i^2$ is the mean-square error of the i-th sample. As the block length L increases, the minimum bit rate needed for a given distortion will decrease. If L goes to infinity, the minimum bit rate approaches a limiting value R depending on D. The function R(D) is called the rate-distortion function. For certain classes of input sequences $\{x_i\}$, explicit solutions for R(D) have been found, and for many other cases, upper and lower bounds are available [Berger, 1971]. These rate distortion functions can be used to determine quantizer performance by comparison of the quantizer distortion and the rate-distortion limits. Operation of the decoder is simple: it stores the code book and looks up the reproduction vector indexed by the encoder. The number of bits/sample is given by

$$R = \frac{1}{L} \log_2 N \tag{1.15}$$

where L is the dimension of the vector and N the number of patterns in the code book. To find appropriate code books, a generalization of the

Lloyd-Max design procedure can be used [Gray, 1984]. Disadvantages of the full-search VQ are the computational and storage requirements. The number of computations per sample is of the order of $N=2^{LR}$ and a memory of LN words is required. For these reasons, the direct use of VQ on speech waveforms with optimal code books has been limited to dimension-rate products (LR) of 8 or less [Abut et al., 1982]. With this limitation, the resulting quantizers compare favorably with scalar quantization (i.e. PCM), but the overall performance achieved is not adequate for practical systems. An important reduction in computational complexity may be reached by using tree-structured code books [Gray and Abut, 1982] to allow a rapid search. In tree-structured VQ, a large part of the set of stored patterns in the code book is used only to guide the search. For example, in a binary tree code book of size 2^{LR}, the search consists of LR binary decisions; only at the last decision stage is the input vector compared with two patterns which are actual code vector candidates for representing the input vector. Two major problems are now that the resulting coding algorithm is sub-optimal and the required memory is increased over the full-search case. A reduction in both coding complexity and memory may be achieved by using a multi-stage approach [Juang, 1982]. Here, the encoding error of a vector quantizer is formed by taking the difference between the original and quantized vectors and then feeding it into a second vector quantizer. The final index to be transmitted is formed by the concatenation of indices from each code book. Another useful structure for a VQ is a product code. A typical example of a product VQ is a gain/shape VQ [Buzo et al., 1980] [Sabin and Gray, 1984], where separate codes are used to code the normalized input vector (shape) and the normalization term (gain). Finally, we mention the use of lattices to design a so-called lattice quantizer [Gersho, 1979], which enables efficient searches and memory usage.

In addition to the memoryless vector quantizers described, structures of VQ with memory have been proposed. Examples are the vector predictive quantizer [Cuperman and Gersho, 1982], which is a generalized DPCM coder, and the finite-state VQ [Foster and Gray, 1982]. Further, we want to mention the possibility of using VQ with other waveform coding techniques such as Sub-Band Coding [Gersho et al., 1984] and Residual Excited Linear Predictive (RELP) coding [Rebolledo et al., 1982].

1.6 Waveform Coding (Frequency Domain Methods)

The waveform coding methods described in the previous sections all treat speech as a single full-band signal. The main differences in the various methods are determined by the degree of prediction that is exploited, and by whether or not schemes are adaptive. This whole class of coding methods is commonly referred to as time domain wave form coders. Another class of coding methods is referred to as frequency domain waveform coders. These coding techniques divide the speech signal into a number of separate frequency components and encode each of these components separately.

1.6.1 *Sub-Band Coding*

In the Sub-Band Coder (SBC) [Crochiere et al., 1976] [Tribolet and Crochiere, 1979] the speech signal is partitioned into a number (typically 4 or 8) of frequency bands (sub-bands). Each sub-band is low-pass transposed and sampled at its Nyquist rate. The samples are encoded by using APCM techniques and the number of bits per sample in each band is chosen according to perceptual criteria for that band. At the reconstruction side, the different bands are decoded and band-pass transposed back, and added to produce the reconstructed speech signal. If the number of bits for different bands is changed dynamically, the noise spectrum can be shaped to reduce the subjectively perceived noise in the reconstructed signal. This approach can produce toll-quality speech at or above 24 kb/s.

1.6.2 *Adaptive Transform Coding*

The Adaptive Transform Coder (ATC) [Zelinski and Noll, 1977] [Tribolet and Crochiere, 1979] divides, analogously to the sub-band coder, the speech band into a number of frequency bands. However, many more bands are used (typically 128) and the translation to the frequency domain is achieved by means of a fast transform algorithm, such as the discrete cosine transform (DCT) [Makhoul, 1980]. The transform coefficients are APCM quantized and transmitted. An inverse transformation gives the reconstructed signal at the receiver. The number of bits assigned to each band is changed adaptively, resulting in an algorithm capable of fully adapting, dynamically, to the spectral properties of speech. An Adaptive Transform Coder exploiting noise shaping techniques is capable of producing toll-quality speech at or above 16 kb/s.

1.6.3 *Harmonic Coding*

Harmonic Coding (HC) [Almeida and Tribolet, 1982] [Almeida and Tribolet, 1984] [Almeida,] is a hybrid coding technique based on combining ATC with a model of voiced speech. The main issue in a HC scheme is that a good reproduction of voiced speech segments is essential for achieving a high synthetic speech quality. This goal is achieved by coding the amplitudes and phases of the spectral lines of voiced speech. The parameters obtained are quantized and coded, and then used to synthesize a spectrum, which is subtracted from that of the input speech, yielding the modeling residual. This residual is also quantized, coded and transmitted together with the model parameters. At the receiver the residual spectrum and the model spectrum are added, and an inverse transformation is applied to obtain the time signal. The available number of bits is dynamically allocated between the model parameters and the modeling residual. In segments that can be accurately modeled (e.g. voiced segments), most or all bits are allocated to the model parameters, and very few or none to the modeling residual. Conversely, in segments where the modeling accuracy is very low (e.g. unvoiced segments) all bits are allocated to the modeling residual. In the latter case the coder behaves like an ATC coder. A Harmonic Coder can produce toll-quality speech at or above 10 kb/s.

1.7 Hybrid Coders Using Adaptive Prediction Techniques

Hybrid coding methods combine features of both waveform and source coders in an attempt to fill the performance gap between these two methods. Although many other hybrid approaches exist, we limit ourselves to time-domain approaches that use Adaptive Prediction techniques. These coders can be considered as residual coders, similar to the APC coder with forward adaptation. In forward adaptation the various parameters are computed from a block of the speech signal and transmitted as side information in addition to the residual. In the APC system described the quantizer is included in the predictor loop and every sample is quantized. The coded prediction residual usually requires a significantly larger number of bits per second in comparison to the side information. To enable further reduction in bit rate, we have to use a coarser quantization of the residual. From this point of view, source coding - where the residual is replaced by either noise or pulse excitation - is the ultimate in residual quantization. Due to the low speech quality of vocoders, we have to search for better solutions that produce (near) toll-quality speech at bit rates below 16 kb/s.

To reduce the bit rate required for coding the residual signal, we cannot simply decrease the number of quantization levels. Such an operation results in either peak clipping or a significant amount of granular noise. Moreover, the quantization noise cannot be assumed to be white due to its high correlation with the signal. The effects mentioned are undesirable. The APC coder with a center-clipper [Atal, 1982] is based on the observation that accurate quantization of high amplitude portions of the prediction residual is necessary for achieving low perceptual distortion in the decoded speech. This means that most of the available bits are used for encoding the high-amplitude portions of the residual signal. To keep the bit rate within a specified value, the prediction residual can be center-clipped prior to quantization by a multi-level quantizer.

1.7.1 *Base-Band Coding*

Another approach to the quantization of the residual signal is the Base-Band Coder (BBC) [Viswanathan et al., 1982] (Fig. 1.11). Instead of transmitting the full-bandwidth (W Hz) residual (at 2W rate) as is done in APC, we only transmit a low frequency portion (base-band) of it, B Hz wide at 2B rate. Typically a ratio of W/B = 3 or 4 is used. At the receiver, we have to regenerate the high-frequency components of the residual from the base-band. A widely used method of High Frequency Regeneration (HFR) is that of spectral folding [Makhoul and Berouti, 1979]. A disadvantage of the base-band coder with HFR is the effect of "tonal noises" in the output speech due to a mismatch between down-sampling frequency and pitch frequency of the speech signal.

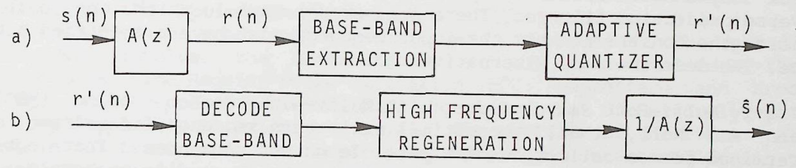


Figure 1.11. Base-Band Coding system (BBC), (a) encoder, (b) decoder.

Many of the proposed solutions for this problem [Viswanathan et al., 1982] [Hedelin, 1983] introduce some other forms of distortion (such as hoarseness). Another solution using a pitch predictor for removing periodicity prior to down sampling has been reported to give good results without the introduction of other forms of distortion [Sluyter et al., 1984].

1.7.2 *APC With Delayed Decision Coding*

Thus far the quantization of the residual has been achieved on an instantaneous basis, in contrast to the forward prediction which is done on a non-instantaneous basis. In computing the predictor parameters we attempt to minimize some global error criterion over the frame while not doing so for the residual. If we perform the quantization of the residual on a frame basis, we can achieve a more optimal performance. Such an approach is referred to as Delayed Decision Coding [Jayant and Noll, 1984, Chapter 9] The problem could be formulated as follows: given a desired bit rate and the side information (spectral and pitch information in APC), find the input sequence that minimizes a given error criterion over a frame of speech. A general block diagram is shown in Fig. 1.12, where the

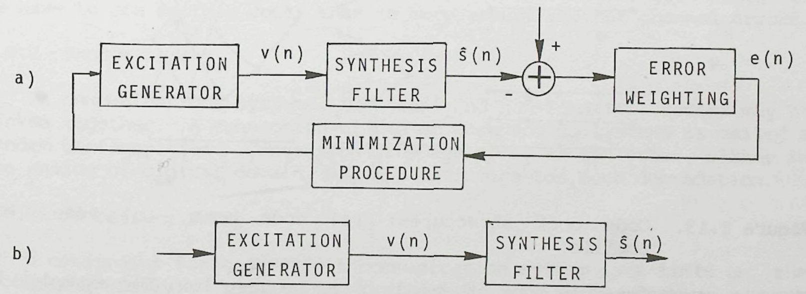


Figure 1.12. Adaptive Predictive Coding with Delayed Decision Coding. (a) encoder, (b) decoder.

error criterion is incorporated in the block "error weighting", and the block "synthesis filter" represents both the short- and long-term inverse prediction filters. There are several ways in which one could impose the constraint that the input sequence must have a specified bit rate. We describe some alternatives.

1.7.2.1 *Multi-Path Search Coder* With Multi-path Search Coding (MSC) [Fehn and Noll, 1982] the optimal excitation sequence for a frame is determined from a collection of N possible output sequences. That output sequence which achieves a minimum error (according to some criterion) between the input and output signal will be transmitted. The collection of output sequences that is either stored or deterministically generated must be available at both transmitter and receiver. If we use a stored collection of output sequences, we only have to transmit the index of the sequence that produces a minimum error (code book coding). The code book can be generated with representative samples of the underlying signal, or signals that bear a close resemblance to the signal to be coded, such as Gaussian noise for residual signals [Atal, 1982] [Atal and Schroeder, 1984]. For the latter case, the code can be generated on-line and excessive storage space is avoided. When the code book contains many sequences and when we want to find the optimal sequence for large frames (> 20 samples), the search procedure becomes impractical very quickly. To reduce search times, sub-optimal methods such as tree and trellis coding [Fehn and Noll, 1982] [Stewart et al., 1982] can be used. In these methods different sequences have a number of common elements. Each sequence forms a path through the tree or trellis. The transmitted code will provide information about how to trace through the tree or trellis. Fig. 1.13 shows the different structures of the coding books.

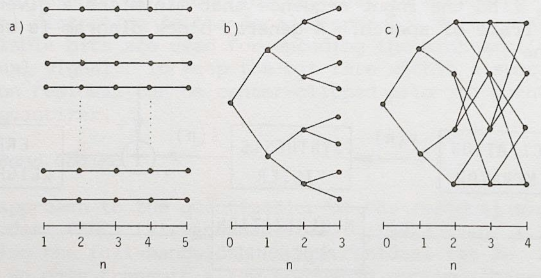


Figure 1.13. Code book structures: (a) code book, (b) tree, (c) trellis.

The bit rates for this type of encoding can be very low. For example in [Atal and Schroeder, 1984] it was reported that for achieving (near) toll quality, only 2 kb/s are required for encoding the excitation sequence.

1.7.2.2 *Regular-Pulse and Multi-Pulse Excitation Coder* Rather than selecting the optimum sequence from a collection of possibilities, the excitation can be specified by a small set of parameters. The encoding procedure involves the determination of the optimum values of these parameters. The Regular-Pulse Excitation (RPE) [Deprettere and Kroon, 1985] coder represents the residual signal as a set of uniformly spaced pulses (typically 10 pulses per 5 ms). The position of the first pulse within a frame and the amplitudes have to be determined during the encoding procedure. The Multi-Pulse Excitation (MPE) coder [Atal and Remde, 1982] represents the residual as a sequence of pulses located at non-uniformly spaced intervals. The coder searches these pulses (typically 8 pulses per 10 ms) such that the generated output signal is as close to the original as possible, for the given fidelity criterion. Finding the pulse positions and their corresponding amplitudes is a highly non-linear problem and sub-optimal strategies have to be used. As in the case of the Multi-path Search Coders, pitch prediction and noise shaping increase the perceived speech quality for the RPE and MPE coders. Both coders can produce (near) toll quality speech at 10 kb/s.

1.8 Additional Requirements For Speech Coding Systems

The primary goal of speech coding is to achieve the best quality at the lowest bit rate. But depending on the application, several other factors could affect the coding strategy. Some of these factors are described in the remainder of this section.

1.8.1 *Channel Errors*

If the coded signals are transmitted over noisy channels, they are subjected to transmission errors. Depending on the parameters and/or coding method involved, these errors result in unwanted perceivable effects. To prevent such undesired effects, we have to add error protection bits to the coded data at the expense of a higher bit rate, or we have to use another coder that is more robust against channel errors.

1.8.2 *Tandem Coding*

In communications systems, identical and non-identical coders may be linked together. A connection of two or more coding systems is called a tandem configuration. Conversion from one coder to the other, either in the analog or digital domain, may not introduce too much degradation.

1.8.3 *Processing Delay*

A constraint for coders in a communication system is a limit on the processing delay. Increasing the delay in telecommunication systems makes communications more sensitive to echos. Forward prediction systems such as the APC coder and MSC and MPE coders have a processing delay equal to or greater than the duration of one analysis frame.

1.8.4 *Transmission Of Non-Speech Signals*

In certain applications (telephony systems) not only speech has to be transmitted but also non-speech signals such as in-band signals and data signals from modems. The problem is that the spectral or temporal characteristics of non-speech signals are different from those of speech. This leads to a sub-optimal performance for data signals of coders optimized for speech signals.

1.8.5 *Speech Signal Degradation*

In practice, the quality of the input speech can be degraded. For example, the speech may be distorted by the use of non-linear (e.g. carbon) microphones. Another effect is the band limitation introduced by a practical telephone system. This can lead to a loss of the lower band of the speech signal (up to 300 Hz). During coder design the effects of these degradations have to be considered.

1.8.6 *Variable Rate Coding*

For speech coding applications in digital networks such as packet transmission systems [Gold, 1977] and Integrated Services Digital Networks (ISDN) [Decina, 1982], it is desirable to use coding techniques that can operate at variable rates. Rather than dropping parts of active speech in the case of channel overload, the coders are temporally switched to a lower rate. Obviously such a variable rate coding also implies a variable quality coding, but the network performance will be better than when fixed rate coding is used. Coding techniques which show some hierarchy in the encoded data are very suitable for variable bit rate systems. Due to the hierarchical structure, the less important parameters can be dropped to decrease the bit rate. This type of coder is called an embedded coding system (e.g. PCM). Some coding techniques (e.q. DPCM) which lack this hierarchy can be realized as an embedded system by modifying the coder structure [Jayant, 1983].

1.9 **Evaluation Of Speech Coders**

The evaluation of speech coders has many aspects. The most important one is the analysis of the speech quality and of conditions (such as channel errors and acoustical background noise) that could deteriorate the coded speech quality. Besides the quality aspects, we also consider the algorithm complexity and its storage requirements. These factors could be of crucial importance for real-time implementations. Speech quality is a hard thing to measure due to its close connection to human perception. Two methods for quality assessment of coder performance could be distinguished: objective tests and subjective tests. Objective tests use some fidelity measure that reflects as much as possible the quality of the subjectively perceived speech. Subjective tests depend heavily on tests conducted with human listeners.

1.9.1 *Objective Testing Procedures*

Waveform coders and hybrid coders, which are close to the former, try to reconstruct the original signal as closely as possible. This means that we could use the difference between the original signal s(n) and the reconstructed signal ŝ(n) as a measure of quality. A simple and widely used measure is the long-time-averaged signal-to-noise ratio (SNR).

$$\mathrm{SNR} = 10 \log_{10} \frac{\sum_n s^2(n)}{\sum_n [\hat{s}(n)-s(n)]^2} \quad \mathrm{dB} \qquad (1.16)$$

where summations are typically done over several seconds of speech. A disadvantage of this measure is its susceptibility to high energy components, which leads to an unfair treatment of low energy segments (such as fricatives) whose preservation is perceptually important.

A better measure is the segmental SNR (SNRSEG) [Noll, 1974], which is the average of SNR values, computed over short-time (typically 10-20 ms) segments.

$$\mathrm{SNRSEG} = \frac{1}{N} \sum_{i=1}^{N} (\mathrm{SNR})_i \quad \mathrm{dB} \qquad (1.17)$$

where $(\mathrm{SNR})_i$ corresponds to the SNR in dB for segment i computed in the same manner as in Eq. (1.16) and N corresponds to the number of segments i in the speech utterance.

The SNRSEG tends to assign more equitable weightings to loud and soft parts of a sentence, thereby reflecting better the quality of the perceived signal. Although SNR and segmental SNR measures correspond adequately well with the subjectively perceived quality of waveform coders, these measures tend to fail if used for hybrid coders employing spectral noise shaping. This is mainly due to the fact that spectral shaping reduces the SNR but enhances the perceived speech quality. Despite this restriction, an objective measure such as the SNRSEG serves well as a guideline for optimizing coder parameters.

1.9.2 *Subjective Testing Procedures*

Speech quality is traditionally assessed by the criterion that a listener understands what is being said (intelligibility), and who said it (speaker recognizability). Speech intelligibility tests are sometimes used to assess the quality of synthetic speech. All of these tests depend upon a human listener indicating his understanding of what was said (syllables, words or sentences). A widely used intelligibility test is the Diagnostic Rhyme test [Voiers, 1977]. A disadvantage of using intelligibility as a measure for quality is that speech intelligibility is usually not a major problem in waveform and hybrid coded speech unless the bit rate is very low. Furthermore, speaker recognizability is very easy for waveform and hybrid coders. Many vocoders, on the other hand, have a tendency to make everybody sound alike and thus have poor speaker recognizability. Some characteristics of a good procedure for determining speech quality are convenience (so that testers will use it), validity (so that coders that provide good quality speech will obtain high scores), and reliability (so that a given condition will obtain the same score in different tests). A quality test that is gaining more and more acceptance is the Mean Opinion Score (MOS) test [Daumer, 1982] [Goodman and Nash, 1982].

1.10 Summary

During the presentation of many existing coding concepts, the reader may have lost his or her way in the maze of the most important developments. In this section we try to summarize the most important concepts in relation to associated applications. Vocoders are not very suitable for commercial telephony systems due to their synthetic quality. However, for applications where low bit rates are required and low speech quality is accepted (military systems), they can be useful. They have been very suitable as well for applications where manipulation with speech material is required such as in text-to-speech converters and source coding methods. On the other hand, there are simple waveform coding systems such as log-PCM and APCM which are capable of producing toll quality speech at 64 kb/s. Currently μ-law and A-law log PCM are used as an international standard in commercial telephony systems. ADPCM is being considered as a new international standard for 32 kb/s channels in commercial telephony systems. There is a growing interest in hybrid coding techniques such as Adaptive Predictive Coding in combination with Delayed Decision Coding, which are capable of producing (near) toll quality speech at transmission rates around 10 kb/s. The availability of 9600 b/s modems which operate reliably over conventional dial-up lines and the growth of mobile telephony systems offers many applications for these hybrid coders. The high complexity of the hybrid coders described is worthy of note. Even with modern signal processing chips these algorithms are difficult to implement in real-time systems.

Further Reading

This chapter served as a general introduction to the subject of speech coding. For a reader who is interested in a more thorough treatment of the topics discussed, we will describe some of the books that we consider as good reference works. The speech production process and human speech perception is treated in [Flanagan, 1972]. Although not really up to date, a good reference for linear prediction coding of speech is [Markel and Gray, 1976]. An overview of digital signal processing techniques for speech coding, and a description of waveform coders and vocoders can be found in [Rabiner and Schafer, 1978]. The most general overview of waveform coders and hybrid coders is given in [Jayant and Noll, 1984]. This book deals with applications of the techniques described to the encoding of both speech and video signals.

References

Abut, H., R.M. Gray, and G. Rebolledo, "Vector quantization of speech and speech like waveforms," *IEEE Trans. Acoust., Speech, Signal Processing* Vol. **ASSP-30** pp. 423-435 (June 1982).

Adams, W.C. and C.E. Giesler, "Quantizing characteristics for signals having laplacian amplitude probability density functions," *IEEE Trans. Commun.* Vol. **COM-26** pp. 1295-1297 (August 1978).

Almeida, L.B. and J.M. Tribolet, "Nonstationary spectral modeling of voiced speech," *IEEE Trans. Acoust., Speech and Signal Processing* Vol. **ASSP-31**(3) pp. 644-678 (June 1983).

Almeida, L.B. and J.M. Tribolet, "Harmonic coding: a low bit-rate , good quality speech coding technique," *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing*, pp. 1664-1667 (March 1982).

Almeida, L.B. and J.M. Tribolet, "Harmonic Coding: an introduction," *Proc. IEEE Int. Conf. Communications*, p. 38.4 (May 1984).

Atal, B.S. and M.R. Schroeder, "Adaptive Predictive Coding of speech signals," *Bell Sys. Tech. J.*, pp. 1973-1986 (Oct. 1970).

Atal, B.S. and S.L. Hanauer, "Speech analysis and synthesis by linear prediction of the speech wave," *J. Acoust. Soc. Amer.* Vol. **50** pp. 637-655 (1971).

Atal, B.S. and L.R. Rabiner, "A pattern recognition approach to voiced-unvoiced-silence classification with applications to speech recognition," *IEEE Trans. Acoust., Speech, Signal Processing* Vol. **ASSP-24** pp. 201-212 (June 1976).

Atal, B.S. and M.R. Schroeder, "Predictive coding of speech and subjective error criteria," *IEEE Trans. Acoust., Speech, Signal Processing* Vol. **ASSP-27** pp. 247-254 (June 1979).

Atal, B.S., "Predictive coding of speech at low bit rates," *IEEE Trans. Communications* Vol. **COM-30**(4) pp. 600-614 (April 1982).

Atal, B.S. and J.R. Remde, "A new model of LPC excitation for producing natural-sounding speech at low bit rates," *Proc. IEEE Int. Conf. Acoust., Speech and Signal Processing*, pp. 614-617 (April 1982).

Atal, B.S. and M.R. Schroeder, "Stochastic coding of speech signals at very low bit rates," *Proc. IEEE Int. Conf. Communications*, p. 48.1 (May 1984).

Berger, T., *Rate distortion theory*, Prentice Hall, Englewood Cliffs NJ (1971).

Bosscha, G.J. and R.J. Sluyter, "DFT-Vocoder using harmonic-sieve pitch extraction," *Proc. IEEE Int. Conf. Acoust., Speech and Signal Processing*, pp. 1952-1955 (April 1982).

Buzo, A., A.H. Gray, R.M. Gray, and J.D. Markel, "Speech coding based upon vector quantization," *IEEE Trans. Acoust., Speech, Signal Processing* Vol. **ASSP-28** pp. 562-574 (Oct. 1980).

Crochiere, R.E., S.A. Webber, and J.L. Flanagan, "Digital Coding of speech in subbands," *Bell Syst. Tech. J.* Vol. **55**(10) pp. 1069-1085 (Oct. 1976).

Cuperman, V. and A. Gersho, "Adaptive differential vector coding of speech," *Conf. Rec. IEEE GLOBCOM*, pp. E6.6.1-E6.6.5 (Dec. 1982).

Daumer, W.R., "Subjective evaluation of several efficient speech coders," *IEEE Trans. Communications* Vol. **COM-30** pp. 655-662 (April 1982).

Decina, M., "Managing ISDN through international standard activities," *IEEE Communications Magazine*, pp. 19-25 (Sep. 1982).

Deprettere, Ed. F. and P. Kroon, "Regular excitation reduction for effective and efficient LP-coding of speech," *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing*, pp. 25.8.1-25.8.4 (March 1985).

Fehn, H.G. and P. Noll, "Multipath search coding of stationary signals with applications to speech," *IEEE Trans. Communications* Vol. **COM-30**(4) pp. 687-701 (April 1982).

Flanagan, J.L., *Speech analysis, synthesis, and perception*, Springer Verlag, Berlin (1972). (2nd edition)

Foster, J. and R.M. Gray, "Finite-state vector quantization," *Abstracts Int. Symp. on IT*, (June 1982).

Gersho, A., "Principles of quantization," *IEEE Trans. Circuits and Systems* Vol. **CAS-25** pp. 427-436 (July 1978).

Gersho, A., "Asymptotically optimal block quantization," *IEEE Trans. Inform. Theory* Vol. **IT-25**(4) pp. 373-380 (July 1979).

Gersho, A., I. Versvik, and T. Ramstad, "Fully vector-quantized sub-band coding with adaptive code book allocation," *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing*, pp. 10.7.1-10.7.4 (March 1984).

Gold, B. and C.M. Rader, "The channel vocoder," *IEEE Trans. Audio and Electroacoust.* Vol. **AU-15** pp. 148-161 (Dec. 1967).

Gold, B. and L.R. Rabiner, "Parallel Processing techniques for estimation pitch periods of speech in the time domain," *J. Acoust. Soc. Amer.* Vol. **46**(2) pp. 442-448 (August 1969).

Gold, B., "Digital Speech networks," *Proc. IEEE*, pp. 1636-1658 (Dec. 1977).

Goodman, D.J. and R.D. Nash, "Subjective quality of the same speech transmission conditions in seven different countries," *IEEE Trans. Communications* Vol. **COM-30** pp. 642-654 (April 1982).

Gray, R.M. and H. Abut, "Full-search and tree-searched vector quantization of speech waveforms," *Proc. IEEE Int. Conf. Acoust., Speech and Signal Processing*, pp. 593-596 (April 1982).

Gray, R.M., "Vector Quantization," *IEEE ASSP Magazine* Vol. **1**(2) pp. 4-29 (April 1984).

Greefkens, J.A. and K. Riemens, "Code Modulation with digitally controlled companding for speech transmission," *Philips Tech. Rev.* Vol. **31**(11/12) pp. 335-353 (1970).

Hedelin, P., "RELP-vocoding with uniform and non-uniform down-sampling," *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing*, pp. 1320-1323 (March 1983).

Huggins, W.H., "A note on autocorrelation analysis of speech sounds," *J. Acoust. Soc. Amer.* Vol. **26** pp. 790-792 (Sep. 1954).

Jayant, N.S., "Digital Coding of speech waveforms: PCM, DPCM and DM quantizers," *Proc. IEEE* Vol. **62** pp. 611-632 (May 1974).

Jayant, N.S., "Variable rate ADPCM coding of speech based on explicit noise coding," *Bell Syst. Tech. J.*, pp. 707-717 (May-June 1983).

Jayant, N.S. and P. Noll, *Digital coding of waveforms*, Prentice Hall, Englewood Cliffs, New Jersey (1984).

Juang, B.H., "Multiple stage vector quantization for speech coding," *Proc. IEEE Int. Conf. Acoust., Speech and Signal Processing*, pp. 597-600 (April 1982).

Klatt, D.H., "Software for a cascade/parallel formant synthesizer," *J. Acoust. Soc. Amer.* Vol. **67** pp. 971-995 (March 1980).

Lloyd, S.P., "Least squares quantization in PCM," *IEEE Trans. Inform. Theory* Vol. **IT-28** pp. 129-137 (1957). reprint March 1982

Makhoul, J., "Linear Prediction: A tutorial review," *Proc. IEEE* Vol. **63** pp. 561-580 (April 1975).

Makhoul, J. and M. Berouti, "Adaptive noise spectral shaping and entropy coding in predictive coding of speech," *IEEE Trans. Acoust., Speech, Signal Processing* Vol. **ASSP-27** pp. 247-254 (1979).

Makhoul, J. and M. Berouti, "High Frequency regeneration in speech coding systems," *Proc. Int. Conf. Acoust., Speech, Signal Processing*, pp. 428-431 (1979).

Makhoul, J., "A fast cosine transform in one and two dimension," *IEEE Trans. Acoust., Speech, Signal Processing* Vol. **ASSP-28** pp. 27-34 (1980).

Markel, J.D. and A.H. Gray, "On autocorrelation equations as applied to speech analysis," *IEEE Trans. Audio and Electroacoustics* Vol. **AU-21** pp. 69-79 (April 1973).

Markel, J.D. and A.H. Gray, *Linear Prediction of speech*, Springer Verlag, Berlin (1976).

Mauersberger, W., "Experimental results on the performance of mismatched quantizers," *IEEE Trans. Inform. Theory* Vol. **IT-25**(4) pp. 381-386 (July 1979).

Max, J., "Quantizing for minimum distortion," *IRE Trans. Inform. Theory* Vol. **IT-6** pp. 7-12 (March 1960).

Noll, P., "Adaptive quantizing in speech coding systems," *Proc. Int. Zurich Seminar*, pp. B3(1)-B3(6) (March 1974).

Noll, P., "A comparative study of various quantization schemes for speech encoding," *Bell Sys. Tech. J.*, pp. 1597-1614 (Nov. 1975).

Paez, M.D. and T.H. Glisson, "Minimum mean squared error quantization in speech PCM and DPCM systems," *IEEE Trans. Communications* Vol. **COM-20** pp. 225-230 (April 1972).

Rabiner, L.R. and R.W. Schafer, *Digital processing of speech signals*, Prentice Hall, Englewood Cliffs, New Jersey (1978).

Rebolledo, G., R.M. Gray, and J.P. Burg, "A multirate voice digitizer based upon vector quantization," *IEEE Trans. Communications* Vol. **COM-30**(4) pp. 721-727 (April 1982).

Roucos, S., R.M. Schwartz, and J. Makhoul, "Segment quantization for very-low-rate speech coding," *Proc. Int. Conf. Acoust., Speech, Signal Processing*, (April 1982).

Sabin, M.J. and R.M. Gray, "Product code vector quantizers for waveform and voice coding," *IEEE Trans. Acoust., Speech, Signal Processing* Vol. **ASSP-32**(3) pp. 474-488 (June 1984).

Shannon, C.E., "A mathematical theory of communication," *Bell Syst. Tech. J.* Vol. **27** pp. 379-423,623-656 (1948).

Shannon, C.E., "Coding theorems for a discrete source with a fidelity criterion," *IRE Nat. Conv. Rec.*, (4) pp. 142-163 (1959).

Sluyter, R.J., H.J. Kotmans, and A. van Leeuwaarden, "A novel method for pitch extraction from speech and a hardware model applicable to vocoder systems," *Proc. IEEE Int. Conf. Acoust., Speech and Signal Processing*, (April 1980).

Sluyter, R.J., H.J. Kotmans, and T.A.C.M. Claasen, "Improvements of the harmonic-sieve pitch extraction scheme and an appropriate method for voiced-unvoiced detection," *Proc. IEEE Int. Conf. Acoust., Speech and Signal Processing*, pp. 188-191 (April 1982).

Sluyter, R.J., G.J. Bosscha, and H.M.P.T. Schmitz, "A 9.6 kbit/s speech coder for mobile radio applications," *Proc. IEEE Int. Conf. Communications*, (May 1984).

Stewart, L.C., R.M. Gray, and Y. Linde, "The design of trellis waveform coders," *IEEE Trans. Communications* Vol. **COM-30** pp. 702-710 (April 1982).

Tremain, T.E., "The government standard linear predictive coding algorithm:LPC-10," *Speech Technology*, pp. 40-49 (April 1982).

Tribolet, J.M. and R.E. Crochiere, "Frequency domain coding of speech," *IEEE Trans. Acoust., Speech, Signal Processing* Vol. **ASSP-27** pp. 512-530 (1979).

Viswanathan, V.R., J. Makhoul, R.M. Schwartz, and A.W.F. Huggins, "Variable Frame Rate transmission: a review of methodology and application to narrow-band LPC speech coding," *IEEE Trans. Communications* Vol. **COM-30** pp. 674-686 (April 1982).

Viswanathan, V.R., A.L. Higgins, and W.H. Russel, "Design of a robust base-band LPC coder for speech transmission over noise channels," *IEEE Trans. Communications* Vol. **COM-30**(4) pp. 663-673 (April 1982).

Voiers, W.D., "Diagnostic evaluation of speech intelligibility," in *Speech intelligibility and speaker recognition*, ed. M.E. Hawley, Dowden Hutchinson Ross, Stroudsburg, PA (1977).

Wong, D.Y., B.H. Juang, and A.H. Gray, "An 800 bits/s vector quantization LPC vocoder," *IEEE Trans. Acoust., Speech, Signal Processing* Vol. **ASSP-30** pp. 770-780 (Oct. 1982).

Zelinski, R. and P. Noll, "Adaptive transform coding of speech signals," *IEEE Trans. Acoust., Speech, Signal Processing* Vol. **ASSP-25** pp. 299-309 (1977).

2. A BASIC STRUCTURE FOR EFFICIENT SPEECH CODING

2.1 Introduction

In the previous chapter we recounted several hybrid coding techniques, which enable the encoding of toll quality speech at bit rates below 16 kb/s. An interesting structure for such a coder can be found when adaptive predictive coding techniques are combined with delayed decision coding techniques. Within the given structure several procedures exist for the determination of the optimum excitation sequence. The choice of a particular procedure will have a big impact on both coder performance and coder complexity. Hence we will report these procedures in separate chapters. In this chapter we will discuss the basic structure and describe the procedures for deriving the predictor parameters. Furthermore, we state the possibility of incorporating a perceptual weighting procedure in the coder structure. We also describe the procedures used for the evaluation of the coder performance.

2.2 Basic Structure

From the previous chapter we know that the redundancy in the speech signal due to the correlation among the speech samples can be effectively removed with Adaptive Prediction techniques (Section 1.5.2). The resulting prediction filters model the short-time spectral envelope due to the frequency response of the vocal tract, and the short-time spectral fine structure of voiced speech, due to the quasi-periodic nature of the signal. A remaining problem is how to find an efficient low bit rate excitation sequence that provides a toll quality synthetic signal. In Section 1.7.2 we described a delayed decision procedure embedded in an adaptive predictive system. The corresponding block diagram is repeated in Fig. 2.1.

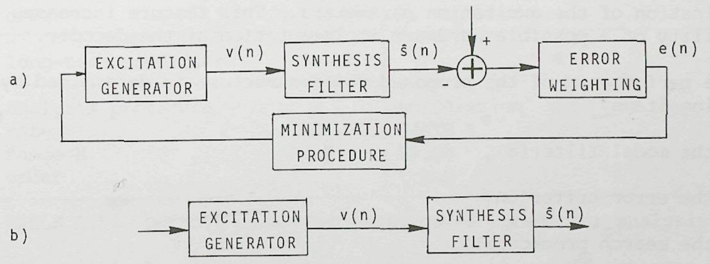


Figure 2.1. Basic structure for encoding speech with a fidelity criterion. (a) encoder, (b) decoder.

The speech signal is encoded in a two-step procedure. First, an appropriate time-varying linear filter which models the linear dependency between the speech samples is determined. The filter can be either a short-time predictor, or a cascade of a short-time and a long-time predictor. Second, for a given bit rate the optimum excitation sequence v(n) for such a filter is searched under the condition that the specified error criterion is minimized. The error criterion can be the mean squared error, but more sophisticated error criteria can be incorporated by an appropriate error weighting procedure. For a given error criterion, the optimum excitation sequence searched within adjacent frames (5 to 15 ms in duration) that will be referred to as search frames (size NSSIZE in samples). The error is minimized over frames of 5 ms to 15 ms in duration (NMSIZE samples). Search frame size and minimization frame size may differ and both could be adaptive. In either case the search frame size must be less than or equal to the minimization frame size. Fig. 2.2 shows the relationship between NSSIZE and NMSIZE.

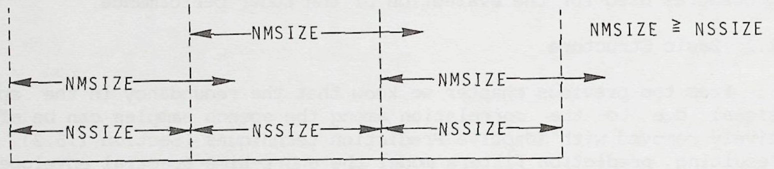


Figure 2.2. Relationship between NSSIZE and NMSIZE.

The decoding or synthesis procedure is very simple. The excitation sequence of length NSSIZE excites the synthesis filter to produce NSSIZE samples of the reconstructed speech signal $\hat{s}(n)$. Notice that the complexity of the synthesis procedure is much lower than that of the encoding or analysis procedure. Another interesting feature is that the synthesis procedure is almost independent of the procedure used for the determination of the excitation parameters. This feature increases the versatility of a possible hardware implementation of the decoder.

The performance of the proposed coder structure is determined by the following items:

1. the model filter(s),

2. the error criterion,

3. the search procedure.

In this chapter we will discuss the first two items. The definition of the excitation sequence and the corresponding search procedure will have a great influence on the coder performance and coder complexity. Hence, we will dedicate several chapters to different approaches. For example, a set of key parameters can be used to describe the excitation sequence, whereby the search procedure incorporates the determination of the

optimum values of those parameters. Such procedures are discussed in Chapter 3 (Regular-Pulse Excitation Coder) and Chapter 4 (Multi-Pulse Excitation Coder). In Chapter 5 we describe methods that use a code book approach to finding the optimal excitation sequence.

2.3 Evaluation Procedure

The performance of the coder parameters has been evaluated by means of computer simulations. To avoid quantization effects all computations were done with floating point arithmetic and none of the parameters was quantized. The experiments have been conducted with real speech data input and both objective and subjective tests have been performed to asses the final speech quality (see also Appendix A). For our experiments the signals were sampled at an 8 kHz rate with a 12 bit linear quantizer, whose performance is roughly equivalent to that of an 8 bit logarithmic quantizer (i.e. the international standard for PCM coded speech in telephony systems [CCITT, 1980]).

The range of variation of the analysis parameters was chosen in accordance with the application area of the coder (i.e. bit rates around 10 kb/s). To examine the effect of a particular parameter, several analysis procedures were performed with varying parameter settings. While varying the value of one parameter, all the other parameters were kept fixed at their default values. The default values were determined after some preliminary experiments and are considered to be a reasonable choice. Table 2.1 lists the parameters and their default values used for the predictors.

short-time predictor		
filter estimation method		correlation + Hamm. window
rate	(NRATE)	10 ms
size	(NSIZE)	25 ms
order	(p)	16
long-time predictor		
analysis procedure		correlation
rate	(NPRATE)	NRATE
range M		16,80
order		1-tap

TABLE 2.1. Default parameter settings for predictor analysis.

To limit practical problems such as computation time and storage space most tests were performed with a subset of the data set described in Appendix A. Two utterances from four different speakers (2 male and 2 female) were selected. These sentences and some of their characteristics are listed in Table 2.2.

utterance	male/ female	av.pitch (Hz)	av.energy (dB)	file
My father failed many tests	f	181	36	WD530
My father failed many tests	m	110	38	WD540
De nieuwe fiets is gestolen	f	151	42	WD505
De nieuwe fiets is gestolen	m	118	41	WD518

TABLE 2.2. Test utterance characteristics.

2.4 A Filter For Modeling The Spectral Envelope

In Chapter 1 we described that the spectral envelope of a speech segment can be approximated by the spectrum of an all-pole filter 1/A(z), where A(z) is defined as:

$$A(z) = 1 + \sum_{k=1}^{p} a_k z^{-k} \tag{2.1}$$

where a_k, $1 \leq k \leq p$ are called predictor coefficients and p is the order of the predictor. The predictor parameters are determined such that the energy of the residual, which is obtained by filtering the speech signal through the filter A(z), is minimal. The resulting set of equations are called the normal equations and can be solved very efficiently. For example, the covariance method can be solved by Cholesky decomposition. The auto-correlation method provides an even more efficient solution procedure, due to the Toeplitz property of the auto-correlation matrix (see also Chapter 8). Both methods use the covariance sequence as an intermediate quantity. In another method, called the Burg method [Burg, 1975], the predictor coefficients are obtained directly from the speech samples. Both auto-correlation and Burg methods are guaranteed to yield stable filters. The covariance method is not guaranteed to yield stable filters every time, but can be modified [Dickinson, 1978] [Atal, 1982] Co do so (modified or stabilized covariance method). Although the three methods mentioned differ in theoretical formulation, the practical results obtained with these methods are quite similar. computational considerations, such as numerical stability and the required analysis frame size (NSIZE) are more important. Some of the differences between the auto-correlation method and the covariance method are examined in [Chandra and Lin, 1974], and a description of the performance of the Burg method as compared to that of the auto-correlation or covariance method is described in [Gray and Wong, 1980]. The methods described are usually applied to data segments of fixed length. Recently proposed methods [Lee et al., 1981] [Deprettere, 1982] [Friedlander, 1982] estimate the predictor coefficients continuously and update the coefficients every time a new speech sample is available. These adaptive algorithms tend to be more complicated than the previously described non-adaptive methods, and have thus far not been shown to produce any perceptible improvement in speech applications.

The predictor coefficients $\{a_k\}$ are determined such that the power of the prediction residual is minimized over frames whose duration may

vary from 10 to 30 ms. The coefficients are updated periodically (once every 10 to 30 ms) to follow the non-stationary behavior of the speech signal. We denote by NSIZE the analysis frame length in number of samples and we denote by NRATE the coefficient update rate in number of samples. For every frame of NRATE samples we obtain a set of predictor coefficients that will be used for the synthesis of NRATE samples. Hence, we could also refer to NRATE as a synthesis frame size. Fig. 2.3 shows how the different parameters are related.

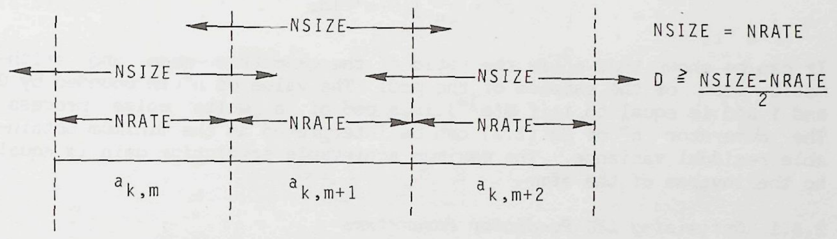


Figure 2.3. Relationship between NRATE and NSIZE.

The value of NSIZE can be equal to NRATE (non-overlapping analysis frames) or greater than NRATE (overlapping analysis frames). Overlapping analysis frames are used to get a smoother transition between coefficient sets of adjacent frames. The actual rate required for transmission of the filter coefficients forms a compromise between quality and bit rate. The number of bits required to code a set of predictor coefficients depends on the number of coefficients and the quantization method used. Typical values for 16 coefficients using optimal scalar quantization methods [Viswanathan and Makhoul, 1975] [Gray and Markel, 1976] vary from 6000 b/s to 3000 b/s for update rates of respectively 10 and 20 ms (see also Chapter 6).

The *number of predictor coefficients* or filter order forms another trade-off between allowed bit rate and desired resolution. In [Markel and Gray, 1976, Chapter 4], it was shown that to adequately represent the vocal tract, the total delay time of the A(z) filter must be equal to twice the time required for sound waves to travel from the glottis to the lips, that is, 2L/c, where L is the length of the vocal tract and c is the speed of sound. For example, the representative values c = 34 cm/ms and L = 17 cm result in a necessary delay of 1 ms. When the glottal and lip radiation characteristics and the imperfection of the all-pole model are also taken into account, a good rule of thumb for the estimation of the order is to take the sampling frequency (in kHz) plus 4 or 5. For an 8 kHz sampling frequency this leads to a 12 or 13-th order filter. To assure robustness for a wide variety of speakers, a commonly used value is 16.

To estimate the predictability of a waveform we describe the shape of a spectrum with a spectral flatness measure (sfm) [Markel and Gray, 1976]

$$\alpha^2 = (\exp[\frac{1}{2\pi}\int_{-\pi}^{+\pi} \ln S(e^{j\omega})d\omega])/\sigma^2 = n^2/\sigma^2 \qquad (2.2)$$

where σ^2 is defined as the variance of the signal s(n) with power spectral density (psd) function $S(e^{j\omega})$

$$\sigma^2 = \frac{1}{2\pi}\int_{-\pi}^{+\pi} S(e^{j\omega})d\omega \qquad (2.3)$$

It can be shown that α^2 is the ratio of the geometric mean and arithmetic mean of the samples of the psd. The value of α^2 is bounded by 0 and 1 and is equal to 1 if $S(e^{j\omega})$ is a psd of a white noise process. The numerator n^2 of Eq.(2.2) can be interpreted as the minimum obtainable residual variance. The maximum achievable prediction gain is equal to the inverse of the sfm.

2.4.1 *Optimizing LPC Predictor Parameters*

LPC analysis attempts to model the spectrum of the input signal by assuming that speech is produced by exciting a linear all-pole filter with white noise. If this filter is excited by a periodic train of impulses with fundamental frequency f_0, then the spectrum of the output signal will be a line spectrum with spectral values at only the harmonics of this fundamental frequency f_0. For high pitched voices, the harmonic spacing is too large to provide an adequate sampling of the spectral envelope. If we apply LPC analysis to this spectrum, we observe discrepancies between the original and the model spectra such as merging or splitting of pole pairs, and increasing or decreasing of pole frequencies and bandwidths [Atal, 1975]. As the fundamental f_0 decreases, these discrepancies also decrease. As a result the filter parameters will be biased for voiced speech, especially for female speakers. Algorithms that try to reduce this pitch-dependent bias on the filter parameters usually make an estimation of the input pulses and use this estimated input as model input [Miyanaga et al., 1982]. In [Singhal and Atal, 1983] the multi-pulse signal was used as an estimation for the input signal, based on the observation that the multi-pulse input contains the same periodicity as the original speech. Reformulating the LPC analysis equations and solving for the prediction parameters result in a set of filter parameters that are to a large extent unbiased. After finding the new predictor parameters, the multi-pulse excitation must be recomputed. Instead of recomputing both locations and amplitudes of the pulses, only the amplitudes are recomputed. The whole process can be repeated until some specified error criterion is met. It was reported that as many as 10 to 20 iterations were required to obtain any perceivable improvement [Atal, private communication]. Besides the enormous complexity, precautions have also to be taken to ensure the stability of the synthesis filters produced by this modified analysis procedure.

2.4.2 *Filter Structures*

To implement the all-zero filter A(z) and the all-pole filter 1/A(z) we can use either direct form filters [Oppenheim and Schafer, 1975], or lattice filters [Gray and Markel, 1973] [Makhoul, 1978]. Direct form filters are direct realizations of the differential equations that describe the input-output relation of the filter. Fig. 2.4 shows the direct form and lattice form structures for the all-pole analysis filter 1/A(z).

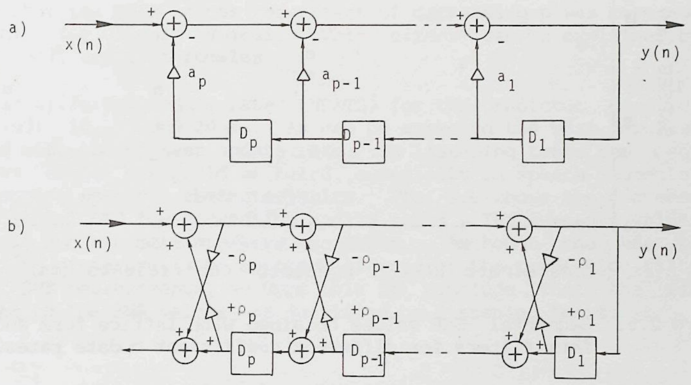


Figure 2.4. Direct form all-zero (a) and all-pole (b) filters.

Lattice filters are canonical realizations of the filter transfer function. The filter coefficients, the so-called reflection coefficients, are all bounded by one, and can be obtained from the direct form filter parameters by a recursive procedure [Markel and Gray, 1976, Chapter 5]. One major disadvantage of direct form structures is their sensitivity to finite word length computations. Lattice filters are more suitable for fixed word length implementations. An additional advantage is that lattice filters use coefficients which can be quantized very efficiently [Markel and Gray, 1976]. When disregarding finite word length effects, both types of filter structures realize the same transfer function; they are equivalent realizations. However, for speech synthesis applications, where the filter coefficients are refreshed at regular intervals, both filter structures show a different behavior. The refreshment of the filter coefficients is organized as follows. For every frame of NRATE samples the filter coefficients are kept fixed. Before the samples of the next frame are computed, all the filter coefficients of the previous frame are replaced by the coefficients belonging to the new frame. To ensure continuous filtering, the states of the filters are not cleared when a new set of coefficients is dumped. In Appendix B a more detailed discussion of the different coder structures is given. The differences in behavior for time-varying coefficients are noticeable when the two different structures are used in the analysis and synthesis procedure of Fig. 2.1. Fig. 2.5 shows the SNRSEG values of the reconstructed speech produced by the MPE coder using either direct form or lattice form filters. The coefficients were updated at regular intervals.

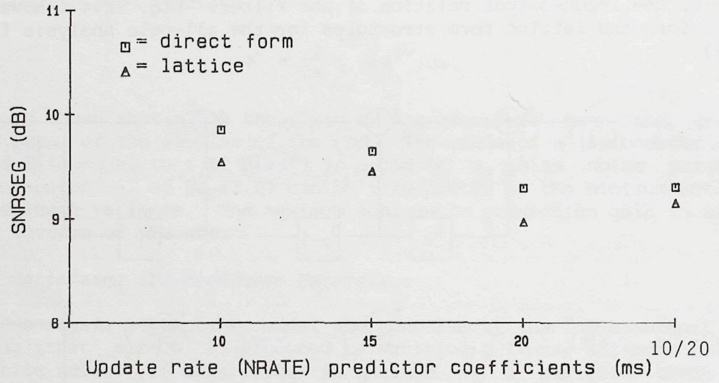


Figure 2.5. Segmental SNR values obtained with lattice form and direct form filters for different coefficient update rates.

From this figure we conclude that direct form filters show a slightly better performance than lattice form filters. Note that all computations were done with floating point arithmetic, and that the differences between the filter structures will probably become less significant if finite precision arithmetic is used. To reduce the transition effects, we applied linear interpolation between adjacent coefficient sets. The interpolation was in both cases performed between the reflection coefficients to assure stable synthesis filters. Fig. 2.5 also shows the results obtained from interpolating once every 10 ms between coefficients that were obtained at a 20 ms rate (10/20). We see no difference in SNR for the direct form filters, but for the lattice form filters there is some improvement. Despite the somewhat better performance of the direct form realizations, we continue our experiments using only lattice realizations.

2.4.3 *Predictor Parameters*

To get some insight into the effect of the predictor parameters on the coder performance, we used the Multi-Pulse Excitation (MPE) method (see also Chapter 4). The possible compensation effect of the error minimization procedure for small inaccuracies in the spectral envelope (as defined by the predictor) was neglected. MPE analysis was performed with predictor parameters obtained with different *predictive estimation algorithms* (auto-correlation method, stabilized covariance method and

Burg method). In listening tests no difference could be heard between the different methods. Even the results obtained with the stabilized covariance method with high-frequency correction [Atal and Schroeder, 1979], which was used in the original paper on MPE coding [Atal and Remde, 1982], were not really different from the results obtained with the auto-correlation method.

The *predictor order* p was varied between 8 and 16. Listening tests revealed that no significant differences could be heard between the reconstructed speech signals for values of p larger than 12. For values of p less than 10 a slight degradation could be heard. Furthermore, we noticed that for male voices the effect of decreasing p was more noticeable than for female voices. This effect can be explained by the shorter vocal tract of females.

Three different *update rates* (NRATE) for the predictor coefficients were used: 10, 15 and 20 ms. As can be expected the best results were obtained with the highest update rate. In listening tests the degradation was small but could be heard, especially in speech segments with fast changing spectral characteristics. Fig. 2.6 shows the decrease in the segmental SNR for decreasing update rates. The averaged values for two female and two male speakers are shown. Although one has to be careful to interpret the spectral effects of the prediction filter in terms of SNR measurements, we were able to conclude that the ranking according to the SNR values was justified by listening tests.

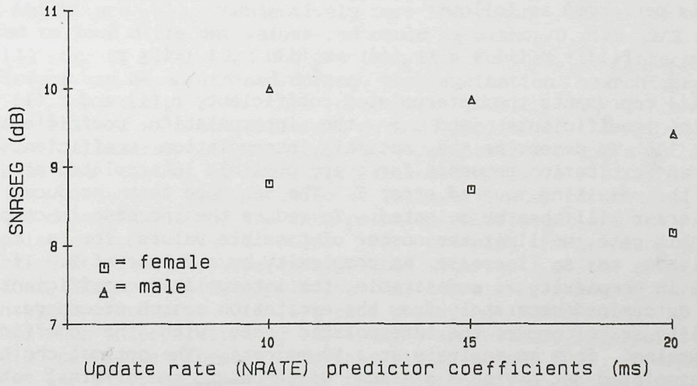


Figure 2.6. Segmental SNR values for different update rates of the predictor parameters.

2.4.4 *Interpolation Of The Filter Coefficients*

In the previous sub-section we described the effect of the update rate of the filter coefficients on the speech quality, and concluded that an update rate of 20 ms was generally not optimal. In this section we describe an approach for obtaining a higher update rate without increasing the bit rate. The spectral envelope of the speech signal is described by the filter 1/A(z). The spectral characteristics of speech vary in time, and therefore the filter coefficients need to be updated at regular time instants. However, these fluctuations are relatively slow, which means that the filter parameters can be determined at a finite rate. Thus the filter coefficients are non-constant functions of time with a bandwidth of typically 50 to 100 Hz. If these functions are sampled at a 20 ms rate, we can obtain the missing samples (for the 10 ms rate) by interpolation. The filter coefficients can be represented by a set of reflection coefficients $\{\rho(i)\}$, which have the nice property that they are bounded by one. This means that we can apply linear interpolation to the reflection coefficients, whereby the interpolated filters are guaranteed to be stable. The idea is to analyze the speech at a 20 ms rate and synthesize the speech at a 10 ms rate, whereby the missing sets are obtained by linear interpolation between the available sets. If we choose an appropriate interpolation coefficient, we obtain virtually a 10 ms rate, while transmitting at a 20 ms rate. When we embed the interpolation procedure in the excitation search procedure, we can define the optimum interpolated set to be the one that minimizes the mean-squared weighted error E. The interpolation of the filter coefficients is performed as follows:

$$\hat{r}(i) = r_1(i) + \sigma\,(r_2(i) - r_1(i)) \quad ,\ 1 \le i \le p \qquad (2.4)$$

where $\hat{r}(i)$ represents the interpolated coefficient, $r_1(i)$ and $r_2(i)$ the available coefficients and σ the interpolation coefficient (σ ($0 \le \sigma \le 1$). To determine the optimal interpolation coefficient, we search an excitation sequence for every possible interpolated set, and compute the resulting squared error E. The sequence that produces the minimum error will then be selected. To reduce the increase in complexity and bit rate, we limit the number of possible values for σ to 4, which leads to an increase in complexity by a factor of 2. If this increase in complexity is undesirable, the interpolation coefficient has to be determined separately from the excitation search procedure. One possibility is to compare the interpolated sets with the coefficient sets obtained from an analysis at a 10 ms rate. The optimal choice is the interpolated set having a minimum "distance" to the original set. A commonly used distance measure between LPC filter sets is the log likelihood ratio [Itakura, 1975]. This distance measure is given in dB by

$$d_1 = 10\,\log_{10}\,(\,E_2\,/\,E_1)\quad \mathrm{dB} \qquad (2.5)$$

where E_1 and E_2 are the mean-squared values of the residual, obtained from the current frame of speech, when the coefficients of the current frame and the interpolated coefficients, respectively, are used.

Instead of using the distance measure d_1 in Eq.(2.5), one may use a simpler measure such as a weighted Euclidian distance between two sets of coefficients. In [Viswanathan et al., 1982] a Euclidian distance measure between Log Area Ratios (LAR's) was proposed. The log area function is given by

$$g(i) = \log_{10}\,[(1+\rho(i))\,/\,(1-\rho(i))] \quad ,1 \le i \le p \qquad (2.6)$$

where $\{\rho(i)\}$ represent the reflection coefficients. The weighted Euclidian distance between the original set $\{g_i\}$ and the interpolated set $\{\hat{g}_i\}$ is defined as

$$d_2 = \sum_{i=1}^{m} w(i)\,[g(i) - \hat{g}(i)]^2 \,/\, \sum_{i=1}^{m} w(i) \qquad (2.7)$$

where $m \le p$ and $\{w_i\}$ is a set of non-negative weights chosen to reflect the relative importance of the different LAR's to the perceived speech quality. In [Viswanathan et al., 1982] m=4 was found to be appropriate and the corresponding weights were: 1.3, 1.2, 1.1 and 1.0. In the same reference it was also noted that there was no difference in speech quality whether the log likelihood ratio measure or the LAR Euclidian distance was used. If the LAR's are also used as transmission parameters, the LAR distance measure is computationally less expensive than the log likelihood ratio measure. Interpolation can be performed on either the reflection coefficients (R) or the Log Area Ratios (L). The LAR representation emphasizes coefficients with absolute values close to one, which are also subjectively more important. The value of σ was limited to four different values and could be either 0, 1/3, 2/3 and 3/3 (set 1) or 0, 1/4, 2/4 and 3/4 (set 2). Further, the interpolation coefficient can be determined during the excitation search procedure (IN) or separately from this procedure (OUT). Fig. 2.7 shows the results obtained with the different methods applied to the MPE coder for different speakers. These figures were obtained using a much larger data base than the set of four test sentences of Table 2.2. The interpolation procedure used is coded as follows: filter parameters, method used for determination of the interpolation coefficient and the set of interpolation coefficients used. For example, RIN1 means that the reflection coefficients were interpolated with set 1 of the coefficients and that the optimal interpolation coefficient was selected during the MPE analysis procedure. Fig. 2.7 also shows the SNRSEG values of speech obtained with interpolation by $\sigma=\frac{1}{2}$ (LOUT0) and without interpolation (10 ms and 20 ms update rates). From this figure we see that interpolation between LAR's is somewhat better than interpolation between reflection coefficients and that interpolation with set 1 (0, 1/3, 2/3, and 3/3) produced the best results. We also see that determining the interpolation coefficient with a distance measure gives no improvement over the use of a fixed interpolation coefficient.

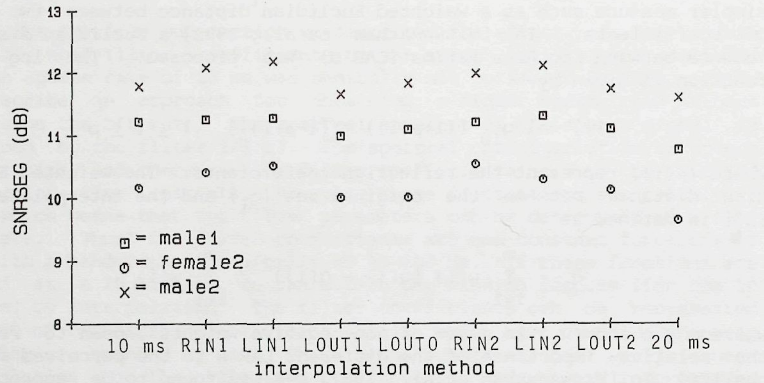


Figure 2.7. SNR values obtained by interpolation of coefficients during MPE analysis (LIN1, RIN1, LIN2 and RIN2) and before MPE analysis (LOUT0,LOUT1 and LOUT2) for different speakers.

An explanation for the improvement in SNR for a fixed interpolation factor was given in the previous sub-section. What is striking is that if the interpolation coefficient is determined during the MPE analysis (IN), the final speech quality is better than without interpolation (10 ms). The variation of the interpolation coefficient σ is illustrated in Fig. 2.8 for LIN1, LOUT1, LIN2 and LOUT2, respectively. From this figure we see that the values of σ determined during the MPE analysis are quite different from those determined separately from the MPE analysis.

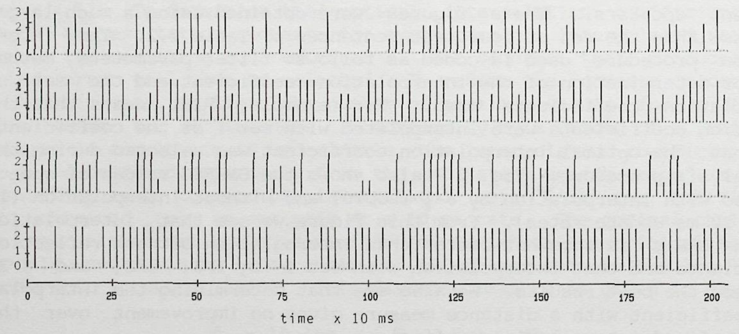


Figure 2.8. Interpolation coefficient for a single utterance for the methods LIN1, LOUT1, LIN2 and LOUT2, respectively.

In comparative listening tests we noted that interpolation always

improves the speech quality.

2.5 A Filter For Modeling The Spectral Fine Structure

In Chapter 1 we described how the periodicity in the speech signal due to pitch pulses can be estimated by a pitch predictor. The general form of a pitch predictor is

$$P(z)-1 = \beta_1 z^{-(M-1)} + \beta_2 z^{-M} + \beta_3 z^{-(M+1)} \quad (2.8)$$

where M represents the distance between adjacent pitch pulses and β_1, β_2, and β_3 are pitch predictor coefficients. This predictor is called a three-tap predictor. If we set β_1 and β_3 to zero, then $P(z)-1$ will reduce to a one-tap predictor. A short-time predictor $1-A(z)$ and a long-time pitch predictor $1-P(z)$ can be combined serially in either order to produce a cascaded predictor. Studies on APC schemes [Atal and Schroeder, 1979] suggest that it is preferable to use the short-time predictor before the long-time predictor. This means that the pitch predictor uses the residual signal as input (Fig. 2.9).

$$\begin{tikzcd} \arrow[r, "s(n)"] & \boxed{A(z)} \arrow[r, "r(n)"] & \boxed{P(z)} \arrow[r, "r'(n)"] & {} \end{tikzcd}$$

Figure 2.9. Cascade of short-time predictor and long-time predictor.

The parameters $\{\beta_i\}$ and M are chosen to minimize the mean squared error E'. For the one-tap predictor ($\beta = \beta_2$) E' is given by

$$E' = \sum_n [r'(n) - \beta\, r'(n-M)]^2 \quad (2.9)$$

Leaving the range of summation unspecified for the moment, the minimization of E' with respect to β yields

$$\beta = \sum_n [r'(n)\, r'(n-M)] \,/\, \sum_n [r'(n-M)]^2 \quad (2.10)$$

Substituting this result in Eq.(2.9) gives an expression for E' as a function of M alone:

$$E' = \sum_n [r'(n)]^2 - \frac{[\sum_n r'(n)\, r'(n-M)]^2}{\sum_n [r'(n-M)]^2} \quad (2.11)$$

In Eq.(2.11) only the second term depends on M, so that maximizing this term minimizes the error E'. The approach is to compute

$$A(M) = \frac{[\sum_n r'(n)\, r'(n-M)]^2}{\sum_n [r'(n-M)]^2} \qquad (2.12)$$

for all values of M within a specified range, and then select the value for which A(M) is maximum. With M known, ß may now be computed by the use of Eq.(2.10).

So far, the range of the summation has been left open. If we use an infinite range of summation and window the data by a finite sized window such that the data outside the window equals zero, we obtain the auto-correlation method. The denominator of Eq.(2.12) becomes fixed and ß will always be less than one, so that the pitch synthesis filter is guaranteed to be stable. To take into account the reduced number of terms in the summation in Eq.(2.12) for increasing values of M, we can normalize A(M) to the number of summed terms. Eq.(2.12) can be further simplified, by only considering either positive or negative correlation terms. This approach also ensures that ß will have only positive or negative values, thereby facilitating quantization procedures. Note that this is not a real limitation to the performance of the predictor. Having a pitch pulse sequence whose sign is alternately positive and negative is very unlikely . If we use a window containing NPSIZE samples, Eq.(2.12) will reduce to

$$A'(M) = \frac{1}{NPSIZE-M} \sum_{n=M+1}^{NPSIZE} r'(n)\, r'(n-M) \qquad (2.13)$$

To make reliable estimations of the pitch period, at least two pitch pulses must fall within this analysis frame. This means that the largest period that can be found is determined by NPSIZE/2. Usually, the minimum value of M is also bounded (e.g. M=16, which corresponds to a 500 Hz pitch frequency at 8 kHz), so that the value of M in Eq.(2.12) varies from M=16 up to M=NPSIZE/2.

Another approach is the covariance method, which minimizes the error over a fixed number of elements, say NPSIZE, and the summation in Eq.(2.9) ranges from n=-M+1 to n=NPSIZE. A disadvantage of this method is that ß can become greater than one, resulting in an unstable pitch synthesis filter. However, in practice instability of the pitch synthesis filter seldom occurs, and can be entirely prevented by not allowing ß to be greater than one. An advantage is that the maximum value of M is not restricted by the value of NPSIZE. The best results were obtained with the covariance method, but a satisfying result could also be obtained by using Eq.(2.13) for determining the delay factor M, and using Eq.(2.10) with a summation range from M+1 to NPSIZE to compute ß (see also Chapter 6, Fig. 6.10).

The three-tap pitch predictor uses the same procedure as the one-tap to find the value of M. The minimization procedure then leads to a set of simultaneous linear equations in the three unknowns β_1, β_2, and β_3.

The solution to these equations does not guarantee a stable pitch synthesis filter for both the auto-correlation and covariance methods. In the small number of cases where instability occurs, one could use a one-tap predictor (with ß < 1), which is always guaranteed to be stable. On the other hand, instability for certain time instants is not necessarily harmful for the quality of the reconstructed speech. In general, the performance of the three-tap pitch predictor is better than that of a one-tap, at the expense of a higher bit rate. Fig. 2.10 shows an example of the speech signal, the residual signal and the remaining signal after 1-tap and 3-tap pitch prediction.

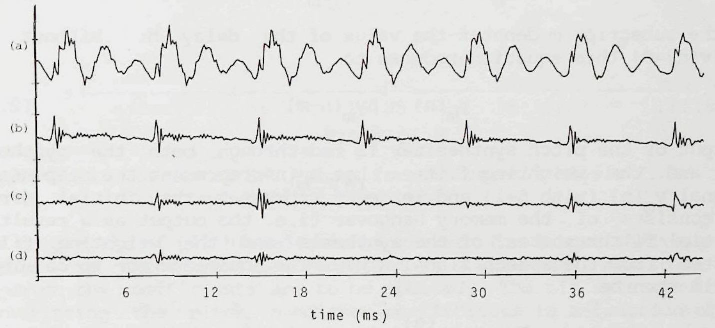


Figure 2.10. (a) Speech signal, (b) residual signal, (c) residual after 1-tap pp, (d) residual after 3-tap pp.

The weighted error minimization procedure of Fig. 2.1 does not really change upon the use of a pitch predictor. The linear filter is replaced by a cascade of a pitch synthesis filter and a speech synthesis filter. The resulting block diagram is shown in Fig. 2.11.

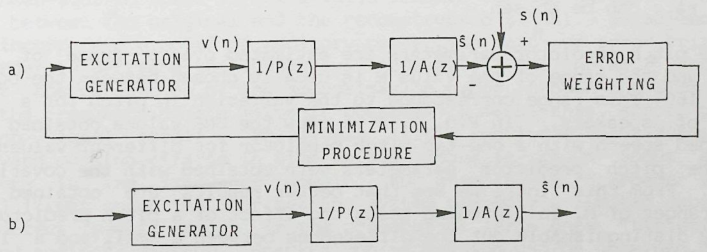


Figure 2.11. Block diagram of the structure of Fig. 2.1 with pitch prediction. (a) encoder, (b) decoder.

For the coder structure of Fig. 2.11, the pitch predictor parameters can be determined either apart from the weighted error minimization procedure [Kroon and Deprettere, 1984], or during the weighted minimization procedure [Singhal and Atal, 1984]. In the latter case the pitch predictor will directly contribute to the minimization of the weighted error. In this approach the predictor parameters are computed prior to the determination of the excitation sequence. In the case of a 1-tap pitch predictor we can write the output y(n) of the pitch synthesizer, with input v(n), as

$$y_m(n) = v(n) + \beta y_m(n-m) \qquad (2.14)$$

where the subscript m denotes the value of the delay M. Without any input (v(n)=0) this equation reduces to

$$y_m(n) = \beta y_m(n-m) \qquad (2.15)$$

The output of the pitch synthesizer is fed through both the synthesis filter and the weighting filter. Let $z_m(n)$ represent the response to the signal $y_m(n)$ (with β=1) and let $e^{(0)}$ represent the initial error, which consists of the memory hangover (i.e. the output as a result of the initial filter states) of the synthesis and the weighting filter subtracted from the speech signal s(n). The squared error to be minimized will then be

$$E = \sum_n (e^{(0)}(n) - \beta z_m(n))^2 \qquad (2.16)$$

This equation is equivalent to Eq.(2.9) and can be solved in a similar way. The final speech quality is not really different for both procedures, but the latter procedure has some advantages when the excitation signal is quantized separately (see also Chapter 6). A disadvantage of this last procedure is that the coefficient update rate is equal to the search frame size, and can be quite high (e.g. 5 ms for the RPE coder). If the parameters are determined from the residual an arbitrary update rate can be chosen.

The pitch predictor parameters are estimated over a frame of size NPSIZE and the range of the value M is usually chosen between the bounds 16 and 160. This range corresponds to the variation in pitch for a wide range of speakers. In Fig. 2.12 we show the SNR values obtained from MPE coded speech with a one-tap pitch predictor for different values of M. The pitch predictor parameters were obtained with the covariance method. From this figure we see that better results are obtained for large ranges of M. In listening tests the effect of a pitch predictor is clearly distinguishable but the differences between a small and a large range of M are less noticeable.

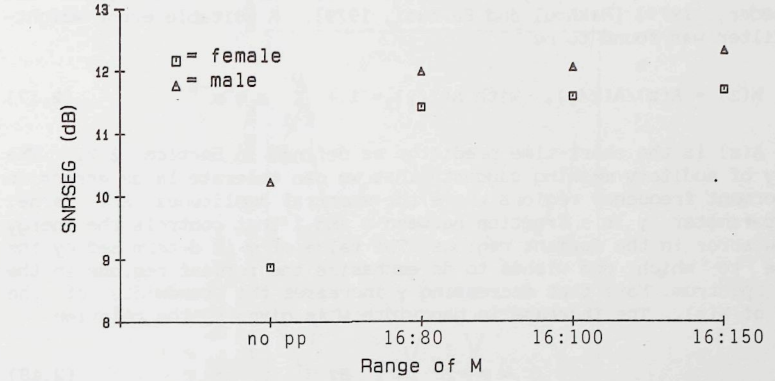


Figure 2.12. Segmental SNR values for different analysis frame sizes of the one-tap pitch predictor.

The complexity of pitch prediction increases for larger analysis frames. Similar to the short-time prediction a rate (NPRATE) has to be defined at which the coefficients are to be updated. The bit rate required for transmitting the pitch predictor coefficients is a function of NPRATE and the range of M.

2.6 Selection Of An Error Criterion

The coder structure of Fig. 2.1 minimizes the error between the original signal s(n) and the decoded signal $\hat{s}(n)$ according to a suitable error criterion. A commonly used error criterion in speech research is the mean squared error, which has an appealing mathematical trackability and provides an adequate performance. However, at rates below 16 kb/s, the mean-squared error is not a valid measure for the perceptual difference between the original and the reconstructed signal. We somehow have to incorporate a model of auditory perception into our error criterion, in order to decrease the subjective loudness of the noise. Knowledge about the perception of the approximation error in both the time domain and the frequency domain makes it possible to mask this error by the speech signal itself. Relatively little knowledge about forward and backward masking effects in time exists. More is known about masking effects in the frequency domain. For example, in [Schroeder et al., 1979] it was shown that our hearing system has only a limited capability to detect small errors in the presence of large ones. This effect is called auditory masking. Spectral noise shaping can be obtained by distributing the noise power in relation to the input speech power. This task can be accomplished by minimizing a frequency weighted error in the block diagram of Fig. 2.1. Note that such a shaping procedure does not affect the bit rate or the complexity of the synthesis procedure. Only the encoder complexity is increased. The use of noise shaping procedures was thoroughly investigated for APC schemes [Atal and Schroeder, 1979] [Makhoul and Berouti, 1979]. A suitable error weighting filter was found to be

$$W(z) = A(z)/A(z/\gamma), \quad \text{with } A(z/\gamma) = 1 + \sum_{k=1}^{p} a_k \gamma^k z^{-k} \tag{2.17}$$

where A(z) is the short-time predictor as defined in Section 2.2. The theory of auditory masking suggests that we can tolerate large errors in the formant frequency regions where the spectral amplitudes are large. The parameter γ is a fraction between 0 and 1 that controls the energy of the error in the formant regions. The value of γ is determined by the degree to which one wishes to de-emphasize the formant regions in the error spectrum. Note that decreasing γ increases the bandwidth of the poles of W(z). The increase in bandwidth w is given by the relation

$$w = -\frac{f_s}{\pi} \ln \gamma \quad \text{Hz} \tag{2.18}$$

where f_s is the sampling frequency in Hz. The optimum value of γ must be determined by suitable listening tests. A commonly used value is 0.80 at 8 kHz. This corresponds to an increase in bandwidth of approximately 500 Hz. Fig. 2.13 shows an example of the spectral speech envelope and the frequency response of the corresponding error weighting filter. The use of the proposed error weighting filter W(z) makes it possible to give an alternative representation of the coder structure of Fig. 2.1. In this figure the excitation signal v(n) is fed through an all-pole filter 1/A(z), and the resulting synthetic signal ŝ(n) is subtracted from the speech signal s(n). The difference signal s(n)-ŝ(n) is fed through the weighting filter A(z)/A(z/γ) [see Fig. 2.14(a)]. All filters are linear, and by assuming the weighting filter to be a cascade of an all-zero filter A(z) and an all-pole filter 1/A(z/γ), we obtain the configuration shown in Fig. 2.14(b). The input signal for the coder is now the residual signal r(n). The whole procedure can be considered as a weighted residual matching procedure. Note that the pitch predictor filter can be easily added to this representation.

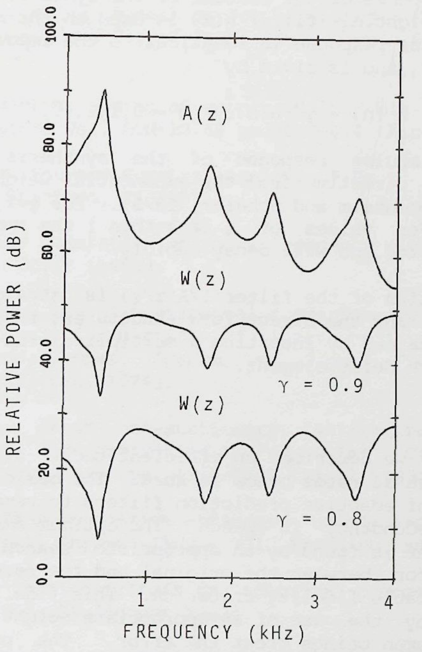


Figure 2.13. An example of the spectral speech envelope and the frequency response of the corresponding error-weighting filter for different values of γ.

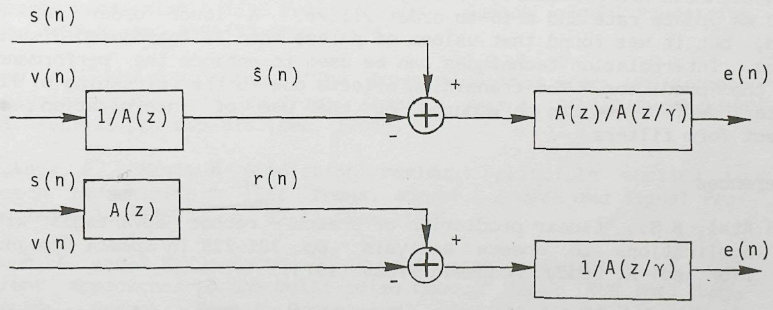


Figure 2.14. Different structures, (a) speech input, (b) residual input.

The impulse response of the cascade of the synthesis filter 1/A(z) and the error weighting filter W(z) is used in the minimization procedure. This impulse response is identical to the impulse response of the filter $1/A(z/\gamma)$, and is given by

$$h_\gamma(n) = \gamma^n\, h(n), \quad n = 0,1,2,\ldots \tag{2.19}$$

where h(n) is the impulse response of the synthesis filter 1/A(z). Eq.(2.19) follows directly from the exponential weighting property of the z-transform [Oppenheim and Schafer, 1975]. For γ=1, $h_\gamma(n)$ is identical to h(n). For values of γ less than 1 the impulse response is exponentially weighted and will decay rapidly.

The implementation of the filter $1/A(z/\gamma)$ is straightforward. For both the lattice and the direct form structures, the desired transfer function can be obtained by inserting a multiplier with a multiplication factor γ before each delay element.

2.7 Summary

In this chapter we described an efficient coder structure for the encoding of speech at rates below 16 kb/s. The basic features of this coder are the use of adaptive prediction filters to remove the long-term and short-term redundancy in speech. The optimum excitation sequence for the model filter is found by an appropriate search procedure that minimizes the error between the original and the reconstructed signal according to a suitable fidelity criterion. This fidelity criterion can be incorporated by the use of an appropriate weighting filter, which reflects the way human beings treat the error. The predictor coefficients can be obtained with different procedures, but it was concluded that the differences in signal quality were hardly perceptible between the different methods. Further, the effect of different predictor parameters such as update rate and order were investigated in combination with the MPE procedure, whereby the best results were obtained with a 10 ms update rate and a 16-th order filter. A lower order can be used, but it was found that values of p less than 12 introduced distortions. Interpolation techniques can be used to enhance the performance of the predictor. The transition effects due to the refreshing of the filter coefficients can be reduced by the use of interpolation and direct form filters.

References

Atal, B.S., "Linear prediction of speech - recent advantages with applications to speech analysis," pp. 221-229 in *Speech Recognition*, ed. R. Reddy, Academic Press (1975).

Atal, B.S. and M.R. Schroeder, "Predictive coding of speech and subjective error criteria," *IEEE Trans. Acoust., Speech, Signal Processing* Vol. **ASSP-27** pp. 247-254 (June 1979).

Atal, B.S. and J.R. Remde, "A new model of LPC excitation for producing natural-sounding speech at low bit rates," *Proc. IEEE Int. Conf. Acoust., Speech and Signal Processing*, pp. 614-617 (April 1982).

Atal, B.S., "Predictive coding of speech at low bit rates," *IEEE Trans. Communications* Vol. **COM-30**(4) pp. 600-614 (April 1982).

Burg, J., "Maximum Entropy Spectral Analysis," Ph.D. dissertation, Stanford Univ., Stanford, CA (1975).

CCITT,, "Pulse Code Modulation of voice frequency signals," Recommendations G711, CCITT (1980).

Chandra, S. and W.C. Lin, "Experimental comparison between stationary and non-stationary formulations of linear prediction applied to speech," *IEEE Trans. Acoust., Speech and Signal Processing* Vol. **ASSP-22** pp. 403-415 (Dec. 1974).

Deprettere, E.F.A., "Fast non-stationary lattice recursions for adaptive modeling and estimation," *Proc. Int. Conf. Acoust., Speech, Signal Processing*, pp. 1721-1726 (April 1982).

Dickinson, B.W., "Autoregressive estimation using residual energy ratio's," *IEEE Trans. Inform. Theory* Vol. **IT-24** pp. 503-506 (1978).

Friedlander, B., "Lattice filters or adaptive processing," *Proc. IEEE* Vol. **70** pp. 654-659 (August 1982).

Gray, A.H. and J.D. Markel, "Digital lattice and ladder filter synthesis," *IEEE Trans. Audio and Electroacoustics* Vol. **AU-21** pp. 491-500 (1973).

Gray, A.H. and J.D. Markel, "Quantization and bit allocation in speech processing," *IEEE Trans. Acoust., Speech, Signal Processing* Vol. **ASSP-24** pp. 459-473 (Dec. 1976).

Gray, A.H. and D.Y. Wong, "The Burg algorithm for LPC speech analysis/synthesis," *IEEE Trans. Acoust., Speech, Signal Processing* Vol. **ASSP-28** pp. 609-615 (Dec. 1980).

Itakura, F., "Minimum prediction residual principle applied to speech recognition," *IEEE Trans. Acoust., Speech and Signal Processing* Vol. **ASSP-23** pp. 67-72 (Feb. 1975).

Kroon, P. and E.F. Deprettere, "Experimental evaluation of different approaches to the multi-pulse coder," *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing*, pp. 10.4.1-10.4.4 (March 1984).

Lee, D., M. Morf, and B. Friedlander, "Recursive least squares ladder estimation algorithms," *IEEE Trans. Circuits and Systems* Vol. **CAS-28** pp. 467–481 (June 1981).

Makhoul, J., "A class of all-zero lattice digital filters: properties and applications," *IEEE Trans. Acoust., Speech, Signal Processing* Vol. **ASSP-26** pp. 304–314 (1978).

Makhoul, J. and M. Berouti, "Adaptive noise spectral shaping and entropy coding in predictive coding of speech," *IEEE Trans. Acoust., Speech, Signal Processing* Vol. **ASSP-27** pp. 247–254 (1979).

Markel, J.D. and A.H. Gray, *Linear Prediction of speech*, Springer Verlag, Berlin (1976).

Miyanaga, Y., N. Miki, N. Nagai, and K. Hatori, "A speech analysis algorithm which eliminates the influence of pitch using the model reference adaptive system," *IEEE Trans. Acoust., Speech and Signal Processing* Vol. **ASSP-30**(1) pp. 88–95 (Feb. 1982).

Oppenheim, A.V. and R.W. Schafer, *Digital signal processing*, Prentice Hall, Englewood Cliffs, New Jersey (1975).

Schroeder, M.R., B.S Atal, and J.L. Hall, "Optimizing digital speech coders by exploiting masking properties of the human ear," *J. Acoust. Soc. Amer.* Vol. **66** pp. 1647–1652 (Dec. 1979).

Singhal, S. and B.S. Atal, "Optimizing LPC filter parameters for multi-pulse excitation," *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing*, pp. 781–784 (March 1983).

Singhal, S. and B.S. Atal, "Improving performance of multi-pulse LPC coders at low bit rates," *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing*, pp. 1.3.1–1.3.4 (March 1984).

Viswanathan, R. and J. Makhoul, "Quantization properties of transmission parameters in linear predictive systems," *IEEE Trans. Acoust., Speech, Signal Processing* Vol. **ASSP-23** pp. 124–321 (June 1975).

Viswanathan, V.R., J. Makhoul, R.M. Schwartz, and A.W.F. Huggins, "Variable Frame Rate transmission: a review of methodology and application to narrow-band LPC speech coding," *IEEE Trans. Communications* Vol. **COM-30** pp. 674–686 (April 1982).

3. REGULAR-PULSE EXCITATION CODER

3.1 Introduction

In this section we present a new speech coding method, which can provide toll quality speech at medium bit rates. The coder produces synthetic speech by exciting a time-varying linear filter with an excitation sequence that has a regular structure. With respect to this regular excitation structure, the difference between the original and the reconstructed signal is minimized. Because of the regular structure of the excitation, we refer to this coder as the Regular-Pulse Excitation (RPE) coder [Deprettere and Kroon, 1985]. The problem of finding the excitation sequence can be phrased in terms of solving a set of linear equations, which can be solved in an efficient way. In the remainder of this section we describe the RPE coder and the procedure for finding the excitation sequence. Further, we describe the effect of the various analysis parameters on the synthetic speech quality. We discuss modifications of the basic RPE scheme and their impact on both the quality of the synthetic speech signal and the complexity of the encoding algorithm. Finally, we shall discuss the coder performance according to objective and subjective tests.

3.2 Regular-Pulse Excitation Coding

In the previous chapter we described an efficient basic structure for a speech coder, in which the redundancies of the speech signal can be effectively modeled with linear filters. The main problem is how to find an appropriate excitation sequence for the modeling filters. Rather than searching for the "optimum" sequence from a large selection of possible sequences, we can use a parametric approach whereby the excitation signal is described by a small set of parameters, and the search procedure involves the search for these parameters. We propose the following structure for the excitation sequence. Within each excitation frame a limited number of samples may have a non-zero amplitude, and all the remaining samples within the frame will have by definition a zero amplitude. Moreover, these non-zero samples, which we refer to as pulses, are regularly spaced in time with a spacing of NS samples. Within a frame the first pulse can be placed on NS different locations, but the number of pulses within a frame and the spacing between the pulses must remain constant. Thus, for each excitation frame there are NS possible excitation patterns. For every pattern the optimal pulse amplitudes have to be determined, such that the weighted mean-squared error between $s(n)$ and $\hat{s}(n)$ is minimized. Finally, the excitation sequence that produces a minimum error is selected. Fig. 3.1 shows a set of excitation patterns for a frame containing 18 samples and a spacing of NS=3. In this figure the locations of the pulses are marked by 1's, while the zero valued samples are represented by dots. Fig. 3.2 gives an example of the original, synthetic and excitation waveforms as produced by the RPE coder. The corresponding power spectra of the speech signal $s(n)$ and the synthetic signal $\hat{s}(n)$ are shown in Fig. 3.3

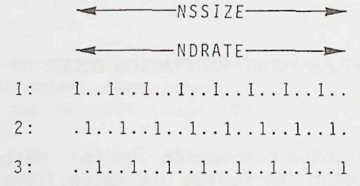


Figure 3.1. RPE excitation signal with NDRATE=NSSIZE=24, and NS=3.

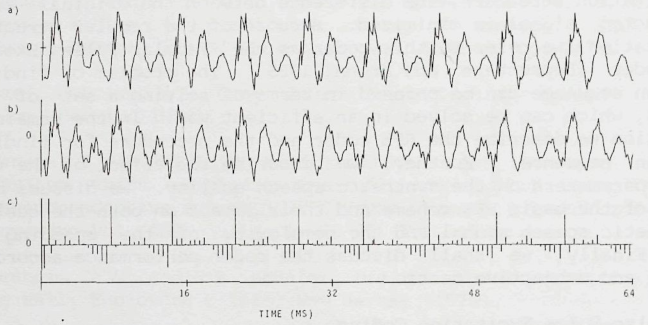


Figure 3.2. Waveforms of the original speech (a), the reconstructed speech (b), and the excitation signal (c) in the RPE coding procedure.

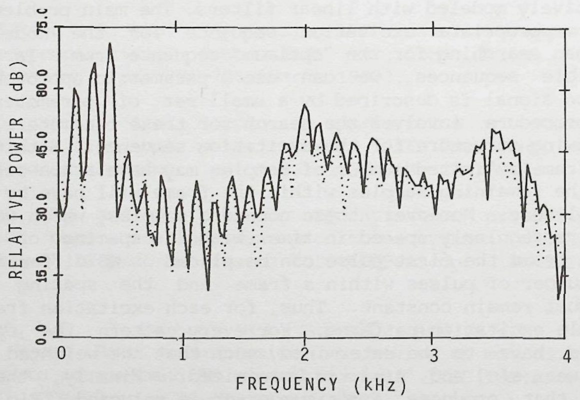


Figure 3.3. Power spectra of the original speech segment (solid line) and the reconstructed speech segment (dashed line). The spectra were obtained from 32 ms segments using a Hamming window.

To give an impression of the signal-to-noise ratio over an complete utterance, we show in Fig. 3.4 the segmental SNR computed every 10 ms

for the utterance "a lathe is a big tool" spoken by both a female and a male speaker.

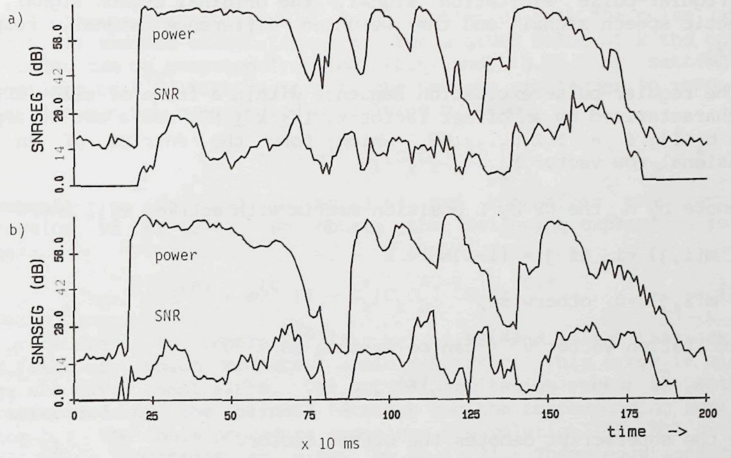


Figure 3.4. Segmental SNR for successive time frames for a female speaker (a), and a male speaker (b). The upper curve represents the relative speech power + 15 dB.

For these examples we used the parameter settings as listed in Table 3.1.

3.3 Error Minimization Procedure

3.3.1 *Definitions*

As described above, the excitation sequence within a frame of NDRATE samples is formed by a set of uniformly spaced pulses (a "grid") with spacing NS (Fig. 3.1). The origin of such an excitation "grid" is given relative to the beginning of the current frame by the offset parameter k, with $1 \leq k \leq NS$. The spacing NS between the pulses may not exceed the frame boundary ($1 \leq NS \leq NDRATE$), and NDRATE/NS must be an integer ratio. The value of k is adjusted every NDRATE samples to minimize the weighted squared error over frames of size NMSIZE (see also Chapter 2). A value of k not equal to one, disturbs the regularity of the excitation signal between adjacent frames; therefore we refer to NDRATE as the disturbance rate. The number of pulses per search frame (NP) is given by the ratio of the search frame size NSSIZE and the pulse spacing NS. The pulse rate per second, which is closely related to the bit rate, is mainly determined by NS. NS has to be chosen optimally for the allowed bit rate.

In the following we explain the procedure for determining the excitation sequence. We represent a sampled time signal $x(n)$ in a frame of L samples by an L-dimensional row vector x. Let $v, s, \hat{s}$ and e represent the regular-pulse excitation signal, the original speech signal, the synthetic speech signal, and the weighted difference signal, respectively.

The regular-pulse excitation sequence within a frame of size NDRATE is characterized by an offset factor k, $1 \le k \le NS$, and a set of amplitudes $b_k(i)$, $i = 1,2,\ldots,$ NP, which form the entries of an NP-dimensional row vector b_k.

We denote by M_k the NP by L position matrix with entries m_{ij}, where

$$m(i,j) = 1 \quad \text{if } j = (i-1)NS + k \tag{3.1}$$

$$m(i,j) = 0 \quad \text{otherwise}$$

The excitation vector $v^{(k)}$ can be written as

$$v^{(k)} = b_k M_k \tag{3.2}$$

where the superscript denotes the offset factor.

3.3.2 *Basic Algorithm*

Let H and W be L by L matrices whose j-th row contains the impulse response caused by a unit impulse $\delta(t-j)$ of the synthesis filter and error weighting filter, respectively. If $\hat{s}^{(0)}$ is the output of the synthesis filter due to the memory hangover of previous synthesis intervals, we can express the synthetic signal produced by $v^{(k)}$ as:

$$\hat{s}^{(k)} = \hat{s}^{(0)} - v^{(k)}H \quad , 1 \le k \le NP \tag{3.3}$$

The error signal $e^{(k)}$, that is,

$$e^{(k)} = e^{(0)} - v^{(k)}HW \tag{3.4}$$

where

$$e^{(0)} = (s - \hat{s}^{(0)})W \tag{3.5}$$

can be written as

$$e^{(k)} = e^{(0)} - b_k G_k \tag{3.6}$$

where G_k is the NP by L matrix defined as follows:

$$G_k = M_k(HW) \tag{3.7}$$

Our objective is to minimize the squared error:

$$E^{(k)} = e^{(k)} e^{(k)*} \tag{3.8}$$

where * denotes transpose. $E^{(k)}$ is a function of both the pulse amplitudes $b_k(i)$ and the offset factor k. For a given value of k the optimal amplitudes can be computed from Eqs. (3.6) and (3.8), by setting the derivatives of $E^{(k)}$ with respect to the unknown amplitudes to zero. The amplitudes can now be computed by solving b_k from

$$b_k = e^{(0)}G_k^*[G_kG_k^*]^{-1}. \tag{3.9}$$

By substituting Eq.(3.9) in Eq.(3.6) and thereafter the resulting expression in Eq.(3.8), we obtain the following expression for the error:

$$E^{(k)} = e^{(0)}[I - G_k^*[G_kG_k^*]^{-1}G_k]e^{(0)*}. \tag{3.10}$$

The procedure is to compute $E^{(k)}$ for k=1,..,NS, and select that excitation sequence which produces a minimum error. This error is minimal under the given constraints. The optimal excitation vector is entirely characterized by the offset factor k and the corresponding amplitude vector b_k. The whole procedure comprises the solution of NS sets of simultaneous equations as given by Eq.(3.9). These equations can be solved in a computationally efficient way, as will be shown in Chapter 8.

In this section we have seen that the optimal excitation sequence can be *found* by a least mean squares procedure. In the following section we give an alternative look at this minimization procedure.

3.4 Another Approach To The RPE Coder

We note that the *regular-pulse excitation sequence* bears some resemblance to the Base Band Coder (BBC, Section 1.7.1). In this section we look at the RPE coder as an optimal version of the Base Band Coder. The blocks drawn with solid lines in the diagram of Fig. 3.5 represent the conceptual structure of a BBC coder.

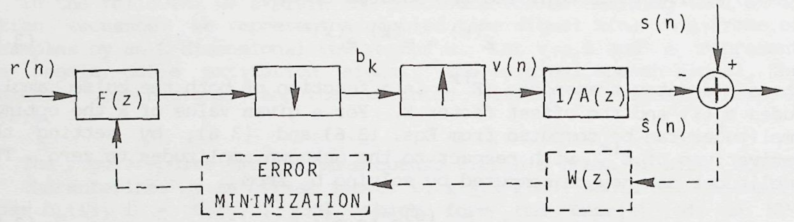


Figure 3.5. Block diagram of a BBC coder (solid lines), and an RPE coder (solid + dashed lines).

The residual signal, obtained by filtering the speech signal through the analysis filter A(z), is band limited by the low-pass filter F(z) and down-sampled. At the receiver an approximating residual signal, obtained by up-sampling, is feed through the synthesis filter to produce the synthetic signal ŝ(n). When the dashed blocks are included in Fig. 3.5 we obtain the RPE encoding scheme. For each value of k (k=1,NS), the filter parameters of the filter $F_k(z)$ are determined to minimize the weighted mean squared error $\Sigma e^2(n)$ over the minimization interval NSSIZE. The subscript k denotes that the filter coefficients depend on the value of k. The value of k and the corresponding filter coefficients of $F_k(z)$ that minimize this error are used to generate the RPE excitation sequence v(n). In the following we derive the procedure for finding the coefficients of the filter $F_k(z)$:

$$F_k(z) = \sum_{i=1}^{L} f_i^k z^{(-i+1)} \tag{3.11}$$

where

$$f^{(k)} = (f_1^k \; f_2^k \quad f_L^k) \tag{3.12}$$

is the finite impulse response of the filter $F_k(z)$. Without loss of generality we shall assume that W(z)=1. We recall that the initial error $e^{(0)}$ is given by

$$e^{(0)} = (s - \hat{s}^{(0)}) \tag{3.13}$$

where $\hat{s}^{(0)}$ is the output of the synthesis filter due to the memory hang-over of previous synthesis intervals. Let $r_0(i)$ (i=1,L) denote the set of residual samples of the current frame and $r\_(i)$, (i=1,L) those of the previous frame. Then we can write

$$e^{(k)} = e^{(0)} - f^{(k)} \begin{bmatrix} r_0(1) & r_0(2) & . & r_0(L) \\ r\_(L) & r_0(1) & . & r_0(L-1) \\ r\_(L-1) & r\_(L) & . & r_0(L-2) \\ . & . & . & . \\ r\_(2) & r\_(3) & . & r_0(1) \end{bmatrix} H \tag{3.14}$$

where the matrix H was defined in Section 3.3.1. The down-sampled signal b_k (with down-sampling factor N) can be written as

$$b_k = f^{(k)} \begin{bmatrix} r_0(1) & r_0(N+1) & r_0(2N+1) & . & r_0(nN+1) \\ r\_(L) & r_0(N) & r_0(2N) & . & r_0(nN) \\ r\_(L-1) & . & . & . & . \\ . & . & . & . & . \\ r\_(2) & . & . & . & . \end{bmatrix} \tag{3.15}$$

$$= f^{(k)} R_k$$

The excitation vector $v_{(k)}$ can expressed as the product

$$v^{(k)} = f^{(k)} \begin{bmatrix} r_0(1) & 0 & . & 0 & r_0(N+1) & 0 & . & r_0(nN+1) & . \\ r\_(L) & 0 & . & 0 & r_0(N) & 0 & . & r_0(nN) & . \\ . & . & . & . & . & . & . & . & . \\ r\_(2) & . & . & . & . & . & . & . & . \end{bmatrix} \tag{3.16}$$

Hence

$$e^{(k)} = e^{(0)} - f^{(k)} R_k M_k H \tag{3.17}$$

$$= e^{(0)} - f^{(k)} R_k H_k$$

where M_k is the position matrix as defined in Eq.(3.1) and $H_k = M_k H$. Minimizing $e^{(k)} e^{(k)*}$ we obtain the expression

$$f^{(k)} = e^{(0)} (R_k H_k)^* [R_k H_k (R_k H_k)^*]^{-1} \tag{3.18}$$

When solving this equation for $f^{(k)}$, the resulting vector b_k is equal to the pulse amplitude vector b_k obtained via the procedure described in Section 3.3.2.

Proof:

When the index k is dropped, Eq.(3.18) can be written as

$$fR[HH^*]R^* = e^{(0)}H^*R^*$$

Multiplying both sides with R gives

$$fR[HH^*]R^*R = e^{(0)}H^*R^*R$$

and assuming that RR^* is non-singular (which will always be the case for speech signals), we get by multiplying both sides by $(RR^*)^{-1}$

$$fR[HH^*] = e^{(0)}H^*$$

Substituting b = fR in the last equation, we obtain Eq.(3.9).

End of proof.

Fig. 3.6 shows some examples of the filter spectra obtained with real speech data.

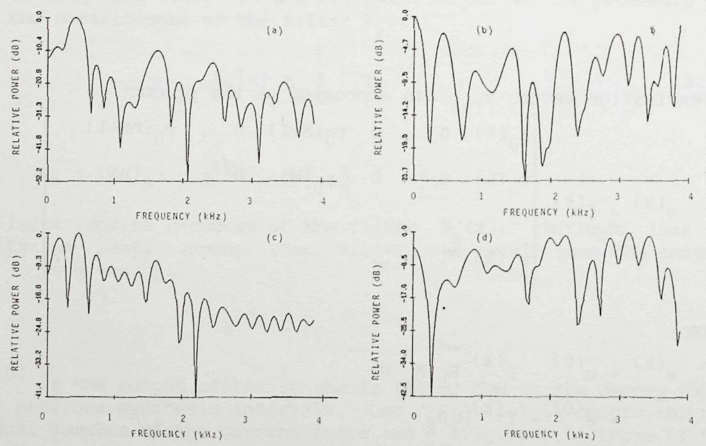


Figure 3.6. Power spectra of a 5 ms speech segment (a,c), and the power spectra of the corresponding filter $F_k(z)$ (b,d).

From this figure we see that the filters $F_k(z)$ are rather different from those used in the Base Band Coder.

3.5 Evaluation Of The RPE Algorithm

The procedures and test sentences used for evaluating the RPE coder have been described in Chapter 2. We used a set of default parameter values, while the parameter under investigation was varied. In Table 3.1 the parameters and their default values are listed.

spacing between pulses	(NS)	4
disturbance rate	(NDRATE)	NSSIZE
search frame size	(NSSIZE)	5 ms
minimization frame size	(NMSIZE)	5 ms
weight factor γ		0.80

TABLE 3.1. Default parameters RPE analysis.

3.5.1 *RPE Analysis Parameters*

The RPE analysis parameters that could affect the final speech quality are listed below.

1. predictor parameters,
2. NS and NDRATE,
3. search interval size (NSSIZE) and minimization interval size (NMSIZE),
4. error weighting filter.

The effects of the predictor parameters were examined in Chapter 2 and we will consider only the parameters used for determination of the excitation sequence.

3.5.1.1 *NDRATE And NS* We observed earlier that the structure of the excitation signal for the case in which there is no disturbance in the distance between the pulses (NDRATE=∞) looks like the down-sampled residual signals used in BBC coders. The main difference is that in the RPE coder no low-pass filter is used. However, this observation can give us a rough estimate of the minimum spacing (NS) between the pulses, required for a good synthetic speech quality. Assuming a maximum fundamental frequency of 500 Hz, we have to use a sampling rate of minimally 1000 Hz. For an 8 kHz sampling rate, the RPE coder should use a pulse spacing less than or equal to 8.

Increasing NDRATE will clearly improve the final speech quality, at the expense of a higher bit rate. To investigate the effect of different disturbance rates and pulse spacings, we did a set of experiments and computed the SNR values of the synthetic speech signals. Fig. 3.7 shows the averaged segmental SNR values for two female and two male speakers for different values of NS and NDRATE. As far as possible, we have chosen the same frame sizes for different values of NS. Also shown in this figure are the averaged SNR values obtained with the MPE coder, which will be discussed in Chapter 4, for different amounts of pulses per 10 ms.

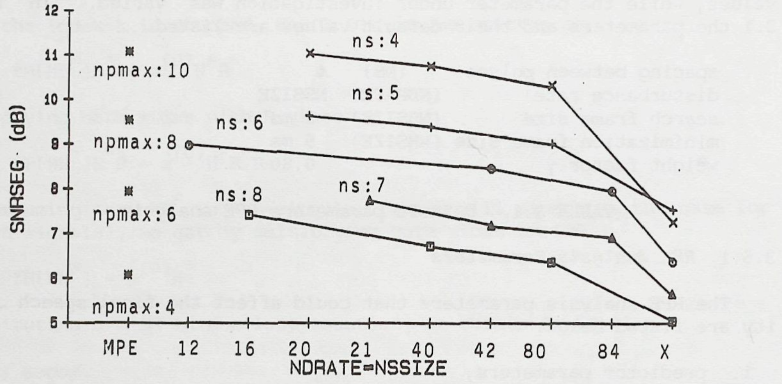


Figure 3.7. Segmental SNR values for different disturbance rates and pitch spacings.

From this figure we can see that for values of NS =4, the RPE coder is better than the MPE coder with 8 pulses / 10 ms. Informal listening tests confirmed this observation. Increasing the disturbance rate, always increases the SNR, but we see that there is no real trade-off between the values of NDRATE and NS. Decreasing NDRATE always introduced a greater improvement than increasing NDRATE. For values of NS greater than 5, some of the utterances (especially those by female speakers) sounded distorted. A similar distortion was observed for MPE coded speech with less than 5 pulses per 10 ms. From this experiment, we conclude that the RPE coder with NS=4 and NDRATE=5 ms can yield results similar to those obtained with the MPE coder operating at the same bit rate (NP=8).

3.5.1.2 *Search Frame Size And Minimization Frame Size* The pulses are to be located in a frame of size NSSIZE while minimizing the error in an interval of size NMSIZE (Fig. 2.2). For the multi-pulse coder we concluded that a minimization frame greater than the search frame introduced a global deterioration in SNR (see Chapter 4). Although a similar effect can be expected for the RPE coder, we did some experiments for different ratios of NSSIZE/NMSIZE. Fig. 3.8 shows the measured SNR values. Again we see that no global improvement can be expected for minimization frame sizes greater than the search frame size.

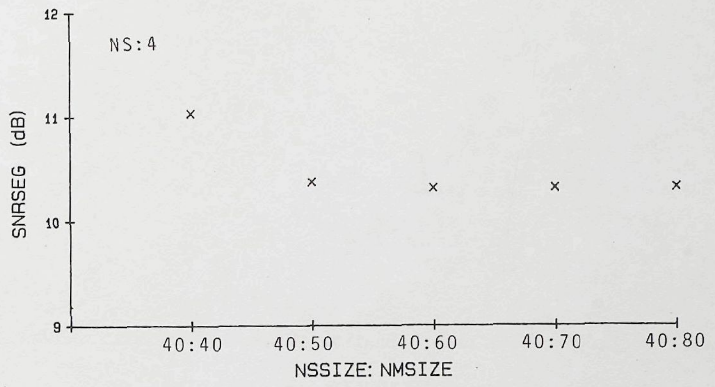


Figure 3.8. Segmental SNR values for different ratios of NSSIZE and NMSIZE.

The effect of the size of the search frame is hard to determine. When search frame size and minimization frame size are identical, we need from a perceptual viewpoint frames greater than or equal to 5 ms. Usually we take NSSIZE equal to NDRATE. To obtain a certain quality level NDRATE is usually bounded by some maximum value. The value of NSSIZE need not be equal to NDRATE, but NSSIZE/NDRATE must be an integer ratio and NSSIZE $\geq$ NDRATE. Within a search frame the possible number of excitation sequences is given by

$$C = NS^{j} \qquad ,j = NSSIZE/NDRATE \tag{3.19}$$

Hence, choosing a search frame greater than NDRATE results in a more complex search procedure. Fig. 3.9 shows the possible excitation patterns for NSSIZE=24, NDRATE=12 and NS=3.

```
        <-----------NSSIZE----------->
        <--NDRATE--><--NDRATE-->
   1:   1..1..1..1..1..1..1..1..
   2:   .1..1..1..1.1..1..1..1..
   3:   ..1..1..1..11..1..1..1..
   4:   1..1..1..1...1..1..1..1.
   5:   .1..1..1..1..1..1..1..1.
   6:   ..1..1..1..1.1..1..1..1.
   7:   1..1..1..1..1.1..1..1..1
   8:   .1..1..1..1...1..1..1..1
   9:   ..1..1..1..1..1..1..1..1
```

Figure 3.9. RPE excitation pattern with NDRATE=18, NSSIZE=36, and NS=3.

Fig. 3.10 shows the SNR values for different search frame sizes and a fixed disturbance rate.

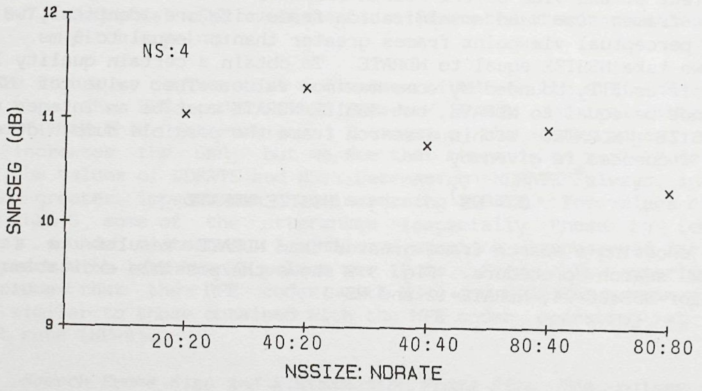


Figure 3.10. Segmental SNR values for different search frame sizes NSSIZE.

From this figure we see a small improvement in SNR for larger search frames, at the expense of a much higher complexity.

3.5.2 *Application Of A Pitch Predictor*

An examination of the regular-pulse excitation (see for example Fig. 3.2) reveals the periodic structure of the excitation for voiced sounds. Obviously, the RPE algorithm aligns the excitation "grid" to the major pitch pulses, thereby introducing the possibility that the remaining pulses within the grid are not optimally located. If we model the major pitch pulses with a pitch predictor, the remaining excitation sequence can be modeled by the regular pulse excitation sequence. The analysis procedure with a pitch predictor is as follows. The pitch synthesizer $1/P(z)$ generates an output signal as a function of past excitation samples and the pitch predictor parameters $\{\beta_i\}$ and M. This excitation signal together with the memory contents of the synthesis filter $1/A(z)$ produces a synthetic signal $\hat{s}^{(0)}(n)$ that serves as a first approximation to the reference signal $s(n)$. Both signals are compared, and the resulting weighted difference signal is used to determine the regular-pulse excitation sequence. The procedure for finding this sequence is similar to the procedure described in Section 3.3, except that the pulses are fed through both the pitch synthesis and the LPC-synthesis filter. The pitch predictor and its parameters were described in detail in Chapter 2 and need no further explanation. The pitch predictor parameters were determined from the residual signal using the covariance method, the range of M was between 16 and 80, and the update rate was equal to that of the synthesis filter coefficients. Fig. 3.11 shows the SNR values of synthetic speech signals produced by the RPE coder with a pitch predictor for different values of the pulse spacings and for different update rates. The corresponding values for the MPE coder with and without a pitch predictor are given as reference.

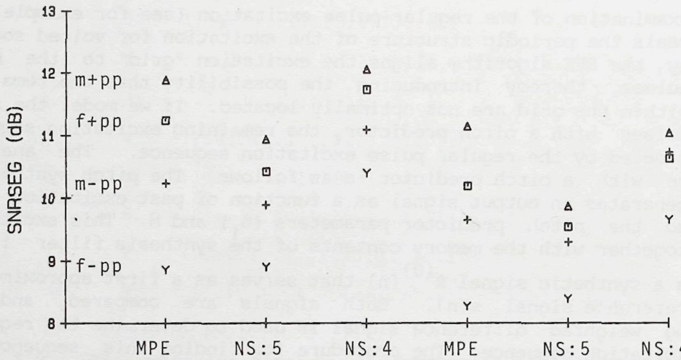


Figure 3.11. Segmental SNR values of the RPE and MPE coder with (+pp) and without (-pp) pitch prediction for different update rates of the predictors (NPRATE=NRATE) and different pulse spacings NS (f=female, m=male).

From this figure we see that the use of pitch prediction improves the performance in terms of SNR. However, this improvement was less perceptible in listening tests.

3.5.3 *Error Weighting Filter*

Spectral noise shaping is not a completely understood procedure. Although the effect of such a procedure can be heard, the real mechanism behind this effect is not clear. The question is whether the proposed noise-shaping filter [Eq. 2.17)] is an effective choice or not. We will skip this question and will concentrate on the effect of the suggested filter and its control parameter γ. This weight factor determines the degree of de-emphasizing the noise power in the formant regions of the speech spectrum. Besides the value of γ, the order of the noise-shaping filter could also be of importance. By default the order is equal to the order of the predictor filter, but a lower order is also possible. While reducing the order we nevertheless must take care that the formant structure of the filter response is preserved, otherwise the effect of noise shaping is lost. Noise shaping reduces the SNR, but improves the perceived speech quality. The effect of noise shaping is small, but it can be heard. An optimal value for γ was found to be between 0.80 and 0.90 at a 8 kHz sampling rate, which corresponds to a 2 dB decrease in SNR. Figs. 3.12 and 3.13 show the power spectra of the speech signal s(n) (solid line) and that of the difference signal s(n)-s(n) (dashed line) for different values of γ.

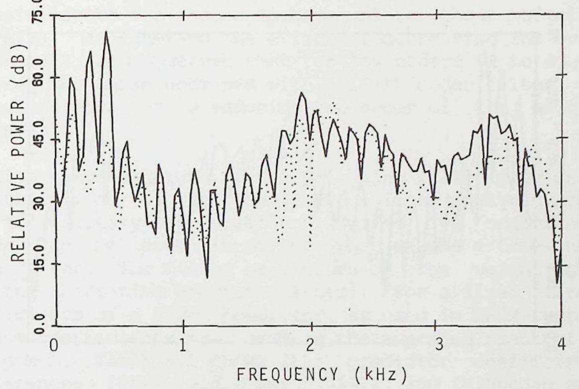


Figure 3.12. Power spectra of the speech signal (solid line) and the difference signal $s(n)-\hat{s}(n)$ (dashed line) for γ=1.0. The spectra were obtained from a 32 ms segment using a Hamming window.

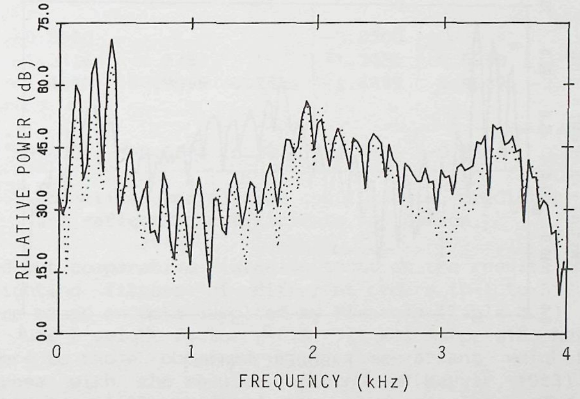


Figure 3.13. Power spectra of the speech signal (solid line) and the difference signal $s(n)-\hat{s}(n)$ (dashed line) for γ=0.80.

From this figure that the error increase in the formant regions and decreases in the regions between the formants. Figs. 3.14 and 3.15 show the power spectra of the speech signal s(n) (solid line) and that of the difference signal s(n)-s(n) (dashed line) for different values of γ, when pitch prediction is applied.

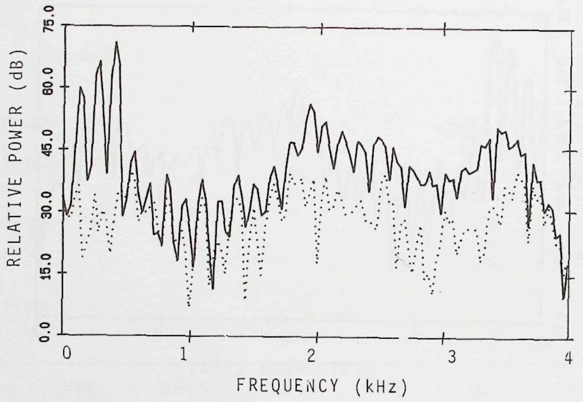


Figure 3.14. Power spectra of the speech signal (solid line) and the difference signal $s(n)-\hat{s}(n)$ (dashed line) for $\gamma=1.0$, and pitch prediction.

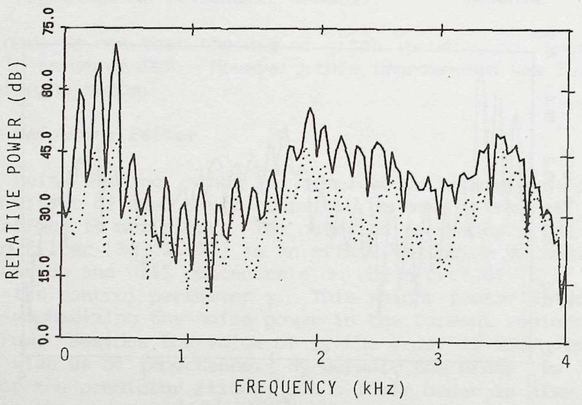


Figure 3.15. Power spectra of the speech signal (solid line) and the difference signal $s(n)-\hat{s}(n)$ (dashed line) for $\gamma=0.80$, and pitch prediction.

From this figures we see that in the case of pitch prediction the error power level decreases, and that the error spectrum is more white.

We can describe the RPE coder as a weighted residual matching procedure (Section 2.6). In that case the error between the regular-pulse signal and the residual signal is weighted with the filter $1/A(z/\gamma)$. This filter is time-variant and the order p and the update rate of the

filter coefficients are chosen to be equal to those of the synthesis filter $1/A(z)$. We examined the effect of decreasing the order p of this filter $1/A(z/\gamma)$, and observed that for low orders (2 to 4), the results were close to those obtained with a 16-th order filter. The computational savings obtained by reducing the order of the weighting filter are marginal.

The time varying nature of the weighting filter provides a significant contribution to the complexity of the analysis procedure. For example, the Cholesky decomposition required for computing the pulse amplitudes can be done in advance when we use a time-invariant noise weighting filter. The filter parameters of the weighting filter are those of the short-time predictor $1-A(z)$. For a fixed filter we can use the coefficients of a fixed predictor, as used in DPCM systems (Chapter 1). These coefficients are based on the averaged spectral characteristics of speech. Table 3.2 shows the predictor coefficients obtained from references [Paez and Glisson, 1972] and [Flanagan et al., 1979]. In this table the normalized correlation terms and the corresponding a-parameters and reflection coefficients are shown.

	Flanagan			Paez/Glisson		
	1	2	3	1	2	3
cor.	0.8500	0.5620	0.3080	0.8456	0.5470	0.2680
p=1	-0.8500	-	-	-0.8500	-	-
p=2	-1.3416	0.5784	-	-1.3435	0.5888	-
p=3	-1.4845	0.9099	-0.2471	-1.4399	0.8089	-0.1638
ref. coef.	0.8500	-0.5784	0.2471	0.8456	-0.5888	0.1638

TABLE 3.2. Normalized correlation coefficients, predictor coefficients and reflection coefficients for speech.

We carried out comparative listening tests on the results obtained with fixed weighting filters of different orders (p=1 to 3). The coefficients were based on data supplied by Flanagan (Table 3.2). Note that we still use a weight factor $\gamma=0.8$. It was surprising how close these results were to those obtained with time-variant weighting filters, which agrees with the results reported in [Nayyar, 1983]. The second order filter especially produced very good results. The first-order filter produced more irregularities than the second- and third-order filters but still sounded reasonable. We also compared synthetic speech obtained with a first-order weight variable weighting filter with speech obtained by fixed weighting filters, and preferred second-order fixed filters over variable first-order filters.

3.6 Complexity Reduction Of The RPE Coder

The analysis procedure of the RPE coder necessitates the solution of NS sets of simultaneous equations, where NS represents the spacing between the pulses. However, the matrices $G_k G_k^t$ which have to be inverted have a displacement rank α = NS+2 [Kailath et al., 1979] and can be solved very efficiently, as we will see in Chapter 8. In this section we will look for modifications of the algorithm to reduce the complexity without affecting the coder performance.

The matrix G_k is obtained by multiplication of the position matrix M_k [Eq.(3.1)] with the impulse response matrices H and W [Eq.(3.7)]. The matrix product HW is given by

$$HW = \begin{bmatrix} g(1) & g(2) & . & g(L) \\ 0 & g(1) & . & g(L-1) \\ 0 & 0 & . & g(L-2) \\ . & & . & . \\ 0 & 0 & 0 & g(1) \end{bmatrix} \qquad (3.20)$$

where L denotes the length of the minimization interval and g(n) are samples of the impulse response of the cascade of the filters 1/A(z) and W(z). The matrix product $G_k G_k^*$ [Eq.(3.9)] depends on the value of k (k=1,2,..,NS). We shall now reconfigure the structure of the solution equations to force the matrix $G_k G_k^*$ to become a single Toeplitz matrix which is independent of the value of the offset factor k. The combined impulse response g(n) of the synthesis filter and the weighting filter decays rapidly for values of γ less than one. By choosing a minimization frame two times the search frame size and setting the impulse response samples g(i) to zero for values of i greater than L=NSSIZE we obtain the following matrix HW:

$$HW = \begin{bmatrix} g(1) & g(2) & . & g(L) & 0 & . & 0 \\ 0 & g(1) & . & g(L-1) & g(L) & . & 0 \\ 0 & 0 & . & g(L-2) & g(L-1) & . & 0 \\ . & . & & . & & . & . \\ 0 & 0 & . & g(1) & . & g(L) & 0 \end{bmatrix} \qquad (3.21)$$

The resulting matrix product $G_k G_k^*$ is now a Toeplitz matrix, independent of the value of k. This result will only hold if the impulse response is constant within the minimization frame. This condition is not satisfied during transitions from one set of filter coefficients to another, as is shown in Fig. 3.16

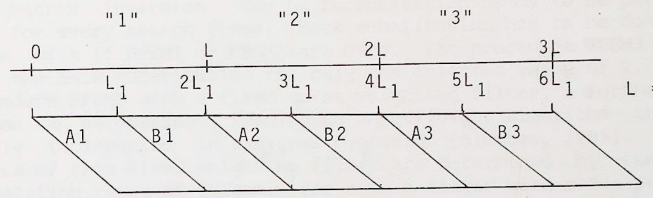


Figure 3.16. Structure of the matrix HW and transition of the filter coefficients.

For the matrices labeled with "A" the impulse response is invariant. For the matrices labeled with "B" the filter coefficients are changed, which means that the impulse response is variant. To force the impulse response to be constant for the matrices "B", a new set of reflection coefficients, which is obtained by linear interpolation between the adjacent sets can be used. For example the interpolated set between set 1 and set 2 can be used for the construction of the matrix B1. For every search frame the $G_k G_k^*$ matrix is now Toeplitz and independent of the value of k. We refer to this method as RPEM1.

Instead of minimizing over the interval of size NMSIZE=2*NSSIZE we can assume that NMSIZE is infinite and that the data is appropriately windowed by a window of size L and is equal to zero outside the window (auto-correlation method). For that case we can still use the HW matrix defined in Eq.(3.20) and we do not need to interpolate between adjacent sets of reflection coefficients. Again we obtain one single Toeplitz matrix $G_k G_k^*$, which is independent of the value of k. This procedure will be referred to as RPEM2

By observing the matrix $G_k G_k^*$ we noted that its structure is mainly diagonally dominant. This structure is due to the fast decaying impulse response g(n) (γ < 1). In finding the optimal excitation within a search frame, we have to select that k that minimizes $E^{(k)}$ (see Eq. 3.10). This is equivalent to searching for the k that maximizes

$$T^{(k)} = \frac{e^{(0)} G_k^* \; G_k e^{(0)*}}{G_k G_k^*} \qquad (3.22)$$

If we replace the matrix product $G_k G_k^*$ by a diagonal matrix, the denominator of Eq.(3.22) becomes constant and we select the k that maximizes

$$e^{(0)} G_k^* \; G_k e^{(0)*} \qquad (3.23)$$

Once the k has been selected, the excitation string b_k is computed according to Eq.(3.10), with for G the matrix defined by Eq.(3.20). This method is referred to as RPEM3. Fig. 3.17 shows the segmental SNR ratios for the different methods.

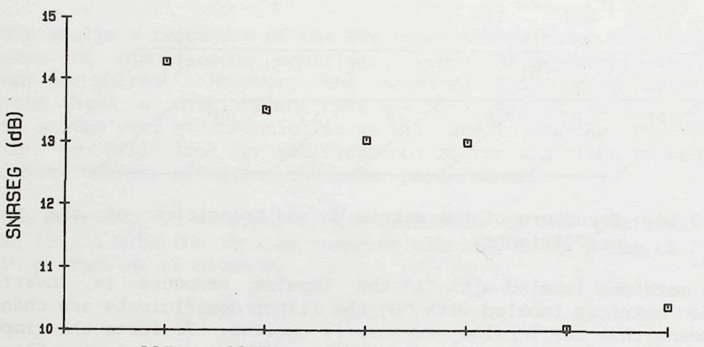


Figure 3.17. Segmental SNR ratios for modified RPE algorithms.

The SNR values were obtained by averaging the SNR values of two utterances by two different speakers. From this figure we see that the modifications introduced resulted in a *slight* decrease in SNR. Informal listening tests revealed that almost no differences between the modified methods and the original RPE method could be heard. Fig. 3.18 shows the segmental SNR values per 10 ms for the modified methods. The solid line represents the SNR of the original RPE and the dashed lines represent the SNR of the modified RPE coders.

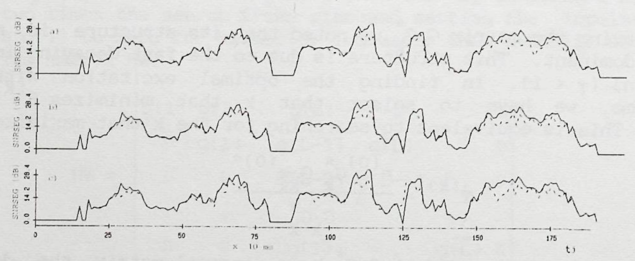


Figure 3.18. Segmental SNR ratios for RPE (solid line) and modified methods (dashed line), (a), RPEM1, (b) RPEM2, (c) RPEM3.

The advantage of the *modified* methods is that we need to perform only

one matrix inversion. The LU factorization needs to be performed only once for every search frame. Back substitution has to be done for every value of k if RPEM1 or RPEM2 are used. The procedure RPEM3 has to perform the back substitution for only the selected value of k. By using procedure RPEM3 with a fixed noise weighting filter, a further simplification can be obtained, leading to an efficient algorithm that can be easily implemented in a signal processor [Sluyter, 1985]. The coefficients of this fixed weighting filter are determined by the long-term *correlation terms of speech. For such a filter with order* p=2 or 3, the matrix $G_k G_k^*$ is diagonal dominant, which is due to the fast decaying impulse response of the weighting filter. According to Eq.(3.9) the amplitude vector b_k of a set of pulses is given by

$$b_k = e^{(0)} G_k^* [G_k G_k^*]^{-1} \qquad (3.24)$$

By replacing the product $G_k G_k^*$ by a diagonal matrix D, whose diagonal elements consist of the first auto-correlation coefficient of the filter impulse response, we get

$$b_k = e^{(0)} G_k^* D^{-1} \qquad (3.25)$$

If we assume the residual signal to be zero outside the minimization frame, or equivalently disregard the filter states, we may write $e^{(0)}$ as

$$e^{(0)} = rG \qquad (3.26)$$

where r is the residual signal. Substituting Eq.(3.26) in Eq.(3.25) we get

$$b_k = rG_k G_k^* D^{-1} = rC \qquad (3.27)$$

where C is a symmetric band matrix with the element c_i, on the i-th sub-diagonal (i.e. main diagonal is i=0). The coefficients $\{c_i\}$ are the normalized auto-correlation coefficients of the impulse response of the weighting filter. From Eq.(3.27) it can be easily seen that we can compute a vector b of size NSSIZE and compute the corresponding b_k by down-sampling this vector. The optimal excitation vector is the one that minimizes $E^{(k)}$ or equivalently maximizes

$$T^{(k)} = e^{(0)} G_k^* G_k e^{(0)} = b_k b_k^* \qquad (3.28)$$

The whole procedure is now extremely simple. The residual signal r is "smoothed" with the smoother represented by the matrix C. The resulting output vector is downsampled and the b_k which maximizes $T^{(k)}$ [Eq.(3.28)] is selected. This procedure is shown in Fig. 3.19.

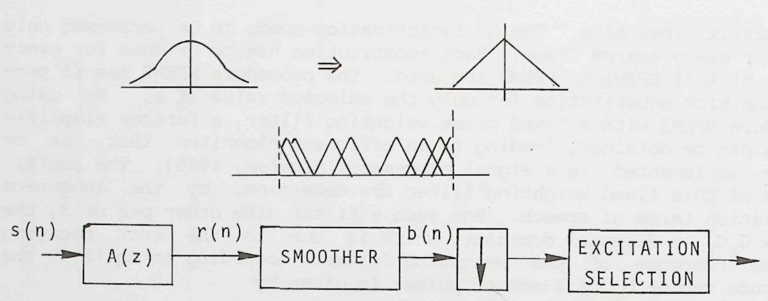


Figure 3.19. Simplified RPE procedure.

For comparison the averaged SNRSEG values are shown in Fig. 3.17. The RPE coder using a fixed weighting filter is referred to as RPF1, while the procedure outlined above is referred to as RPF2. In Fig 3.20 the same comparison is made for the SNRSEG as a function of time for both a male and a female speaker.

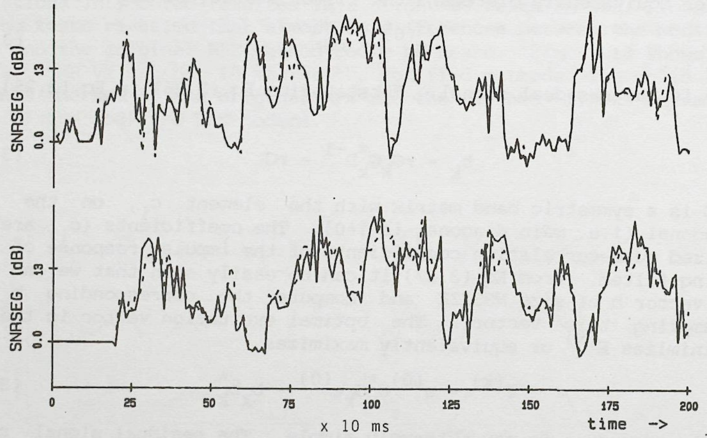


Figure 3.20. Segmental SNR for two RPE coders using a fixed weight filter for a female (a) and a male (b) speaker. RPF1 procedure (solid line), RPF2 procedure (dashed line).

From this figure it is clear that for a fixed weighting filter procedure RPF2 provides a quality comparable to that of procedure RPF1. The advantage of the former is its ease of implementation.

3.7 Summary

This chapter introduced a new coding concept that models the excitation sequence by a regular pulse excitation. The excitation signal is determined in such a way that the perceptual error between the original and the synthetic signal is minimized. The computational effort is only moderate and can be further reduced by using a fixed weighting filter or truncating the impulse response. The coder can produce (near) toll quality speech at bit rates around 9600 b/s by using a pulse spacing equal to 4 and quantizing each pulse with 3 bits. The use of pitch prediction improves the speech quality, but it was concluded that the RPE coder performs adequately without a pitch predictor.

References

Deprettere, Ed. F. and P. Kroon, "Regular excitation reduction for effective and efficient LP-coding of speech," *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing*, pp. 25.8.1-25.8.4 (March 1985).

Flanagan, J.L., M.R. Schroeder, B.S. Atal, R.E. Crochiere, N.S. Jayant, and J.M. Tribolet, "Speech Coding," *IEEE Trans. Communications* Vol. COM-27(4) pp. 710-736 (April 1979).

Kailath, T., S.Y. Kung, and M. Morf, "Displacement ranks of matrices and linear equations," *J. of Mathematical Analysis and applications* Vol. 68(2) pp. 395-407 (April 1979).

Nayyar, G.P., "Multi-pulse excitation source for speech synthesis by linear prediction," Report 439, Institute for Perception Research, Eindhoven, the Netherlands (1983).

Paez, M.D. and T.H. Glisson, "Minimum mean squared error quantization in speech PCM and DPCM systems," *IEEE Trans. Communications* Vol. COM-20 pp. 225-230 (April 1972).

Sluyter, R.J., "Coding the prediction residual of speech at low bit rates," Internal Report, Philips Research Laboratories, Eindhoven, The Netherlands (1985).

4. MULTI-PULSE EXCITATION CODER

4.1 Introduction

This chapter describes another parametric approach to finding an appropriate excitation sequence. The procedure was proposed in [Atal and Remde, 1982] and is referred to as the Multi-Pulse Excitation (MPE) coder. *The excitation sequence consists of a sequence of non-uniformly* spaced pulses with varying amplitude (multi-pulse excitation). The locations and the amplitudes of the pulses are chosen to be optimal according to the selected fidelity criterion.

We shall discuss a number of procedures for determining the multi-pulse excitation signal. Then, we shall discuss the coder performance according to objective and subjective tests. Further, we discuss modifications to the MPE coder to improve the performance of the coder without increasing the bit rate. Finally, we describe some alternatives for the pulse search procedure to decrease the complexity without too much reduction in quality.

4.2 Multi-Pulse Excitation Coding

In the previous chapter we described a parametric approach for finding the optimum excitation sequence for the model filter(s), such that the weighted squared error between the original and the reconstructed signal is minimized. The first successful parametric approach was proposed in [Atal and Remde, 1982], and is called Multi-Pulse Excitation (MPE) coding. The excitation sequence consists of a small number of non-zero pulses, whose amplitudes and positions have to be determined such that the error criterion between the original and the reconstructed speech is minimized. The procedure to finding the excitation uses a sub-optimal iterative approach, which searches one pulse at a time. In Section 4.2 this search procedure is described in detail, but first we give some examples of the results that can be obtained with the MPE method.

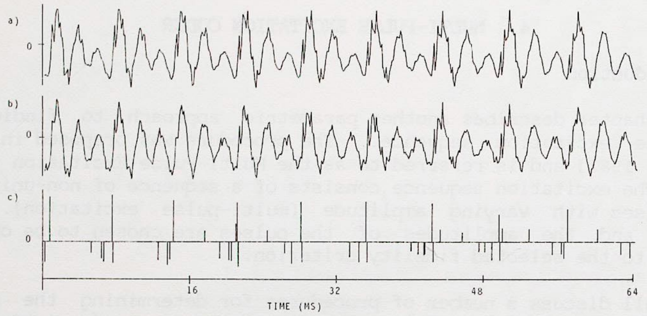


Figure 4.1. Waveforms of the original speech (a), the decoded speech (b), and the excitation signal (c).

Fig. 4.1 shows an example of the original and decoded waveforms, and the corresponding excitation signal. The excitation sequence was determined using the default procedure as listed in Table 4.1. The corresponding power spectra of the original and the reconstructed speech are shown in Fig. 4.2.

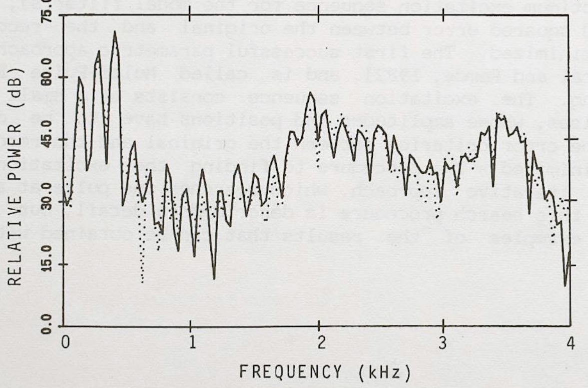


Figure 4.2. Power spectra of a 32 ms segment of the input speech (solid line) and the reconstructed speech (dashed line).

The segmental SNR for the utterance "A lathe is a big tool" spoken by both a male and a female speaker is shown in Fig. 4.3.

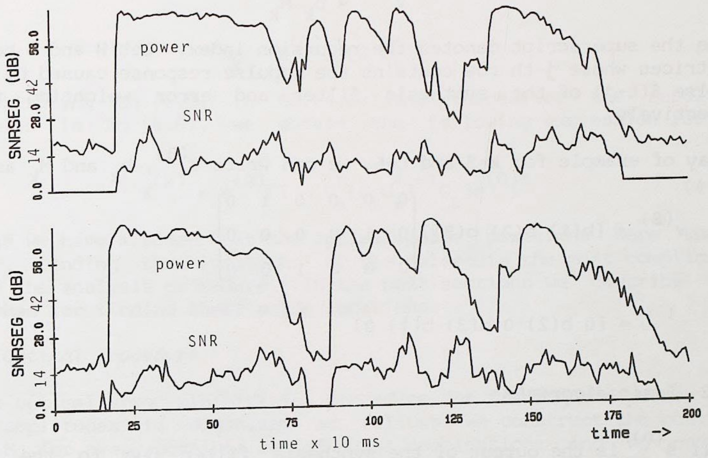


Figure 4.3. Segmental SNR for successive time frames for a female speaker (a), and a male speaker (b). The upper curve represents the signal energy shifted + 15 dB.

4.3 Error-Minimization Procedures

4.3.1 *Definitions*

The excitation search procedure determines the excitation in frames of size NSSIZE, while minimizing the error in frames of size NMSIZE. In general, the maximum number of pulses (NP) per search frame is fixed, so that we obtain a constant pulse rate for relatively short time intervals (10-30 ms).

We represent a signal $x(n)$ in a frame of L samples by an L-dimensional row vector x. Let $v, s, \hat{s}$ and e represent the multi-pulse excitation signal, the original speech signal, the reconstructed speech signal, and the weighted difference signal, respectively.

The multi-pulse excitation sequence within a search frame is characterized by a set of locations $n(i)$ with corresponding amplitudes $b(i)$, $i = 1,2,\ldots,k$ and $1 \le k \le NP$. Denoting by M_k the k by L position matrix with

$$m(i,j) = 1 \quad \text{if } j = n(i) \tag{4.1}$$

$$m(i,j) = 0 \quad \text{otherwise}$$

and considering the set $b(i)$ as entries in an k-dimensional amplitude vector b_k we can write the excitation signal $v^{(k)}$ as

$$v^{(k)} = b_k M_k \tag{4.2}$$

where the superscript denotes the recursion index. Let H and W be L by L matrices whose j-th row contains the impulse response caused by a unit impulse $\delta(t-j)$ of the synthesis filter and error weighting filter, respectively.

By way of example for k=3 and L=6, we can write $v^{(3)}$, b_k and M_k as:

$$v^{(3)} = [b(1)\ b(2)\ b(3)] \begin{bmatrix} 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \end{bmatrix} =$$

$$= [0\ b(2)\ 0\ b(3)\ b(1)\ 0]$$

4.3.2 *Basic Algorithm*

If $\hat{s}^{(0)}$ is the output of the synthesis filter due to the memory hangover of previous synthesis intervals, we can express the synthetic signal produced by k pulses as:

$$\hat{s}^{(k)} = \hat{s}^{(0)} + v^{(k)}H \quad , 1 \le k \le NP \tag{4.3}$$

The error signal $e^{(k)}$, that is,

$$e^{(k)} = e^{(0)} - v^{(k)}HW \tag{4.4}$$

with

$$e^{(0)} = (s - \hat{s}^{(0)})W \tag{4.5}$$

can be written as

$$e^{(k)} = e^{(0)} - b_k G_k \tag{4.6}$$

where G is the k by L matrix defined as follows:

$$G_k = M_k\ (HW) \tag{4.7}$$

Our objective is to minimize the squared error:

$$E^{(k)} = e^{(k)}\ e^{(k)*} \tag{4.8}$$

where * denotes transpose. $E^{(k)}$ is a function of both the pulse amplitudes and the pulse positions. Once the pulse positions have been determined, the optimal amplitudes can be computed from Eqs. (4.6) and (4.8), by setting the derivatives of $E^{(k)}$ with respect to the unknown amplitudes to zero. The amplitudes can now be computed by solving b_k

from

$$b_k = e^{(0)}G_k^*[G_kG_k^*]^{-1} \tag{4.9}$$

By substituting Eq.(4.9) in Eq.(4.6) and thereafter the resulting expression in Eq.(4.8), we obtain the following expression for the error:

$$E^{(k)} = e^{(0)}[I - G_k^*[G_kG_k^*]^{-1}G_k]e^{(0)*} \tag{4.10}$$

Thus far we have assumed that the optimal pulse positions were known. However, finding the locations of the pulses is the most complicated part of the analysis procedure. In the next sections we describe some approaches for finding these pulse locations.

4.3.3 *Optimal Procedure*

The optimal error minimization procedure for finding the pulses and their amplitudes is organized as follows: We construct the position matrix M_k for every possible set of pulse combinations and we compute the matrix G_k and the error $E^{(k)}$ using Eqs. (4.7) and (4.10), respectively. The optimum set of NP pulse locations is the set that corresponds to the minimum of $E^{(NP)}$. Solving Eq.(4.9) for the optimal set of locations will yield the optimal excitation sequence $v^{(NP)}$. This procedure is extremely complex due to the large number of possible sets. The number of possible combinations of k pulses in a segment of L samples is given by

$$C_{op} = \frac{L!}{(L-k)!\ k!} \tag{4.11}$$

Even for small values of L and k, the number of combinations becomes quickly impractical. To avoid this large number of iteration steps, we will use sub-optimal solutions.

4.3.4 *Sub-Optimal Procedures*

Instead of finding all the pulses at once, we can determine the locations and the amplitudes one pulse at a time. The optimum location for any of these pulses is found by computing the error for all possible pulse locations in a given interval and by locating the minimum of the error. Once the location is known, we can compute the corresponding pulse amplitude, update the error and start searching for the next pulse. The number of possible positions in which to search for k pulses in a segment of L samples is now

$$C = \tfrac{1}{2}\ k(2L - k + 1) \tag{4.12}$$

For computing locations and amplitudes one pulse at a time several procedures exist. The simplest approach is to compute at stage k only the location and amplitude of the k-th pulse and assume the locations and amplitudes of the previous k-1 pulses to be known. A major drawback of this approach is the interaction between the pulses, resulting in pulse amplitudes that are jointly not optimal. New pulses are often required to compensate for inaccuracy in earlier stages. Henceforth we refer to this procedure as SUB1. To correct for the interaction between the pulses, we can compute the jointly optimal pulse amplitudes for all k pulses by solving Eq. (4.9) every time a new pulse has been found. For the given set of pulse locations, the resulting squared error $E^{(k)}$ is now minimal. We will refer to this procedure as SUB2. A further refinement of this approach is to include the jointly optimal pulse amplitude computation in the search procedure. For every possible location of the new pulse, we compute the optimal amplitudes of all k pulses and the resulting error $E^{(k)}$. The location that produces the minimum error will be selected. This fairly complex procedure will be referred to as SUB3. Fig. 4.4 illustrates the first 4 steps of the sub-optimal error minimization procedure (SUB1).

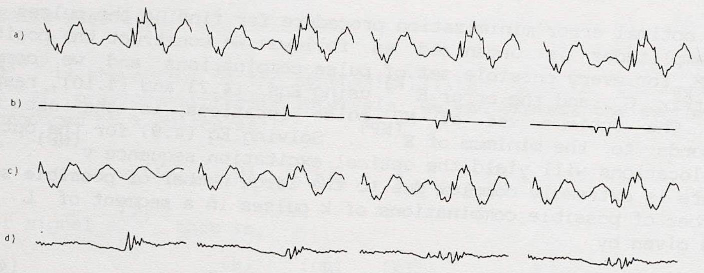


Figure 4.4. Illustration of the first 4 steps in the SUB1 procedure. a) speech segment s(n), b) excitation sequence v(n), c) reconstructed speech segment ŝ(n) and d) weighted error e(n).

These different procedures for finding pulse amplitudes and locations can also be described in terms of projection operators. The initial vector $e^{(0)}$ lies in an L-dimensional space *HW*, spanned by the row vectors of the L by L product matrix HW. The approximation vector $y^{(k)}$ defined by

$$y^{(k)} = v^{(k)}HW = b_k G_k \qquad (4.13)$$

lies in a sub-space of *HW* spanned by the row vectors of the matrix G_k [Eq.(4.7)]. Minimizing the mean-squared error $E^{(k)}$ between $e^{(0)}$ and $y^{(k)}$ is equivalent to performing an orthogonal projection of the vector $e^{(0)}$ on this subspace. This can be easily verified by substituting Eq.(4.9) in Eq.(4.13) to obtain

$$y^{(k)} = e^{(0)}P \quad , \text{with} \quad P = G_k^*[G_k G_k^*]^{-1}G_k \qquad (4.14)$$

where P is called the projection operator. This projection is shown in

Fig. 4.5.

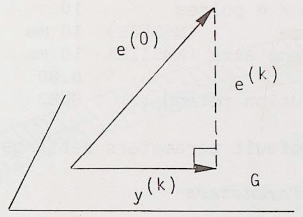


Figure 4.5. Orthogonal projection of $e^{(0)}$ on G_k.

Procedure SUB1 can be represented as an orthogonal projection of the error $e^{(k-1)}$ on one of the basis vectors of *HW*, and selecting the error vector $e^{(k)}$ that has a minimum norm. The final error vector $e^{(NP)}$ is built up recursively and need not be orthogonal to $y^{(NP)}$. For SUB2, the procedure for finding the pulses is similar to that of SUB1, except that after every step we perform an orthogonal projection of $e^{(0)}$ on the space containing the partial approximation vector $y^{(k)}$. The search for the next pulse location is not an orthogonal procedure. That is, the next pulse location is one that minimizes the error vector obtained by projection on one of the basis vectors of *HW*. Since these basis vectors are not orthogonal to the previous solution space, the new error vector will not be orthogonal to the previous solution space. The projection of the $e^{(0)}$ vector on the new solution space will of course correct for this non-optimal decomposition; nevertheless, the last pulse is not optimal. This problem is solved in the SUB3 procedure. In this procedure we project $e^{(0)}$ on every possible subspace obtained by adding one of the basis vectors of *HW* to the previous partial subspace and select the error vector that has a minimum norm. Every new pulse is now selected in the most optimal way. Procedure SUB3 can also be described in a form similar to procedure SUB1 where the error is built up recursively. To make sure that every pulse provides an optimal contribution to the minimization of the error, we orthogonalize the basis vectors of *HW* to the space spanned by the partial approximation vector $y^{(k)}$. The new pulse can now be found by performing an orthogonal projection on one of the orthogonalized basis vectors of *HW*. Note that for this procedure it is not necessary to recompute amplitudes of previously found pulses. After finding all pulse locations the amplitudes of the pulses relative to the original set of basis vectors are found by projecting $e^{(0)}$ on $y^{(k)}$.

4.4 Evaluation Of The MPE Algorithm

The remainder of this section is mainly intended to provide a better understanding of the parameters involved in the multi-pulse analysis procedure, and the way these parameters are affecting the quality of the reconstructed speech signals. In Table 4.1, the MPE parameters and their default values are listed.

ratio # samples / # pulses	10
search frame size (NSSIZE)	10 ms
minimization frame size (NMSIZE)	10 ms
weight factor γ	0.80
sub-optimal solution method	SUB2

TABLE 4.1. Default parameters settings for MPE analysis.

4.4.1 *MPE Analysis Parameters*

The analysis procedure of the MPE coder involves many parameters that could affect the final speech quality. We consider the following parameters:

1. predictor parameters,
2. number of pulses per time interval,
3. search interval size (NSSIZE) and minimization interval size (NMSIZE),
4. error weighting filter,
5. solution method for finding pulse locations and amplitudes.

The effects of the predictor parameters were discussed in Chapter 2 and will not be discussed in this chapter.

4.4.1.1 *Number Of Pulses Per Time Interval* The required minimum number of pulses per interval can be estimated as follows. The fundamental frequency of speech ranges from 50 Hz up to 500 Hz, corresponding to pitch periods between 20 and 2 ms, respectively. Assuming that we need at least one pulse in every pitch period, we have to provide at least 4 or 5 pulses for every 10 ms. Although such an estimation can only be made for voiced speech segments, it is expected that for an adequate representation of unvoiced speech segments more than 4 or 5 pulses for every 10 ms are needed.

As can be seen in Fig. 4.6, the *number of pulses per time interval* has the greatest impact on the quality of the reconstructed speech.

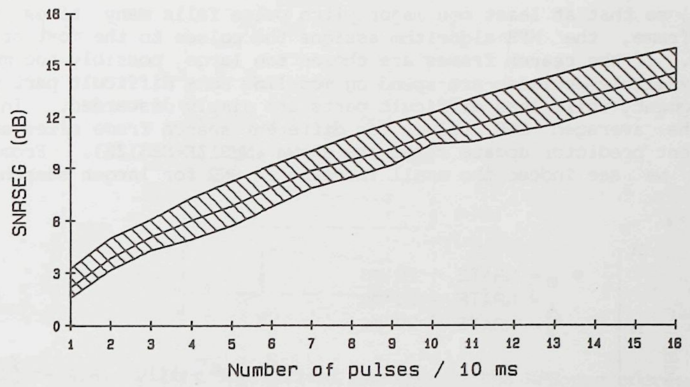


Figure 4.6. Segmental SNR values for different numbers of pulses/time interval.

In this figure the upper and lower limits represent maxima and minima, respectively over four speakers (two male and two female), and the middle curve gives mean values (averages over four speakers). Adding more pulses always increases the SNRSEG although the increase is smaller for high numbers. The quality of the reconstructed speech using 8 pulses per 10 ms is close to toll quality for male speakers, but some female speakers sound a little bit rough and the pulse rate has to be increased to 12 or 16 pulses to get rid of this effect. Beyond 16 pulses per 10 ms the SNRSEG will still increase, but the sound quality will remain constant. The reconstructed signal is then undistinguishable from the original signal.

4.4.1.2 *Search Frame Size And Minimization Frame Size* The pulses are to be located in a frame of size NSSIZE while minimizing the error in an interval of size NMSIZE (see Chapter 2, Fig. 2.2). The choice of the optimal search frame size depends on both qualitative and computational aspects. The computational complexity (in terms of multiply/add operations) is both a function of the search interval size (NSSIZE) and the number of pulses (NP), and increases tremendously for increasing search frame sizes. Considering qualitative aspects, it can be argued that the search frame size may not be chosen too small (less than 5 ms), to avoid non-optimal localized pulses. The following reasoning shows why small search frames (less than 5 ms) are not optimal. Suppose we want to model low pitched voiced speech. From examining multi-pulse excitations we know that many pulses are located around the major pitch pulse. If the average pitch period is greater than the search frame size used, probably many search frames will contain no major pitch pulse at all. If the search frame size is much smaller than the pitch period, we try to model a less important part of the pitch period with, say, 4 pulses, while 2 pulses would be enough. If the search frame sizes are large enough, so that at least one major pitch pulse falls many times within this frame, the MPE algorithm assigns the pulses to the most critical places. If the search frames are chosen too large, possibly too much of the available pulses are spend on modeling some difficult part within the segment, while less difficult parts are simply discarded. In Fig. 4.7 the averaged SNR values for different search frame sizes and for different predictor update rates are shown (NMSIZE=NSSIZE). From this figure we see indeed the small increase in SNR for larger search frame sizes.

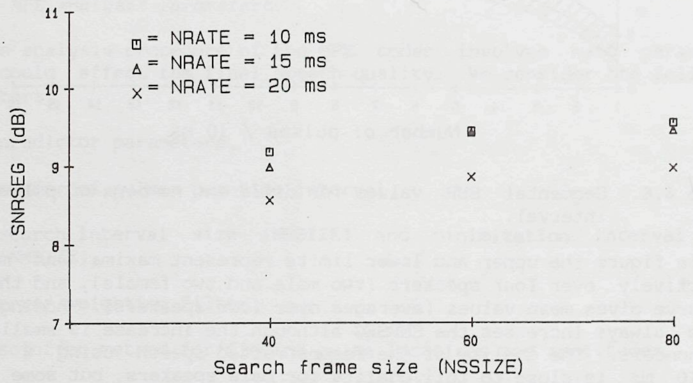


Figure 4.7. Segmental SNR values for different search frame sizes.

Depending on its position and its impulse response length, an allocated pulse will have an effect in one or more succeeding search frames. Moreover, for equal search frame sizes and minimization frame sizes, the contribution of pulses located at the end of the search frame is small, resulting in a bias of the pulses located at the beginning of the frame. To incorporate both effects, we could choose NMSIZE greater than NSSIZE. The ratio NMSIZE/NSSIZE depends on the length of the impulse response of the corresponding prediction filter and can be made adaptive.

Although the predictor frame rates (NRATE) and search frame sizes (NSSIZE) can be chosen independently, the complexity will be less if NRATE and NSIZE are chosen as integral ratios and the frame boundaries are aligned. In this way we avoid problems that could arise if the filter coefficients change in the search interval. However, under the condition that we use a time varying weighting filter and NMSIZE is greater than NSSIZE, we always have to deal with coefficient changes, as is illustrated in Fig. 4.8.

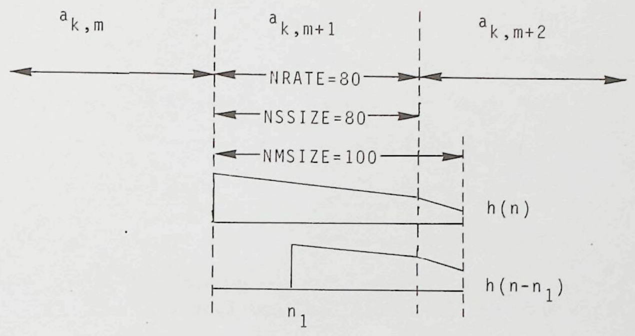


Figure 4.8. Filter coefficient changes in the MPE procedure.

In Fig. 4.9 the averaged SNR values for different ratios of NSSIZE and NMSIZE are shown. We see from this figure that the averaged SNR decreases if NMSIZE is greater than NSSIZE.

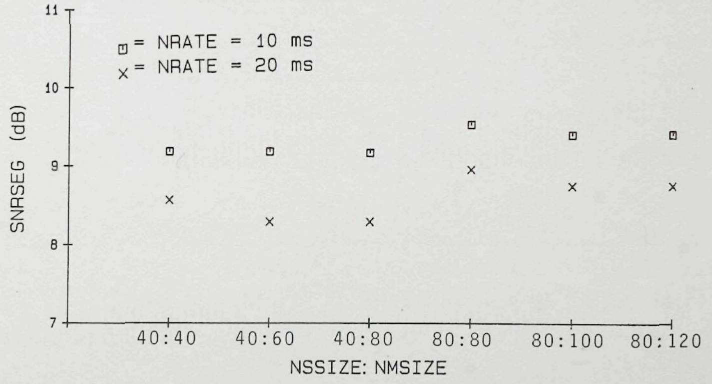


Figure 4.9. Segmental SNR values for different ratios of NSSIZE and NMSIZE.

Sometimes we find an improvement locally as is shown in Fig. 4.10, which gives the SNR as a function of time for a short segment of an utterance by a female speaker.

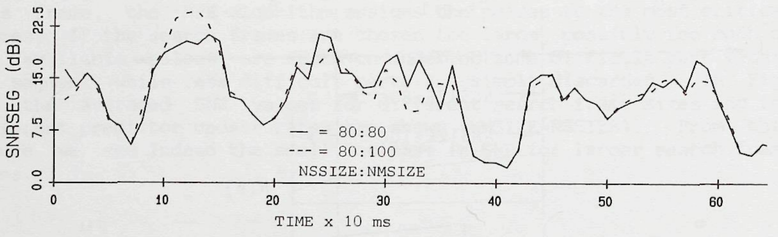


Figure 4.10. Differences in segmental SNR for NMSIZE = NSSIZE (solid line) and NMSIZE > NSSIZE (dashed line).

To find an explanation for this effect, we computed the distribution of the allocated pulses within the search frame. Fig. 4.11 shows the histograms, obtained from the multi-pulse analysis of one single female speaker, for NMSIZE=NSSIZE (upper histogram) and NMSIZE > NSSIZE (lower histogram). For NMSIZE=NSSIZE, we see a small bias for locations at the beginning of the search frame. If we choose NMSIZE greater than NSSIZE, we get a much larger bias in the opposite direction.

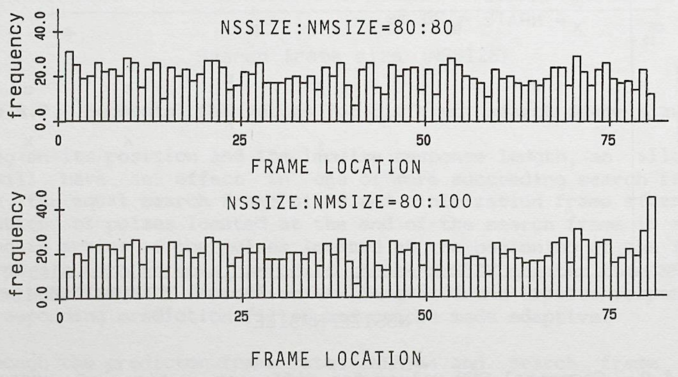


Figure 4.11. Distribution of pulses within a search frame for different ratios of NSSIZE:NMSIZE.

An explanation for this effect is that large errors outside the search interval are most effectively minimized by locating pulses at the end of the search frame. Any pulse located at the end of the search frame has only a small contribution to the error minimization within this frame. As a result we have a non-optimally placed pulse for both the search frame area and for the part of the minimization frame that lies outside the search frame. The averaged SNR values indicate that this effect is,

viewed globally, a disadvantage.

4.4.1.3 *Error Weighting Filter* As noted before, noise shaping reduces the SNR, but improves the perceived speech quality. In Fig. 4.12 the averaged SNR values for different values of γ are shown. Given as well is a rating of quality (relative scale) as determined by listening tests.

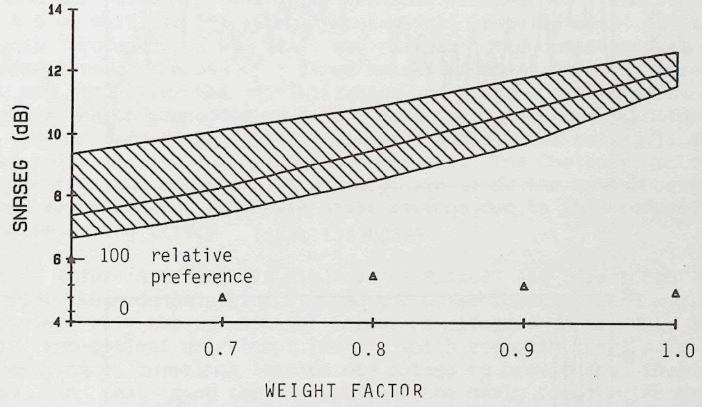


Figure 4.12. Averaged segmental SNR values and subjective relative preference for different values of γ.

The effect of noise shaping is small, but it can be heard. An optimal value for γ was found to be between 0.80 and 0.90 at a 8 kHz sampling rate. From this figure we see that 0.80 corresponds to a 2 dB decrease in SNR. In Fig. 4.13 we show the spectra of the original speech segment, and the corresponding error spectra for different values of γ.

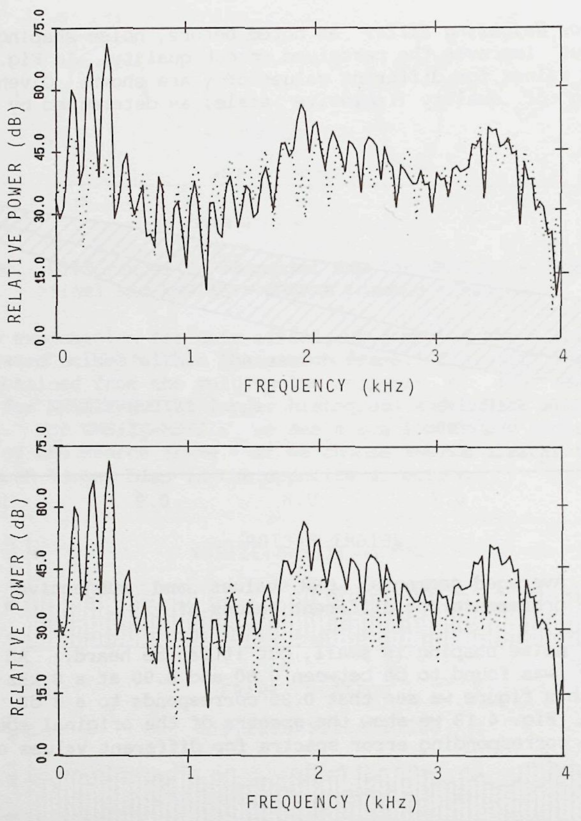


Figure 4.13. Power spectra of a 32 ms segment of the input speech, and of the error signal $s-\hat{s}$ (dashed line), for different values of γ. a) $\gamma=1.0$, b) $\gamma=0.80$.

From this figure the effect of error weighting is clearly visible. For $\gamma = 1.0$, the coding error energy is larger that the signal error energy, especially in the 1 kHz region. The application of error weighting increases the error in the formant regions, but decreases the error energy around 1 kHz.

We examined just as with the RPE coder the effect of decreasing the order p of the weighting filter $1/A(z/\gamma)$, and observed that for low orders (2 to 4), the results were close to those obtained with a 16-th order filter. The computational savings obtained by reducing the order

of the weighting filter are marginal. We carried out comparative listening tests on the results obtained with different order fixed weighting filters (p=1 to 3). The coefficients were based on data supplied by Flanagan (Table 3.2). Note that we still use a weight factor $\gamma=0.80$. *Similar as with the RPE coder it was surprising how close these* results were to those obtained with time-variant weighting filters. The second order filter especially produced very good results. We preferred second-order fixed filters over variable first-order filters. In Section 4.6 we will examine the computational requirements of the MPE analysis procedure. We will now discuss the computational savings obtained through the use of a fixed noise weighting filter. These savings are small for the initialization and state update procedures. In the pulse search procedure we do not need to compute the denominator of the second term of Eq.(4.10). However, the biggest saving is obtained in the pulse amplitude computation procedure. The Cholesky factors can be computed in advance and we only have to do the back-substitution. For the error update, we obtain small savings due to the reduced order of the weighting filter.

4.4.1.4 *Solution Method For Finding The Pulses* As described before, different methods exist for finding the pulse locations and amplitudes. We consider only the sub-optimal solutions referred to as SUB1, SUB2 and SUB3. Sub-optimal solution 1 (SUB1), which computes every pulse amplitude only once, sometimes locates new pulses in previously found locations. In that case the amplitude of the newly found pulse has to be added to that of the already found pulse. This phenomenon usually occurs in voiced segments if more than 6 pulses per 10 ms have been allocated. For unvoiced segments many more pulses must have been allocated before this effect (pulse doubling) occurs. To avoid pulse doubling we can exclude already allocated locations from future selections (SUB1X). If the amplitudes are jointly optimized every time a new pulse has been found, the solution for the given set of locations is optimal, and the next pulse will not be located on one of the previous found locations. Mixed form combinations such as performing SUB1 or SUB1X analysis are also possible, and once all NP pulses have been found, the amplitudes of all pulses are recomputed. These procedures are referred to as SUB1R and SUB1XR. Fig. 4.14 lists different possibilities and the corresponding SNR values.

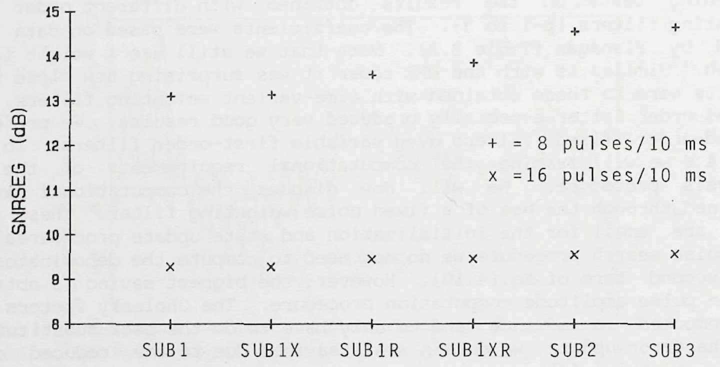


Figure 4.14. Segmental SNR values for different solution methods.

From Fig. 4.14 we see that the differences among the different methods are small for a low number of pulses, but become more significant for an increasing number of pulses. For both cases SUB3, which recomputes the amplitudes of all pulses during the search procedure, yields the best results. A good alternative to using SUB3 is to use SUB2, or SUB1, to inhibit pulse doubling and to make a final amplitude recomputation of all pulses found.

4.5 Improvement Of The MPE Coder

The MPE coder as described in the previous section can produce good quality speech at medium bit rates. However, for parameter settings suitable for 10 kb/s rates, degradations in the reconstructed speech signal can still be heard. Especially high-pitched (e.g. female speakers) sounds suffer from a perceivable deterioration. In this section we describe some of the possible ways to improve the performance of the MPE coder, without affecting the bit rate.

4.5.1 *Variable Pulse Rate*

In Section 4.4.1.1 we demonstrated that a pulse rate of 8 pulses per 10 ms is sufficient for most speakers. Female speakers sometimes need a higher rate, which can be explained by the higher fundamental frequency of female speakers. Male speakers usually sound very good, except for small segments containing transitions or pitch inclinations. A solution that reduces these effects is to use a varying number of pulses. The error criterion already used in the MPE analysis procedure can also be used to determine the appropriate number of pulses within every search frame. Obviously this approach is most suitable for non real-time-applications such as minimizing memory requirements for speech storage. For communication applications, the resulting variable bit rate is

usually an undesired feature. For these applications we can impose the constraint that the number of pulses per specified time interval must be fixed. The size of this interval is chosen to be equal to the maximum allowable delay time. For example, the maximum delay time is 30 ms, and MPE analysis is done on 10 ms frames. The available number of pulses can now be optimally distributed over the three 10 ms frames, thereby avoiding the complexity and possible non-optimality of doing an MPE analysis on 30 ms frames at once. The procedure for allocating the number of pulses within a search frame is organized as follows. To avoid energy gaps due to missing pulses, a minimum number of pulses NPMIN has to be placed within every search frame. To determine the required number of pulses we used the ratio

$$E_1^{(k)} = \sum_n s^2(n) \;/\; \sum_n [e^{(k)}(n)]^2 \tag{4.15}$$

as a measure. In this formula $s(n)$ represents the input signal, $e^{(k)}(n)$ the weighted error, and the superscript k the number of pulses. New pulses will be added until $E_1^{(k)}$ reaches some predefined threshold T. To prevent an allocation of too many pulses, we limit the maximum amount of pulses to NPMAX. Further, to prevent wasting pulses on silent segments, we first test whether the energy within the current frame exceeds some predefined level, and if not, no more than NPSLNT pulses are allocated. We also observed that the segmental SNR for voiced segments is much higher than for unvoiced segments. This is due to the impossibility of approximating a noisy signal with only a few pulses. However, even with a small number of pulses per segment, unvoiced frames still sound close to the original. During the pulse rate adaptation process we have to take care not to assign too many pulses to unvoiced parts. The *normalized prediction error, which is defined as the ratio of the residual* energy and the speech energy, is much higher for unvoiced sounds than for voiced sounds [Makhoul, 1973]. This means that the prediction error can be utilized as a scaling factor for $E_1^{(k)}$. The new measure $E_2^{(k)}$ is given by

$$E_2^{(k)} = E_1^{(k)} \cdot \sum_n r^2(n) \;/\; \sum_n s^2(n) \tag{4.16}$$

$$= \sum_n r^2(n) \;/\; \sum_n [e^{(k)}(n)]^2$$

where $s(n)$, $e(n)$ and $r(n)$ represent the input speech signal, the weighted difference signal and the residual signal, respectively. We used both measures $E_1^{(k)}$ and $E_2^{(k)}$ and adjusted the threshold T such that the average pulse rate was 800 pulses/second, and compared the results with those obtained with a fixed pulse rate of 800 pulses/second and heard no perceivable differences. Fig. 4.15 shows the behavior of the different parameters for an utterance by a female speaker. This utterance was analyzed with NPMAX=10, NPMIN=6 and NSLNT=2. Fig. 4.15(a) shows the energy in dB. In Figs. 4.15b) and 4.15c) the error measures $E_1^{(k)}$ and $E_2^{(k)}$ are drawn after locating the pulses. From these figures it is clear that $E_2^{(k)}$ shows a more constant behavior. The dotted lines represent the threshold levels of respectively 14 and 3.5. The number of pulses per search frame for the two different measures is shown in Figs. 4.15d) and 4.15e), and the resulting segmental SNR values for respectively measure $E_1^{(k)}$ and $E_2^{(k)}$ are shown in Figs. 4.15f) and 4.15g). The solid line represents the fixed rate, and the dashed line the variable rate.

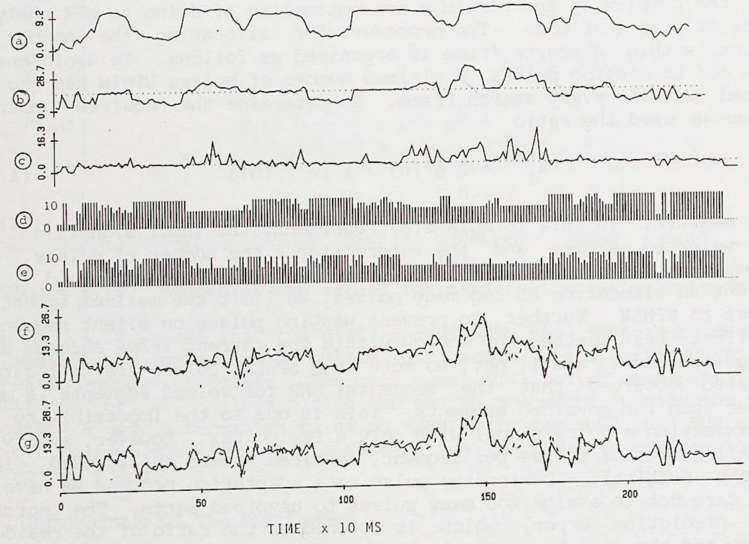


Figure 4.15. Pulse rate adaptation process. a) signal energy in dB, b) error E_1, c) error E_2, d) and e) number of pulses per search frame for respectively E_1 and E_2, f) and g) segmental SNR for fixed rate (solid line) and variable rate (dashed line).

4.5.2 *Use Of A Pitch Predictor*

In Chapter 2 we discussed the use of a pitch predictor and the procedures for determining its parameters. In this section we investigate the effect of the use of a pitch predictor [Kroon and Deprettere, 1984]. The update rate (NPRATE) of the pitch predictor coefficients is usually coupled to the update rate of the short-time prediction coefficients. In Fig. 4.16 SNR values for different update rates of the one-tap pitch predictor coefficients are shown. The range of M was chosen between 16 and 80.

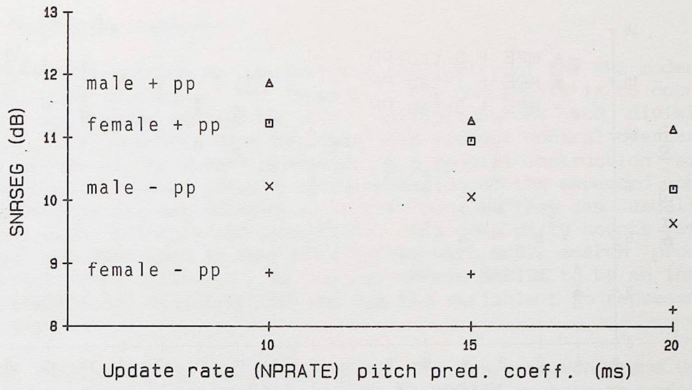


Figure 4.16. Segmental SNR values MPE with (+ pp) and without (- pp) pitch prediction for different update rates of the one-tap pitch predictor parameters (NPRATE=NRATE).

The improvement in SNR is less for lower rates but is still significant. Note that due to the limited range of M, female voices gain more in SNR than male voices. Fig. 4.17 shows the effect of pitch prediction on MPE analysis for different numbers of pulses. The curves for both a one-tap predictor and a three-tap predictor are drawn. For the three-tap predictor no correction was made for unstable filters. Despite the fact that about 7% of the filters were unstable, no undesired effects could be heard. A possible explanation is that the filters show a mild form of instability, and unstable filters are usually followed by stable ones. We found that replacing the unstable filters by stable one-tap filters had no perceivable effect and introduced a slight decrease in SNR.

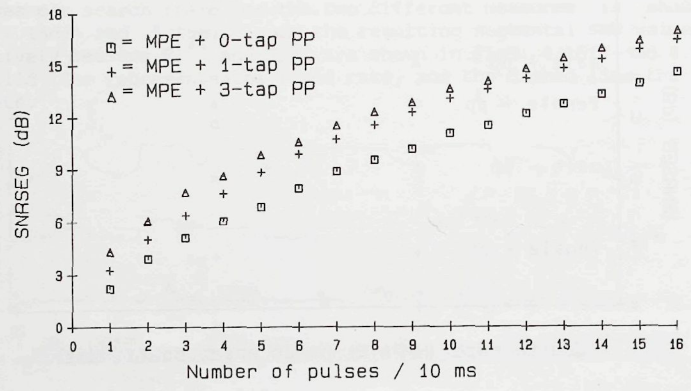


Figure 4.17. Improvement in segmental SNR by using pitch prediction in the MPE analysis procedure.

The effect is really dramatic, in both figures as well as in sound quality. The quantitative improvement in SNRSEG is about 2 dB for one-tap predictors, and about 3 dB for three-tap predictors. The same improvement can be obtained by allowing three to four additional pulses/10 ms. Although the three-tap predictor scores better (in terms of SNR) than the one-tap version, the perceivable differences between the two versions are small. If the number of bits required for transmitting the pitch prediction parameters is less than the bits required for transmitting these extra pulses, there is talk of a real improvement. Using the coding procedure described in [Atal, 1982], we need 10 and 19 bits per parameter set for the one-tap and the three-tap predictor, respectively, and approximately 8 bits for every pulse. From these figures we conclude that the use of a one-tap pitch predictor is really attractive in terms of bit rate and quality. Another advantage of pitch prediction is that it tends to remove the large pitch pulses, thereby reducing clipping effects in the quantized signal [Makhoul and Berouti, 1979]. This effect will be examined in detail in the section on quantization and coding. A disadvantage of using pitch prediction is not only additional complexity, but also its possible vulnerability to channel errors and background noises.

4.6 Complexity Reduction Of The MPE Coder

A disadvantage of the MPE analysis procedure is its complexity. The number of computations involved in finding the pulses is enormous. In this section we describe some alternatives for the pulse search procedure, which decrease the complexity without too much reduction in quality.

4.6.1 *Complexity Analysis*

In this sub-section we consider the complexity of the MPE coder. A commonly used procedure for measuring the complexity is to count the number of arithmetic operations (e.g. multiplications and divisions). Although this measure does not take into account control overhead and the attributes of the target processor (e.g special instruction set or architecture), it still gives a good impression of the expected complexity. Additions are not counted separately because they can usually be combined with multiply/add operations. All complexity counts refer to the algorithm described in Section 4.3 (MPE with SUB2 search procedure and direct form filters). We further choose NSSIZE to be an integral ratio of NRATE and NSSIZE=NMSIZE and use the variable L to represent the frame size.

4.6.1.1 *Initialization And State Update* For every search frame we have to compute the initial error $e^{(0)}$ [Eq.(4.6)] and we have to update the filter states, after finding NP pulses (i.e. computing $e^{(NP)}$. This means that we have to filter two times a frame of L samples through both the synthesis filter 1/A(z) and the error weighting filter W(z). These operations require 2*((p+3p)*L) multiplications, where p represents the order of the filters.

4.6.1.2 *Pulse Search Procedure* The k-th pulse has to be located in such a manner that $E^{(k)}$ is minimized. $E^{(k)}$ as a function of the pulse position and the previous error signal $e^{(k-1)}$ can be obtained from Eq.(4.10):

$$E^{(k)} = e^{(k-1)}e^{(k-1)^*} - e^{(k-1)}hw_m^*(e^{(k-1)}hw_m^*) / (hw_m hw_m^*) \qquad (4.17)$$

where hw_m is the m-th row of the matrix product HW. To minimize the squared error $E^{(k)}$ it can be seen that the best position for a single pulse is that value of m which maximizes the second term in Eq.(4.17). The denominator of the second term in this equation needs to be computed only once for every search frame, and requires L multiplications and L divisions. In addition, for every pulse the cross-correlation (numerator) has to be computed, which requires L*(1+L)/2 multiplications.

4.6.1.3 *Jointly Optimal Amplitude Computation* To compute the jointly optimal amplitudes, we have to solve a set of simultaneous linear equations, given by Eq.(4.9). The correlation matrix $G_kG_k^*$ is symmetric and positive definite; hence the set of equations can be solved using Cholesky decomposition. At first glance, the reader may think that for every new pulse we have to perform a Cholesky decomposition. However, the matrices $G_kG_k^*$ for different values of k are related to each other, which means that the Cholesky factors of the new matrix are also related. According to Cholesky's theorem, a positive definite symmetric matrix Q can be factored into the form $Q=CC^*$, where C is a lower triangular matrix. The Cholesky factor C at iteration step k can be obtained by combining the Cholesky factor of step k-1 and a new row. Moreover, the back-substitution procedure can be updated for every new iteration step.

The total number of operations required for solving the NP equation sets will necessitate approximately $O(NP^3)$ multiplications.

4.6.1.4 *Error Update* After finding a partial set of pulses $(k < NP)$, we have to compute the new error signal $e^{(k+1)}$. This can be done by filtering, but more efficiently by convolution, which requires approximately NP *(NP+1) *L/2 multiplications.

The total number of operations is given in Table 4.2.

operation	#operations
initialization + state update	8*p*L
pulse search procedure	NP*L(L+1)/2 + 2*L
amplitude computation	$O(NP^3)$
error updating	NP*(NP+1)*L/2

TABLE 4.2. Operation count MPE analysis procedure.

From this table we see that the complexity of the MPE coder is mainly determined by the pulse search procedure. The total number of operations required depends strongly on the values of the search frame size NSSIZE and the number of pulses per search frame (NP). Due to quality and bit rate constraints these parameters cannot be changed and other ways have to be found to reduce the complexity. In the following subsections we discuss some approaches that reduce the complexity of the analysis procedure.

4.6.2 *Auto-Correlation Analysis*

We minimize the mean-squared error over a frame of NMSIZE samples, and make no assumptions about the signal outside this frame. Here the matrix $HW(HW)^*$ (Section 4.3.1) is a matrix of covariance terms, and we refer to this approach as the covariance method.

When we extend the summation limits to $-\infty$ and $+\infty$. and assume the speech signal s(n) to be windowed such that it is zero outside the minimization frame, the matrix $HW(HW)^*$ will be a Toeplitz matrix of correlation terms. This auto-correlation formulation simplifies the search procedure, especially when we use the SUB1 approach. The denominator of the second term in Eq.(4.17) does not depend on m and the error $E^{(k)}$ is minimized by searching the position that maximizes $|e^{(k-1)} hw_m^*|$. This means that we avoid many time-consuming divisions. This approach was also described in [Araseki et al., 1983], and was called the maximum cross-correlation search algorithm. If we compare the SNR values obtained with the auto-correlation method with those obtained with the covariance method (Fig. 4.18),

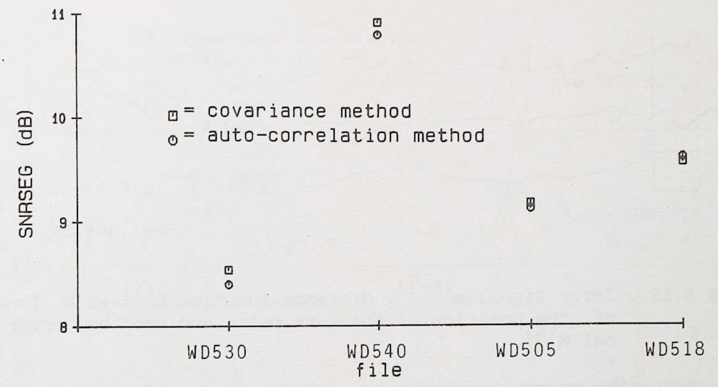


Figure 4.18. Differences in segmental SNR between auto-correlation method and covariance method.

we see that the latter gives slightly better results. Listening tests revealed that these differences were difficult to hear. In [Berouti et al., 1984] it was noted that for sine wave like inputs, the differences between the two methods become more significant.

4.6.3 *Reduction Of Search Frame Size*

In searching for optimal pulse positions, we have to evaluate the resulting error for every possible location. This procedure puts a major burden on the computational complexity (Section 4.6.1). To reduce the complexity, we have somehow to reduce the length of the search interval NSSIZE to an effective interval RSSIZE. A possible solution is suggested by the following (see also [Kroon and Deprettere, 1983]). From computer simulations we have observed that there is a strong correlation between the local minima of the distance function $E^{(k)}$ [Eq.(4.10)] as a function of the position of the k-th pulse, and the local concentrations of energy in the error signal $e^{(k-1)}$. A typical example is shown in Fig. 4.19.

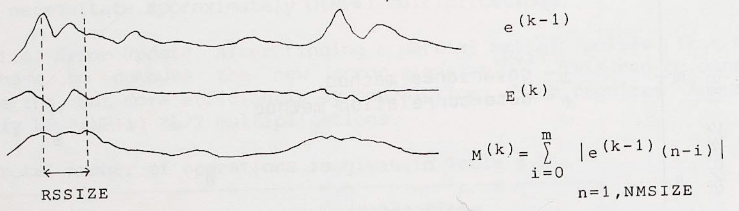


Figure 4.19. Error signal $e^{(k-1)}$, distance function $E^{(k)}$ as a function of the position of the k-th pulse and smoothed error signal $M^{(k)}$.

Hence short-time energy measurements could be used to detect the high-energy levels in the error signal, which will most likely correspond to the positions of the minima in $E^{(k)}$. We used an average magnitude function $M^{(k)}$ instead of a short-time energy measurement. $M^{(k)}$ is defined by

$$M^{(k)} = \sum_{i=0}^{m} |e^{(k-1)}(n-i)| \quad , n = 1,NMSIZE \tag{4.18}$$

where m is the integration interval. Our approach has been the following. If $M^{(k)}$ reaches its maximum value at n=nmax then Eq.(4.10) is only evaluated for locations falling within the interval [nmax-RSSIZE+1, nmax], where RSSIZE is the effective search interval (RSSIZE < NSSIZE). The position of the pulse that produces the minimum error $E^{(k)}$ will then be selected. Although RSSIZE will depend on the delay m of the integrator and the impulse response characteristics of the synthesis and error weighting filter, we have used reduced search frames with a fixed size. The optimal value for RSSIZE has to be determined experimentally, but should reduce the complexity without introducing too much degradation. The effect of a reduced search frame size for a male and a female speaker is shown in Figs. 4.20 and 4.21. In both figures the segmental SNR (computed every 10 ms) of the utterance "my father failed many tests" is shown.

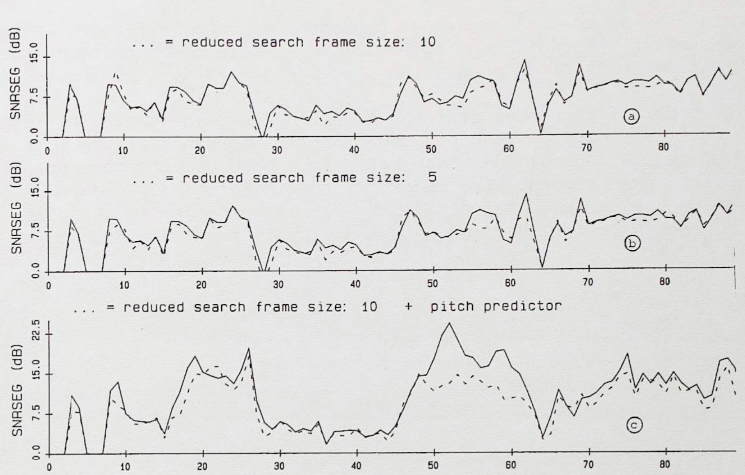


Figure 4.20. Effect of reduced search frame size for a male speaker. a) RSSIZE=5, b) RSSIZE=10 and c) RSSIZE=10 + pitch predictor.

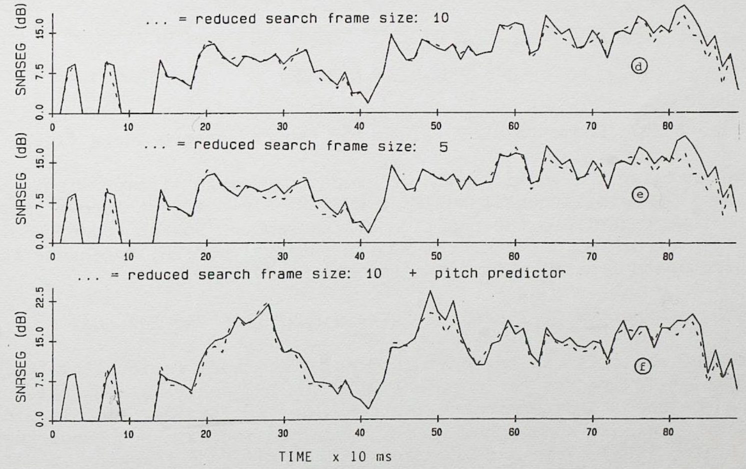


Figure 4.21. Effect of reduced search frame size for a female speaker. d) RSSIZE=5, e) RSSIZE=10 and f) RSSIZE=10 + pitch predictor.

The solid line represents the segmental SNR of the decoded speech signal produced by the original MPE coder. The dashed line represents the segmental SNR of the speech signal produced by the MPE-coder with a reduced search interval. The effect of the reduced search frame size on the MPE coder with pitch prediction (Section 4.5.2) is also shown.

The order of the integrator has to be chosen such that the signal $e^{(k-1)}$ is adequately smoothed, but transitions can still be followed. We found that a fourth-order integrator is a good compromise. Note that for every new segment the memory of the integrator is cleared. Fig. 4.22 shows the averaged segmental SNR values obtained with different reduced search frame sizes and with and without a pitch predictor.

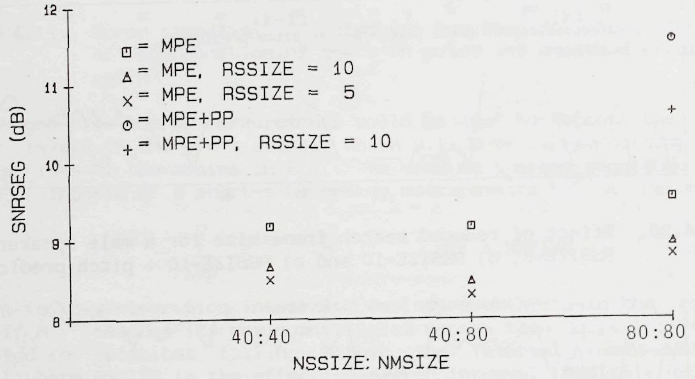


Figure 4.22. Segmental SNR values for reduced search frame sizes.

Observe that for the version with a reduced search frame size, the decrease in SNR is larger if NMSIZE > NSSIZE. This effect is due to the fact that we determined the position of the reduced search frame RSSIZE from the error $e^{(k-1)}$ over the search interval NSSIZE, to ensure that all pulses were located within the search interval.

4.6.4 *Determination Of The Multi-Pulse Signal From The Residual*

The MPE analysis can be considered as a weighted residual matching procedure. To reduce the complexity of the MPE analysis, the residual r(n) can be used to determine the multi-pulse excitation signal v(n). Two very similar approaches were described in [Jain and Hangartner, 1984] [Jain, 1983] [Jain and Hangartner, 1984] and [Parker et al., 1984], respectively. For both cases, the residual samples with maximum absolute values are chosen as estimates for the multi-pulse excitation. To incorporate the effect of the estimated pulses on the filter parameters, the linear prediction equations were reformulated and solved after finding the pulse excitation. This modified LP analysis can be extended to include error-weighting [Jain and Hangartner, 1984]. The whole analysis can be done in an iterative manner, until some specified error

criterion is met. The resulting speech quality is not discussed in both references, but is expected to be substantially less than the results obtained with the original coder. The reduction in complexity depends on the number of iterations, but can be high.

Based on the remark in Chapter 2 that the recomputation of the prediction parameters gives only a small improvement in speech quality, the following approach could also be useful. Compute the filter parameters and the corresponding residual, and feed this residual through a weighting filter $1/A(z/\gamma)$. Then use the procedure suggested in [Jain and Hangartner, 1984] to find the pulse positions and compute the amplitudes of the pulses according to Eq.(4.9). Although this procedure was not verified, a performance close to the other approaches described in this section can be expected.

4.7 Summary

In this chapter, we examined the multi-pulse coder. We investigated the influence of the various analysis parameters on the final speech quality using objective and subjective tests. The multi-pulse coder is capable of producing near toll-quality speech at medium bit rates. Male voices generally sound better than female voices, which can be partly explained by the difference in fundamental frequency. For a good quality of the reconstructed speech at bit rates around 10 kb/s, the following parameters are recommended. Eight pulses have to be located in a frame of 10 ms in duration, thereby minimizing a frame equal in duration. A good procedure for finding the pulses is sub-optimal method SUB2. Noise weighting slightly improves the perceived quality, when the proposed structure is used with a weight factor equal to 0.80.

We further examined different approaches to either improving the performance of the MPE coder or reducing the complexity. The best improvement is obtained by using a long-term pitch predictor. The variable number of pulses can be an advantage for non-real-time applications. Reduction in complexity can be obtained by decreasing the search interval size, or using a fixed noise weighting filter whereby both methods may introduce a slight degradation in the perceived speech quality.

References

Araseki, T., K. Ozawa, S. Ono, and K. Ochiai, "Multi-pulse excited speech coder based on maximum crosscorrelation search algorithm," *Proc. Globecom*, pp. 794-798 (1983).

Atal, B.S. and J.R. Remde, "A new model of LPC excitation for producing natural-sounding speech at low bit rates," *Proc. IEEE Int. Conf. Acoust., Speech and Signal Processing*, pp. 614-617 (April 1982).

Atal, B.S., "Predictive coding of speech at low bit rates," *IEEE Trans. Communications* Vol. **COM-30**(4) pp. 600-614 (April 1982).

Berouti, M., H. Garten, P. Kabal, and P. Mermelstein, "Efficient computation and encoding of the multipulse excitation for LPC," *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing*, pp. 10.1.1-10.1.4 (March 1984).

Jain, V.K., "Simplified algorithm for multi-pulse LPC analysis of speech," *Proc. Globecom*, pp. 789-793 (1983).

Jain, V.K. and R. Hangartner, "An efficient method for creating multi-pulse excitation sequences," *Proc. IEEE Int. Conf. Communications*, p. 48.5 (May 1984).

Jain, V.K. and R. Hangartner, "Efficient algorithm for multi-pulse-LPC analysis of speech," *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing*, pp. 1.4.1-1.4.4 (March 1984).

Kroon, P. and E.F. Deprettere, "On the design of LPC-vocoders with multi-pulse excitation," *Proc. European Conf. Circuit Theory and Design*, pp. 390-394 (Sep. 1983).

Kroon, P. and E.F. Deprettere, "Experimental evaluation of different approaches to the multi-pulse coder," *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing*, pp. 10.4.1-10.4.4 (March 1984).

Makhoul, J., "Spectral analysis of speech by linear prediction," *IEEE Trans. Audio and Electroacoustics* Vol. **AU-21** pp. 140-148 (June 1973).

Makhoul, J. and M. Berouti, "Predictive and residual encoding of speech," *J. Acoust. Soc. Amer.* Vol. **66**(6) pp. 1633-1641 (Dec. 1979).

Parker, A., S.T. Alexander, and H.J. Trussell, "Low bit rate enhancement using a new method of multiple impulse excitation," *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing*, pp. 1.5.1-1.5.4 (March 1984).

5. MULTI-PATH SEARCH CODING

5.1 Introduction

In the previous chapters we described parametric approaches to finding the optimum excitation sequence. The excitation sequence is defined by a small number of parameters, whose values have to be determined by an appropriate procedure. Although the number of parameters is limited, the values of these parameters can have many or an infinite number of values, and a separate quantization procedure is required for transmission and storage purposes. Such a quantization procedure affects the quality of the reconstructed speech signal (see also Chapter 6). A rather different approach, which does not require a separate quantization stage, is to select the excitation sequence from a limited number of possibilities. The two key components in such an approach are the search procedure and the selection of a collection of possible excitation sequences. In this chapter we discuss possible alternatives and report on the simulations done with a stochastic code book approach.

5.2 Search Procedures And Code Book Populations

Rather than defining the excitation sequence by a limited set of key parameters and searching for the optimal parameter values, we can directly search the optimum excitation sequence out of a set of alternatives. Such an approach is referred to as Multi-path Search Coding (MSC) [Fehn and Noll, 1982]. In this sub-section we describe two possibilities for implementing the search procedure: code book coding or tree and trellis coding.

A remaining problem is how to choose the appropriate candidate excitation sequences. Two methods that can provide an adequate performance are called stochastic and iterative methods [Fehn and Noll, 1982]. The stochastic method uses a population that is obtained from a random source having statistics similar to those of the signal to be encoded. The iterative method is based on an optimization procedure that uses a set of training data to select a representative set of data to populate the code book [Buzo et al., 1980]. The latter method is in general superior to the former [Fehn and Noll, 1982], but requires a significant amount of training data and becomes quite impractical for large code books and large vector sizes. In this chapter we consider only the stochastic approach.

5.2.1 *Code Book Approach*

In the code book approach the possible excitation sequences have been stored in a code book of the appropriate size. The optimum excitation sequence is found by trying every code book vector and selecting the one that produces the minimum weighted squared error E. The index of the selected sequence is transmitted and used at the receiver to retrieve the corresponding sequence. The number of bits per sample depends on both the size of the code book and the length of the excitation sequence. For vectors of length L and a code book size of N the number of bits /sample R is given by

$$R = \frac{\log_2 N}{L} \text{ bits/sample} \tag{5.1}$$

For increasing values of N and L the complexity increases exponentially. If a mean-squared error distance criterion is used, the number of multiply/add operations per sample is 2^{LR}.

5.2.2 *Tree And Trellis Coding*

In tree and trellis coding [Fehn and Noll, 1982] [Stewart et al., 1982] the excitation sequence cannot be chosen arbitrarily, but possesses a structure as determined by the tree or trellis. Fig. 5.1(b) shows an example of a tree. The tree can be divided into vertical sets of nodes, where each set corresponds to a time index n.

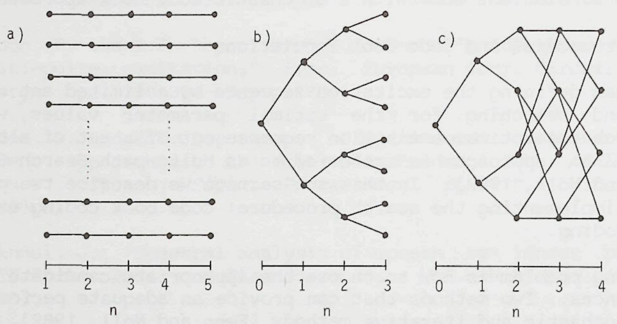


Figure 5.1. Code book (a), Binary tree (b) and trellis (c).

For a binary tree each node splits into two branches. Each branch is associated with a reconstruction value. The set of all these reconstruction values form the code book population. To determine a suitable excitation sequence all possible path sequences should be tried. The number of time samples used during the selection of the optimum path is called the tree depth L. The number of multiply-add operations per sample is $2(2^{LR}-1)/L$ and becomes excessive for large tree depths. In that case sub-optimal procedures have to be used. An effective search algorithm is called the M-algorithm [Anderson and Bodie, 1975] which limits the number of possibilities by considering only the latest M branches that results in a low cumulative error. By using this M-algorithm, the tree depth can be chosen larger than in the case of an exhaustive search and the net result will be that the performance will be better.

A trellis is more structured than a tree, and can be searched more efficiently. An example of a trellis is shown in Fig. 5.1(c). At each time instant only 2^j new elements are added. The value j is called the trellis intensity. To perform an exhaustive search in the trellis, the Viterbi algorithm can be used [Forney, 1973]. The number of multiply-add operations per sample is equal to 2^j and is independent of the trellis depth.

5.3 Delayed Decision Coding With Stochastic Code Book Populations

Both tree and code book coding approaches have been proposed with stochastic populations. The short-term correlations in the speech signal can be removed with a short-term predictor, and the resulting signal will have a probability distribution function (pdf) that resembles that of a Laplace pdf [Paez and Glisson, 1972]. When both short-term and long-term predictors are used the residual will have a pdf close to that of a Gaussian pdf [Atal, 1982]. The use of tree coders in the DDC scheme of Fig. 2.1 has been proposed in [Atal, 1982] for a Gaussian source, and in [Svendsen, 1984] for a Laplace source.

The use of a code book approach was proposed in [Atal and Schroeder, 1984]. Every code book vector contained 40 samples (5 ms at 8 kHz) and the code book consisted of 1024 vectors. This means that the resulting bit rate will be $(\log_2 1024)/40 = 1/4$ bit/sample. Each sequence of 40 candidate excitation samples is scaled in such a way that the energy of the excitation sequence is equal to the energy of the prediction error after both short and long delay prediction. The scaled excitation sequence is filtered through both the pitch synthesis filter $1/P(z)$ and the speech synthesis filter $1/A(z)$. The difference between the resulting synthetic signal and the original signal is appropriately weighted with $W(z)$, and the index of the sequence that produces the minimum error is transmitted. The coefficients of the predictors $A(z)$ and $P(z)$ are updated every 5 ms. For $W(z)$, a filter similar to that used in MPE and RPE analysis can be used. However, in [Atal and Schroeder, 1984] $W(z)$ was chosen to be

$$W(z) = Z(z)\, A(z)/A(z/\gamma) = [1 + \alpha z^{-1}]\, A(z)/A(z/\gamma) \tag{5.2}$$

and α and γ were chosen to be 1.0 and 0.90, respectively. The effect of $Z(z)$ will be that errors in the high frequency region will be de-emphasized for values of $\alpha > 0$.

Another approach is to determine the pitch predictor parameters and the gain factor during the error minimization procedure. The pitch predictor parameters are computed prior to the search for the optimal sequence, and are selected to minimize the initial squared error. Once the pitch predictor parameters have been found we search through the code book. For each vector the optimal gain value G_i is computed as follows. Let the vector $e^{(0)}$ represent the initial error due to the memory hangover of the filters $1/P(z)$, $1/A(z)$ and $W(z)$. Let the vector y_i be the result of filtering the i-th excitation vector v_i through the cascade of $1/P(z)$, $1/A(z)$ and $W(z)$. The value of G_i is determined to minimize the squared difference

$$E = (e^{(0)} - G_i y_i)(e^{(0)} - G_i y_i)^* \qquad (5.3)$$

By setting the derivative of Eq. (5.3) with respect to G_i equal to zero, we obtain for the optimal value of G_i

$$G_i = (e^{(0)} y_i^*) \,/\, (y_i y_i^*) \qquad (5.4)$$

Fig. 5.2 gives an example of the original, reconstructed and excitation waveforms as produced by this approach. The excitation sequence is encoded with a 1/4 bit per sample.

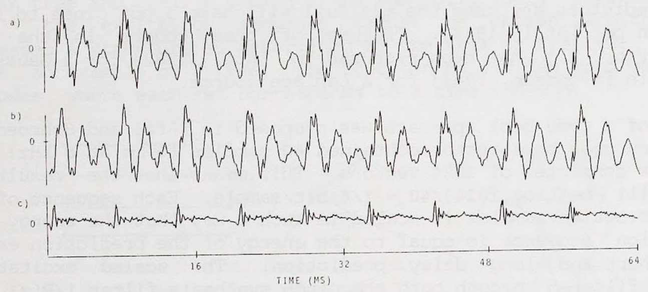


Figure 5.2. Waveforms of the original speech (a), the reconstructed speech (b), and the excitation signal (c) with a code book containing 1024 vectors of 5 ms each. The code book is populated with samples from a Gaussian noise generator.

5.4 Error Minimization Procedure

5.4.1 *Definitions*

5.5 Experimental Evaluation Of The Code book Approach

In this section we describe the result obtained with the procedures explained in the previous section. The procedures are extremely complex and due to lack of computing power we carried out the simulations with only one utterance of 1.5 s (one male and one female speaker). The predictor coefficients were obtained by the auto-correlation method with a 20 ms Hamming window. The coefficients were updated every 5 ms and the filter order was 16. A 3-tap pitch predictor was used and the coefficients were chosen to minimize the error over 5 ms frames. The value of the delay parameter M was determined using the covariance method (see Chapter 2), and the range of M chosen was between 143 and 16. The procedure described in [Atal and Schroeder, 1984] did not give satisfying results. The reconstructed signal was perceived as noisy, and adjustment of the values of γ and α did not lead to improvement. Better results were obtained with the procedure that computes the gain factor and the

pitch predictor parameters during the minimization procedure. Table 5.1 lists some of the combinations we have examined.

label	n-tap	pp	range	weighting	SNRSEG	SNRSEG
aa	1	int	16,143	Z+W	11.86	12.55
bb	1	int	16,143	W	12.19	13.58
cc	3	int	16,143	Z+W	11.42	12.38
ee	3	int	16,143	W	12.99	14.33
gg	1	ext	16,143	Z+W	10.38	11.12
pp	1	ext	16,143	W	10.01	11.54
jj	3	ext	16,143	Z+W	10.59	11.95
qq	3	ext	16,143	W	11.05	12.15

TABLE 5.1. Coder combinations.

The specification int/ext indicates whether the pitch predictor parameters were determined from the residual signal, or during the analysis, i.e. to minimize the weighted error. For all cases the results obtained with the additional high-frequency de-emphasis filter were less good than those obtained without this filter. The one-tap predictor produced the best results if its parameters were determined inside the loop. The three-tap predictor was responsible for disturbing pops and clicks if the parameters were determined inside the loop. When we switched to out-loop computation, these effects disappeared. However, the best results were still those obtained with the internal 1-tap predictor (bb-version). The reconstructed signals sounded quite good, but when they were compared to the original signal, differences could be heard. In general, the sound is less clear than that of the original, but comparable to results obtained with the RPE and MPE coders at a much higher rate. The segmental SNR values of the results obtained for the utterance "a lathe is a big tool" with the procedure labeled bb are shown in Fig. 5.3. In the same figure the SNR values obtained with an external pitch predictor (pp) are shown.

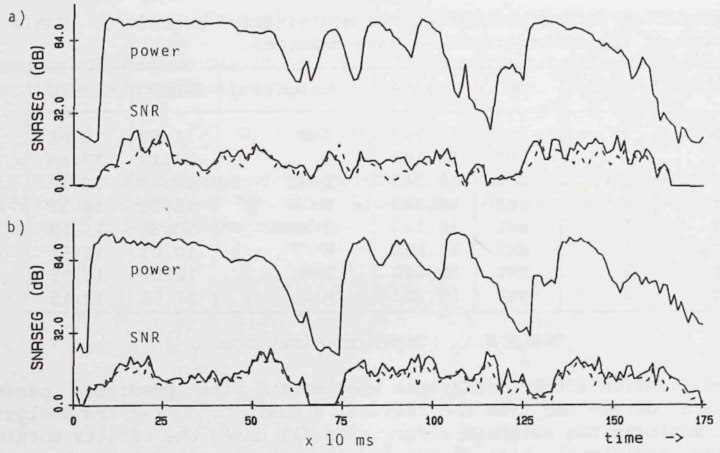


Figure 5.3. Segmental SNR values for a female speaker (a) and a male speaker (b) using procedures bb (solid line) and pp (dashed line). The upper curve represents the speech power + 25 dB.

The power spectra of a 32 ms segment input speech and reconstructed speech are shown in Fig. 5.4.

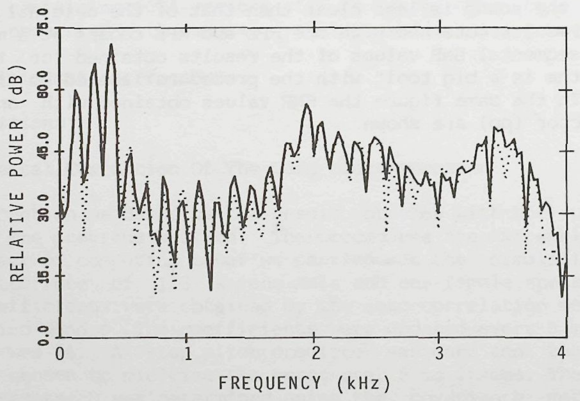


Figure 5.4. Power spectra of a 32 ms segment of input speech (solid line) and the reconstructed speech segment (dashed line).

The resulting error spectrum of the difference between s(n) and ŝ(n) is shown in Fig. 5.5.

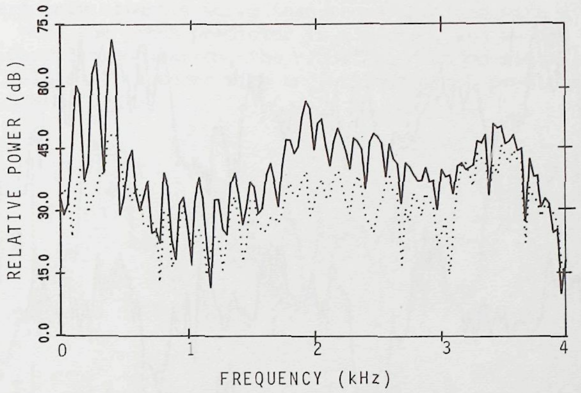


Figure 5.5. Power spectra of a 32 ms segment of input speech (solid line) and of the corresponding error signal s(n) - ŝ(n) (dashed line).

The predictor update rate is fast (5 ms), and many bits are required for encoding the predictor parameters. Reducing the update rate to 10 or 15 ms, and using linear interpolation between the coefficients (see also Chapter 2) gave a slight deterioration in synthetic speech quality, but the result was still acceptable. Further reduction of the update rate to 20 ms resulted in unacceptable distortions.

A reduction in the code book size from 1024 to 512 has no significant influence on the SNR and the perceived quality. The SNR values as a function of time for the bb-version with different code book sizes are shown for both a male and a female speaker in Fig. 5.6.

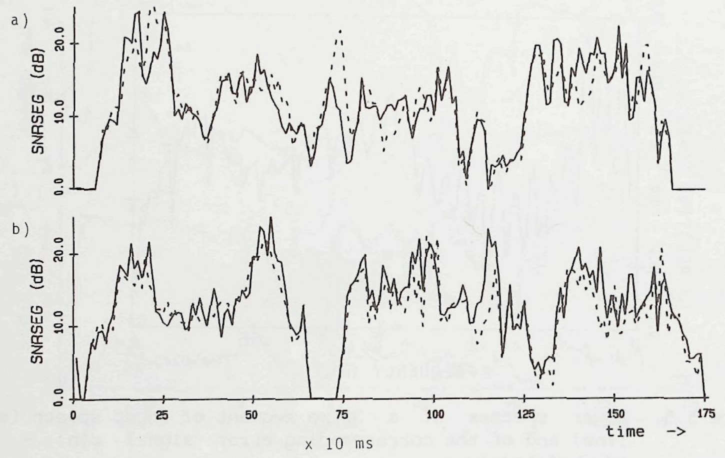


Figure 5.6. Different code book sizes for the bb-procedure. NCSIZE=1024 (solid line), NCSIZE=512 (dashed line).

From this experiment we concluded that a code book size of 1024 is a reasonable choice, and that more efficient code books should be obtained by choosing a more appropriate code book population rather than increasing the code book size. Fig. 5.7 shows, for a typical voiced sound, the normalized weighted squared error as a function of the code book index. The error has been normalized to the initial error $e^{(0)}$ and for each code book vector (NCSIZE=512), the resulting error has been drawn. The search procedure selects the index that produces the smallest error, and it is easily seen that many candidates are alike.

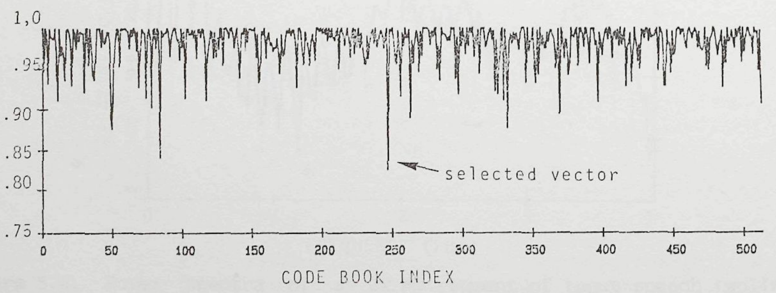


Figure 5.7. Normalized error as function of the code book index.

The use of a uniform noise source to populate the code book gave results

that were only slightly worse than those obtained with a Gaussian population. When the pitch predictor is not used, and we use a code book with a Gaussian population, the resulting decoded signal sounded noisy. The SNR values for a coder with and without pitch prediction are shown in Fig. 5.8.

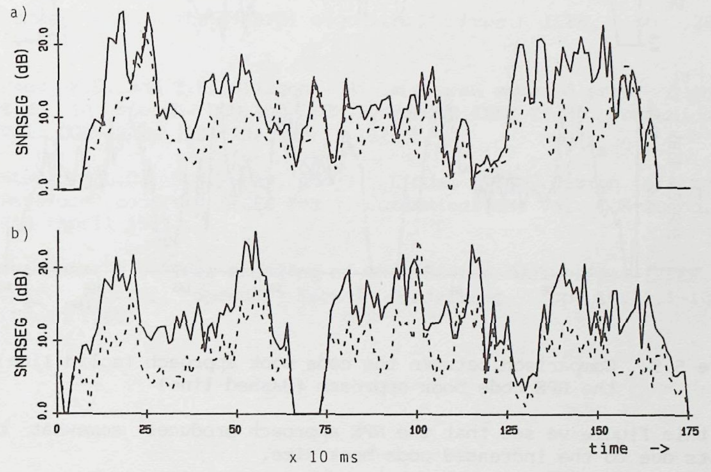


Figure 5.8. Performance of the bb-procedure with (solid line) and without pitch predictor (dashed line).

Using a Laplace source to populate the code book while not using the pitch predictor yielded no real improvement. The code book samples in our simulations were represented in floating point notations. For practical realizations, a fixed point representation will be used. The number of bits used should be chosen as small as possible to reduce the code book size. In [Fehn and Noll, 1982] it was reported that at least 4 bits should be used to represent a code book sample.

In another experiment we used the stochastic code book populated with Gaussian noise to generate regular-pulse excitation strings. For each search frame, NCSIZE regular-pulse excitation strings are generated by taking adjacent frames from the code book. This approach is repeated for every offset value, resulting in an effective code book of size NS*NCSIZE, where NS represents the pulse spacing. For each possibility, the optimum gain is computed, and the offset factor and the code book index are selected to yield the minimum weighted mean squared error. In Fig. 5.9 we show the resulting segmental SNR values for a male and a female speaker (solid line). For comparative purposes we have also drawn the results obtained with the pp-version (dashed line).

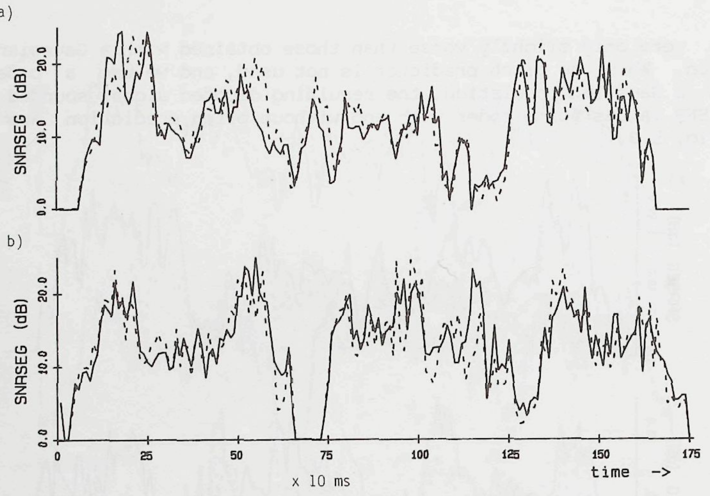


Figure 5.9. Comparison between the code book approach (solid line), and the RPE code book approach (dashed line)

From this figure we see that the RPE approach produces somewhat better results due to the increased code book size.

5.6 Summary

In this chapter we described a multi-path search approach to determining the optimum excitation sequence. Two basic procedures can be distinguished: the code book approach and the tree or trellis approach. We described some preliminary experiments with the code book approach, and it was concluded that the results are comparable to those of the MPE and RPE coders but at a lower bit rate. It was further demonstrated that the use of a pitch predictor is mandatory for good results.

References

Anderson, J.B. and J.B. Bodie, "Tree encoding of speech," *IEEE Trans. Inform. Theory* Vol. IT-21(4) pp. 379-387 (July 1975).

Atal, B.S., "Predictive coding of speech at low bit rates," *IEEE Trans. Communications* Vol. COM-30(4) pp. 600-614 (April 1982).

Atal, B.S. and M.R. Schroeder, "Stochastic coding of speech signals at very low bit rates," *Proc. IEEE Int. Conf. Communications*, p. 48.1 (May 1984).

Buzo, A., A.H. Gray, R.M. Gray, and J.D. Markel, "Speech coding based upon vector quantization," *IEEE Trans. Acoust., Speech, Signal Processing* Vol. ASSP-28 pp. 562-574 (Oct. 1980).

Fehn, H.G. and P. Noll, "Multipath search coding of stationary signals with applications to speech," *IEEE Trans. Communications* Vol. COM-30(4) pp. 687-701 (April 1982).

Forney, G.D., "The Viterbi algorithm," *Proc. IEEE*, pp. 268-278 (1973).

Paez, M.D. and T.H. Glisson, "Minimum mean squared error quantization in speech PCM and DPCM systems," *IEEE Trans. Communications* Vol. COM-20 pp. 225-230 (April 1972).

Stewart, L.C., R.M. Gray, and Y. Linde, "The design of trellis waveform coders," *IEEE Trans. Communications* Vol. COM-30 pp. 702-710 (April 1982).

Svendsen, T., "Tree encoding of the LPC residual," *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing*, pp. 10.11.1-10.11.4 (March 1984).

6. QUANTIZATION AND CODING

6.1 Introduction

To enable the transmission of the speech coder parameters over finite rate channels, these parameters must be represented by a finite set of symbols in such a way that the channel rate is not exceeded. Some types of coders inherently produce a limited set of symbols (e.g. code book indices in Multi-path Search Coding), while other types need a separate quantization stage to map the real valued parameters into a finite subset of real values. Each output point of such a quantizer can be associated with a binary word suitable for storage or transmission. Any quantization procedure that is not incorporated in the encoding scheme will adversely affect the coder performance. Obviously, the optimal mapping onto a finite set of reals minimizes the number of output symbols without noticeable degradation of the synthetic speech quality. In this chapter we examine quantization and coding procedures for the MPE and RPE coders. We investigate the degradation caused by quantization of the coder parameters and describe efficient procedures for coding these quantized parameters. The MPE and RPE coders both produce two different parameters sets:

1. excitation samples,

2. filter coefficients,

for which different quantization and coding strategies exist. Quantization of excitation samples and filter coefficients will be discussed separately.

To protect the data against channel errors extra bits can be added for error correction, thereby increasing the total bit rate. In the remainder of this chapter error free transmission is assumed.

6.2 Quantizing And Encoding The Excitation Parameters

The excitation of the RPE coder consists of frames of regularly spaced pulses, with different amplitudes (Chapter 3). At the beginning of every new frame, the position of the first pulse relative to the beginning of this frame is specified by an offset factor. Within a frame the spacing between the pulses is fixed and only the amplitudes and the offset factor have to be encoded. The excitation of the MPE coder consists of a set of non-uniformly spaced pulses with different amplitudes (Chapter 4). For every pulse the position within a frame and its amplitude have to be encoded.

Sample by sample encoding of the excitation signal is very inefficient due to the high amount of zero valued samples. Coding only the non-zero samples is obviously a better approach. We can consider the pulse sequence as a discrete-time random sequence with amplitude values taken from an infinite (or very large) set of amplitudes. In order to transmit or to store this sequence, the sample values must be quantized to yield a limited set of amplitudes so that they can be represented by a finite set of symbols. In quantizing the non-zero pulses a similar approach can be followed for both the RPE and MPE coders, and we describe this in one single section. The encoding of the pulse positions for the MPE signal will be described in a separate section.

6.2.1 *Excitation Signal Statistics*

Before we started our simulations with different quantization techniques we computed some signal statistics from speech samples in a representative database. All signals were obtained with the analysis procedures summarized in Tables 2.1, 3.1 and 4.1. In computing the statistics only the pulse amplitudes are considered and the pulses are normalized for frames of 10 ms. The normalization enables us to use fixed quantizer ranges and to use the same ranges for computing the histograms. To exclude silent intervals the rms value of a set of pulses within a frame must exceed a threshold T (T=-52 dB relative to the maximum rms value). A histogram of the amplitude distributions was obtained by first dividing the input range up into L equally spaced bins, and thereafter counting the number of amplitudes that falls into each bin. The signal was normalized to unit variance ($\sigma^2 = 1$) by estimating the variance σ^2 from short-time (10 ms) energy measurements. The range was chosen between -4σ and $+4\sigma$ and 128 histogram bins were used (Fig. 6.1).

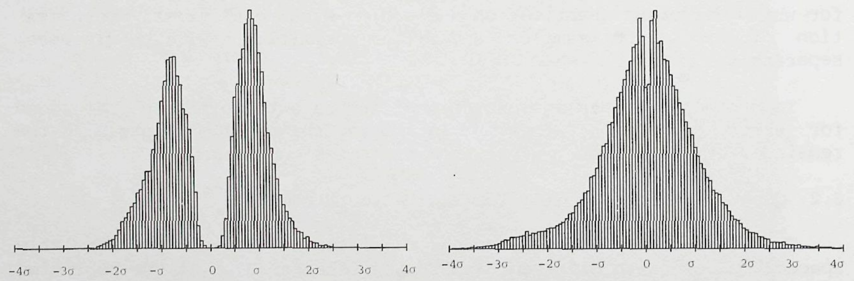


Figure 6.1. Amplitude distribution histograms for unit variance signals (a) MPE, (b) RPE.

The MPE signal has a bimodal distribution function, while the RPE signal tends more to a Laplace distribution. Both types of signals reveal a sparse density for small valued samples, which is more significant for the MPE pulses than for the RPE pulses. In Fig. 6.2 the amplitude histograms are given for both signals normalized on the maximum amplitude within a 10 ms frame.

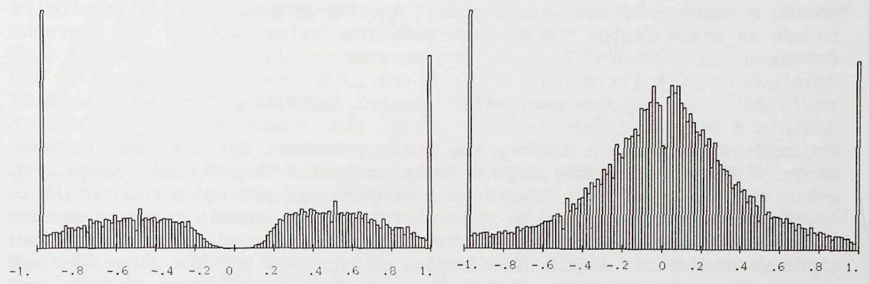


Figure 6.2. Amplitude distribution histograms for amplitude normalized signals (a) MPE, (b) RPE.

The distribution functions of excitation signals normalized on 20 ms frames have the same shape as the functions of Figs. 6.1 and 6.2, but the density for large amplitude values is somewhat higher.

6.2.2 *Quantizer Selection*

To match the quantizer characteristics to the variations of the input level, the quantizer step size can be adjusted proportional to the signal variance [Noll, 1974]. In practical systems the quantizer is fixed and the signal is normalized by multiplication with a scale factor. This scale factor can be computed from samples of the input signal and has to be transmitted to the receiver as side information. This procedure is called forward estimation. Another possibility is to estimate the scale factor from the quantized samples. This procedure is called backward estimation and needs no separate transmission of the scale factor but is generally less optimal than forward estimation [Noll, 1974].

The scale factor of the forward adaptive quantizer is estimated over NGSIZE samples and updated for every new frame of NGSIZE samples. The rate at which such an adaptation should take place depends on the characteristics of the input process. For speech we know that it can be considered quasi-stationary for segments of 10 to 20 ms in duration. This rate will also be suitable for the adaptation rate of the quantizer.

To obtain a unit-variance signal, the scaling factor G should be proportional to the standard deviation of the input signal. An estimate of the standard deviation is given by

$$G = \left[\frac{1}{NGSIZE} \sum_{i=1}^{NGSIZE} x^2(i) \right]^{\frac{1}{2}} \qquad (6.1)$$

The quantizer range can be chosen to be n times the standard deviation σ

where n represents the load factor. Another method of normalization is to use as scale factor the maximum absolute value within the current frame

$$G = \max\{|x(i)|\} \quad i=1,\ldots,NGSIZE \tag{6.2}$$

For both normalization methods the scaling factor follows the dynamic range of the input signal. To be able to follow this dynamic range with a limited number of bits logarithmic companding techniques such as μ-law or A-law can be used. To be able to follow the dynamic range of speech signals (typically 30 dB) at least 4 bits are required. In our simulations we used 6 bit logarithmic coding of the gain and the range was set between 0 and 63 dB. With these characteristics the rms values and amplitude values between 1 and 1333 can be encoded.

A zero-memory N-level quantizer Q may be defined by specifying a set of N+1 decision levels $x_0, x_1, \ldots, x_N$ and a set of N output points $y_1, y_2, \ldots, y_N$. If the value x of an input sample lies in the i-th quantization interval, i.e.,

$$x \in R_i = \{x \mid x_{i-1} \le x < x_i\}, \quad i=1,\ldots,N \tag{6.3}$$

the quantizer produces the output value y_i. Since y_i is used to approximate samples contained in the interval R_i, y_i is itself chosen to be some value in the interval R_i. The interval R_i is called the quantizer step size Δ and is constant for uniform quantizers. The boundary levels x_0 and x_N are chosen to be equal to the smallest and largest values, respectively, that the input samples may have. Both uniform and non-uniform quantizer characteristics can be used. Given the probability density function (pdf) of the input signal, a non-uniform quantizer that minimizes the quantization noise for a given number of levels N can be designed. The design procedure for such a quantizer was independently proposed in [Lloyd, 1957] and in [Max, 1960], and this optimum quantizer is commonly referred to as a Lloyd-Max quantizer.

For the mean-square error criterion, with some fixed value of N, the optimal values for $x_1, x_2, \ldots, x_{N-1}$, and $y_1, y_2, \ldots, y_N$ are found by minimizing the mean-square distortion measure D:

$$D = \sum_{i=1}^{N} \int_{x_{i-1}}^{x_i} (y_i - x)^2 \, p(x)\, dx \tag{6.4}$$

where p(x) is the probability distribution function of x. By setting the derivatives of D with respect to the x_i and y_i parameters to zero we obtain the following conditions:

$$y_i = \int_{x_{i-1}}^{x_i} x\, p(x)\, dx \Big/ \int_{x_{i-1}}^{x_i} p(x)\, dx \quad i=1,2,\ldots,N \tag{6.5a}$$

$$x_i = \tfrac{1}{2}\,[y_i + y_{i+1}] \quad ,i=1,2,\ldots,N-1 \tag{6.5b}$$

Thus 1) each output point y_i must be the centroid or center of mass of the interval R_i with respect to the input density p(x); 2) each decision level must be halfway between the two adjacent output points. The given conditions are necessary but not sufficient conditions for a minimum error. An additional condition, i.e. log p(x) is concave, is required to assure that the quantizer is indeed optimal [Fleischer, 1964]. This condition holds for the Gaussian density as well as for many other common densities. With an iterative procedure [Lloyd, 1957] [Max, 1960] the optimal values of x_i and y_i can be calculated from the conditions in Eq.(6.5). A value for y_1 is selected, Eq (6.5a) is solved for x_1, Eq.(6.5b) is solved for y_2, etc., until a value for y_N is obtained from Eq.(6.5b). The iteration will be stopped if the difference between this value and the one obtained from Eq.(6.5a) is small enough. Otherwise y_1 is appropriately perturbed and the iteration is continued until the desired error criterion is achieved.

The number of levels can be chosen to be odd or even. Odd numbers result in mid-tread quantizers and even numbers in mid-riser quantizers. A disadvantage of mid-riser quantizers is their inability to represent zero levels, which results in an effect called idle channel noise. Taking into account the shape of the amplitude distribution function and the limited number of pulses (especially for MPE signals) we can neglect this effect and use an even number of quantizer levels. The actual number of levels depends on the allowed bit rate and the number of pulses, and will be about 8 levels for both types of coders at a bit rate of around 6400 b/s.

6.2.3 *Location Of The Quantizer*

When we adapt the quantizer step size for every frame of NSSIZE samples (forward adaptive quantization), we can include the quantizer in the analysis procedure. If the quantization procedure is embedded in the pulse search procedure (MPE coder) it is expected that the outcome of the search procedure will compensate to a certain extent for quantization errors. One possibility is to quantize a complete set of pulses at once. In this case only the initial error $e^{(0)}$ in the next search frame is a function of the quantization process, and there is no possibility that the search procedure can compensate for quantization errors in the current frame. The MPE procedure SUB1 (Section 4.3.4) locates the pulses one by one, thereby offering the possibility of instantaneous quantization, i.e. once a pulse has been found it can be quantized immediately. In an extreme case the search procedure (SUB1 or SUB3) can be executed with quantized pulses.

The RPE coder finds all pulses within a search frame at once. Different sets are computed and the set that produces a minimum error is selected. If the resulting set is quantized, only the selection of the excitation sequence in the next search frame is affected by the quantization operation. To incorporate the pulse quantization in the minimization procedure we can quantize each set separately, and select the quantized set that produces the minimum error.

6.2.4 *Quantizer Parameters*

In our experiments concerning the optimum quantization procedure for MPE and RPE coders we did not quantize the reflection coefficients. To evaluate the obtained results we again used the segmental SNR measurements between the original and the synthetic speech signal. We are aware that these measures need to be interpreted carefully. For example, SNR measurements tend to overestimate clipping errors [Makhoul and Berouti, 1979]. In addition to these measurements we evaluated all experiments with informal listening tests. Three different quantizers were used:

1. Uniform quantizer with normalization to the absolute maximum,
2. Non-uniform quantizer with normalization to the absolute maximum,
3. Non-uniform quantizer with normalization to the standard deviation

The quantizer characteristics are shown in Table 6.1 for normalization to the maximum absolute value and in Table 6.2 for normalization to the standard deviation.

i	Uniform		MPE		RPE	
	x	y	x	y	x	y
0	-1.000	-	-1.000	-	-1.000	-
1	-0.750	-0.875	-0.829	-0.954	-0.751	-0.930
2	-0.500	-0.625	-0.598	-0.704	-0.440	-0.572
3	-0.250	-0.375	-0.386	-0.492	-0.203	-0.307
4	0.000	-0.125	0.003	-0.280	0.002	-0.098
5	+0.250	+0.125	+0.375	+0.273	+0.195	+0.093
6	+0.500	+0.375	+0.583	+0.478	+0.427	+0.297
7	+0.750	+0.625	+0.816	+0.688	+0.734	+0.557
8	+1.000	+0.875	+1.000	+0.944	+1.000	+0.912

TABLE 6.1. 8-level Lloyd-Max quantizer characteristics for MPE and RPE signals normalized to the maximum absolute value (10 ms frames).

i	MPE		RPE	
	x	y	x	y
0	$-\infty$	-	$-\infty$	-
1	-1.501	-1.762	-2.003	-2.542
2	-1.067	-1.257	-1.119	-1.463
3	-0.703	-0.878	-0.514	-0.773
4	-0.009	-0.528	-0.002	-0.252
5	+0.673	+0.509	+0.507	+0.246
6	+1.008	+0.836	+1.094	+0.768
7	+1.446	+1.179	+1.911	+1.420
8	$+\infty$	+1.712	$+\infty$	+2.403

TABLE 6.2. 8-level Lloyd-Max quantizer characteristics for MPE and RPE signals normalized to the rms value (10 ms frames).

The gain was quantized with 6 bits and updated every 10 ms. For each coder the pulses within a frame were determined without quantization during the search procedure. Once the pulses were found, they were quantized and the quantized excitation sequence was used to compute the initial error $e^{(0)}$ of the next search frame. Fig. 6.3 shows the SNR values obtained by averaging the results of four different utterances for the MPE and RPE coder, respectively.

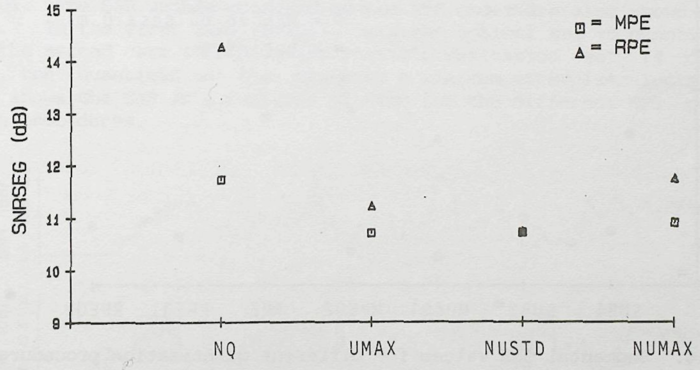


Figure 6.3. Segmental SNR values for different quantizers; NQ=not quantized, UMAX=uniform quantizer with normalization to the absolute maximum, NUMAX and NUSTD= non-uniform Lloyd-Max quantizers with normalization to respectively the absolute maximum and the standard deviation.

A comparison between the SNR for unquantized and quantized signals reveals that the difference in SNR is higher for the RPE coder. However, the ratio in SNR between the MPE and the RPE is retained. Non-uniform quantization is better than uniform quantization and normalization to the maximum absolute value is preferred over normalization to the standard deviation. These results were confirmed by listening tests.

6.2.5 *Quantization During The Minimization Procedure*

As we mentioned in the beginning of this chapter, we can also incorporate the quantizer in the minimization procedure. For the MPE coder we use minimization procedure SUB1 (Section 4.3.4), which locates one pulse at a time and does not perform joint amplitude optimization. Every newly found pulse is immediately quantized and the effect of the quantized pulse is subtracted from the initial error before proceeding with the search for the next pulse. In almost all cases quantized amplitude and unquantized pulses are not identical, thereby introducing additional sub-optimality. As a result the error minimization procedure may allocate the next pulse to the location of the previous pulse. To preclude this possibility, we have to exclude the locations of already found positions. We refer to the SUB1 quantization procedure as MPEQ2 and to procedure SUB2 with one quantization step for all pulses as MPEQ1. Fig. 6.4 shows the differences in SNR for MPEQ1 and MPEQ2.

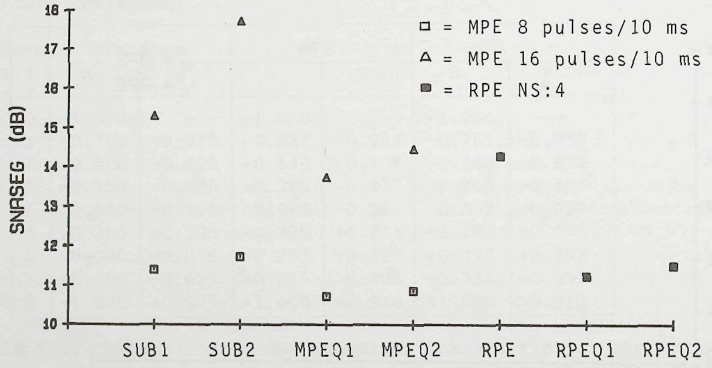


Figure 6.4. Segmental SNR values for different quantization procedures.

The scaling factor is normally determined from the whole set of pulses. For normalization on the maximum value we need all pulses, but if the pulses are quantized one by one the maximum value of the complete set is not known. Experiments reveal that the pulse with the maximum value within a set usually corresponds to the first pulse found. The maximum absolute value within a set is then approximated by the absolute amplitude value of this first pulse. Fig. 6.5 shows the segmental SNR values for the different procedures as a function of time.

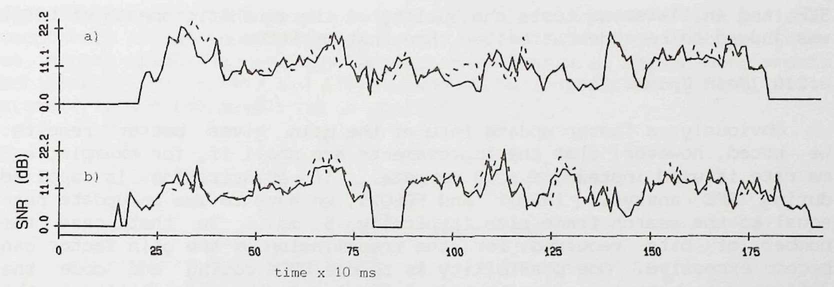


Figure 6.5. Segmental SNR of different quantization procedures for the MPE coder; MPEQ1 (solid line), MPEQ2 (dashed line) for female and male speaker.

From these figures and the listening test we conclude that procedure MPEQ2 with pulse by pulse quantization produces somewhat better results than procedure MPEQ1 with segment quantization. In Fig. 4.11 we saw that for the unquantized pulses, the difference between minimization procedures SUB1 and SUB2 was small for 8 pulses/10 ms but more significant for 16 pulses/10 ms). We compared the behavior of the procedures MPEQ1 and MPEQ2 for 16 pulses/10 ms (see Fig. 6.4) and found that even for this amount of pulses the performance of MPEQ2 was better than MPEQ1. The SNR values produced by the RPE coder are also shown in Fig. 6.4. In the first case (RPEQ1), only the optimal set is quantized, and in the second case (RPEQ2) every possible excitation set is quantized and the quantized set that produces a minimum error is selected. Fig. 6.6 shows the SNR as a function of time for the different RPE quantization procedures.

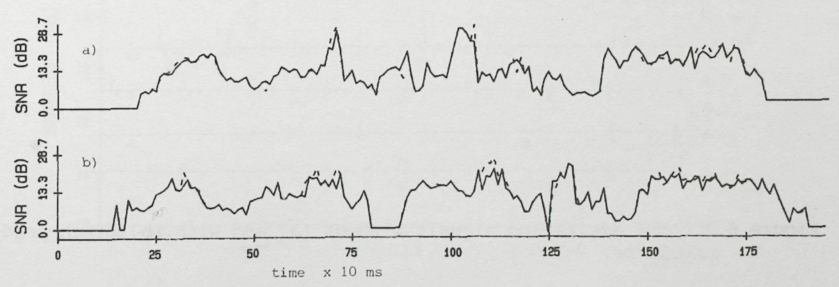


Figure 6.6. Segmental SNR of different quantization procedures for the RPE coder; RPEQ1 (solid line), RPEQ2 (dashed line) for female and male speaker.

For both cases the quantizer gain is updated for every search frame (typically 5 ms). From these figures we see that RPEQ2 yields a higher

SNR, and in listening tests the quality of the synthetic speech of RPEQ2 was judged to be somewhat better than that of RPEQ1.

6.2.6 *Gain Update Rate*

Obviously, a faster update rate of the gain gives better results. We noted, however, that the improvements are small if, for example, a 5 ms rate is used instead of a 10 ms rate. If quantization is applied during RPE analysis (RPEQ1 and RPEQ2), we have to use an update rate equal to the search frame size (typically 5 ms). In that case the number of bits required for the transmission of the gain factor can become excessive. One possibility is to use DPCM coding and code the differences between adjacent gain factors. Another possibility is the following. The scale factor is determined for a large frame (e.g. 20 ms), and the frame is divided into 2 to 4 sub-frames. For every subframe a new gain factor is determined and the resulting gain is coded relative to the gain factor of the whole frame. The sub-gain factors can be expressed as multiples of the main gain with, for example, 2 bits (1/4, 2/4, 3/4, and 4/4). If the scale factor of the main frame is not known (as is the case with RPE coders), it can be estimated from the residual signal.

6.2.7 *Improved Quantizer Performance*

Quantization of the excitation signal deteriorates the synthetic speech quality. In this sub-section we describe methods to reduce these effects.

6.2.7.1 *Use Of A Pitch Predictor* Figs. 6.7(a) and 6.8(a) show the excitation signals corresponding to a voiced signal for the MPE and RPE coder, respectively.

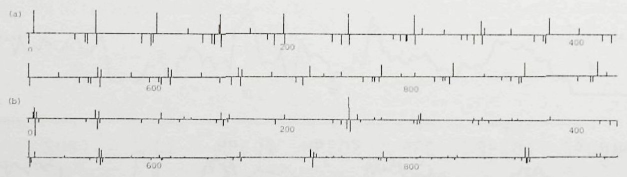


Figure 6.7. MPE excitation signals without (a) and with (b) pitch predictor.

In both sequences a regular excitation pattern can be recognized. These pulses are due to the periodic structure of the input signal (pitch pulses). The large values of these pulses make the normalization process be dominated by the value of these pitch pulses. As a result the remaining, and usually smaller, pulses will be quantized very roughly. To obtain a better match between input pulses and the quantizer characteristic we have to remove these large pulses. In Chapter 2 we

discussed the use of a pitch predictor for the synthesis of the periodic components in the input signal. By the application of a pitch predictor we expect that the remaining excitation pulses will show a more smooth behavior. Figs. 6.7(b) and 6.8(b) show the corresponding excitation signals if a pitch predictor is applied.

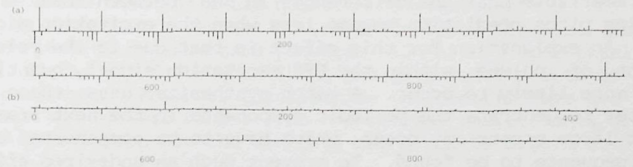


Figure 6.8. RPE excitation signals without (a) and with (b) pitch predictor.

From this figure we see that pitch prediction removes to a certain extent the large pitch pulses. The resulting signal can now more easily be adapted to the quantizer characteristics, and the quantization error will be less than without pitch prediction. Fig. 6.9 shows the SNR values obtained from MPE and RPE analysis procedures, both incorporating a pitch predictor. The pitch predictor coefficients were updated at a rate of 10 ms.

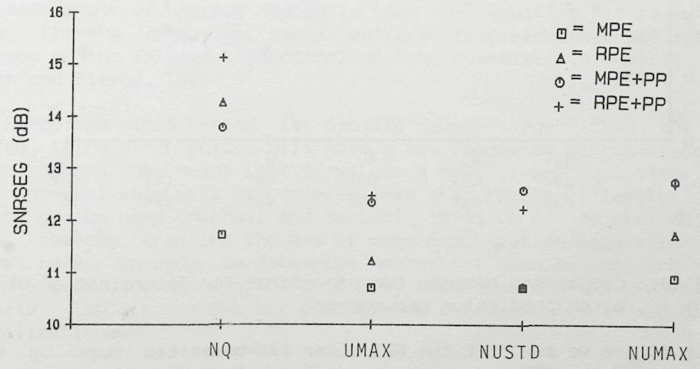


Figure 6.9. Segmental SNR values for quantization with and without pitch prediction.

From this figure we see that the SNR increases for both coders. In Chapters 3 and 4 it was noted that pitch prediction improves the quality of the coders. The quality improvement is higher for the MPE method. In fact the RPE method performs very well without pitch prediction. But if the pulses are quantized, we found that the difference between RPE with and without a pitch predictor was more noticeable. During all these experiments the pitch predictor parameters were not quantized, but β was limited to a value between 0 and 1 , and M was in the range of 16 to 79.

It is remarkable that the differences in SNR between MPE and RPE coders using pitch prediction become less when the excitation signal is quantized. An explanation for this effect is that due to the relatively high amount of pulses within the RPE excitation signal, quantization errors are more likely to occur. A pitch synthesizer uses these quantized pulses to generate the periodic components in the next frame, and as a result large errors may occur, which have to be compensated by the excitation sequence to be found. To prevent such an undesired effect we can determine the pitch predictor parameters from the quantized excitation signal (Section 2.3). The effect of this "internal" computation of the pitch prediction parameters is shown in Fig. 6.10. The parameter M was chosen between 16 and 79 and β was limited to a value between 0 and 1.

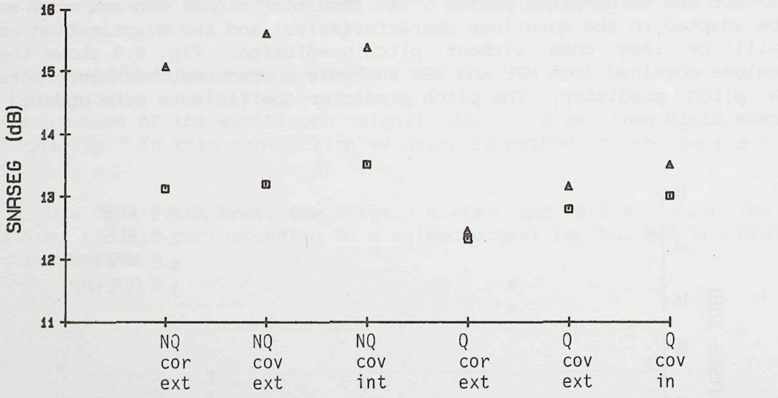


Figure 6.10. Comparison between two procedures for determination of the pitch prediction parameters.

From this figure we see that the RPE coder is benefited more by this procedure than the MPE coder. However, for both coders the sound quality was judged better when compared to synthetic speech obtained with "external" determination of the pitch prediction parameters.

6.2.7.1.1 *Quantization Of The Pitch Predictor Parameters* The pitch predictor parameter set consists of a pitch period M and k predictor coefficients (k=1 or k=3). We demonstrated in Chapters 3 and 4 that a pitch predictor is most effective for high pitched voices (M is low).

By limiting the range of M we can reduce the number of bits required for encoding. A suitable range for M is from 16 to 79 which requires 6 bits for encoding. This range was actually used in all simulations with pitch predictors and did not degrade the performance of the pitch predictors. For a one-tap pitch predictor we quantized the coefficient β uniformly with 4 bits and the range was set between 0 and 1. Listening tests revealed that for these quantizer characteristics no differences could be heard with respect to the unquantized signals. The total set of pitch predictor parameters requires 10 bits. In [Atal, 1982] a method is described for quantizing the gain coefficients of a three-tap pitch predictor. Prior to quantization the coefficients are transformed to reduce the dynamic range. The transformed coefficients are quantized with 13 bits, which for the complete set gives 19 bits.

6.2.7.2 *Entropy Coding* To reduce the bit rate without reducing the number of levels of a quantizer entropy coding can be used. The output of an N-level quantizer is one of the symbols $y_1, y_2, \ldots, y_N$, each having a corresponding probability of occurrence. Instead of transmitting $\log_2 N$ bits/sample, variable length coding such as Huffman coding can be used [Huffman, 1966]. Such a code assigns a word with a high number of bits to a symbol with a low probability and a short word to a symbol with a high probability. The resulting average number of bits/sample approaches the entropy of the quantizer output

$$H = - \sum_{i=1}^{N} p_i \log_2 p_i \tag{6.6}$$

A disadvantage of entropy coding is that the resulting bit rate is variable, thereby requiring large buffers to yield fixed bit rates. If entropy coding is used the optimal uniform quantizer is nearly optimal [Gish and Pierce, 1968].

Since the pitch period is usually much larger than the sample period, large pitch pulses will have a low frequency of occurrence. For voiced speech this means that if we use a multi-level quantizer (n>8), the outer levels will be occupied less than the inner levels. Entropy coding can be used [Makhoul and Berouti, 1979] to reduce the number of bits, thereby enabling the use of more quantization levels for a fixed output rate. In order to determine how useful such an approach will be, we measured the entropy of uniformly quantized MPE and RPE excitation signals. Table 6.3 shows the entropy for different levels and different normalizations.

# levels	entropy MPE (bits)	entropy RPE (bits)
normalization to max		
8	2.82	2.80
9	2.97	2.95
10	3.12	3.11
normalization to rms		
11	2.34	2.65
12	2.35	2.74
13	2.49	2.87
14	2.52	2.94

TABLE 6.3. Entropy of uniformly quantized MPE and RPE signals.

Practical implementations of entropy quantizers will result in somewhat higher entropy values than those listed in Table 6.3. This means that entropy coding is not very suitable for quantizers with normalization to the absolute maximum. For quantizers with normalization to the standard deviation the entropy can be lowered by extending the quantizer range. A range between -4σ and $+4\sigma$ will be large enough to avoid peak clipping. From Table 6.3 we see that a 14-level quantizer with entropy coding requires approximately 3 bits per pulse. Simulations with such a quantizer revealed that there was no improvement over an 8-level quantizer with normalization to the maximum absolute value. From these results we can conclude that no quality improvement can be obtained from the use of quantizers with entropy coding.

6.3 Coding Of The Pulse Positions Of The MPE Excitation

The coding of the pulse positions can be done on a frame base. To introduce no additional delay we can use the same frame as is used for computation of the quantizer gain. For k pulses within a frame of L samples there exist

$$c = \binom{L}{k} = \frac{L!}{(L-k)!k!} \qquad (6.7)$$

possible sets. The minimum number of bits required for coding all these positions is given by

$$c_1 = \log_2 c \quad \text{bits} \qquad (6.8)$$

If the position of a pulse is coded as an offset to the frame boundary we need c_2 bits per frame where c_2 is given by

$$c_2 = k \log_2 L \quad \text{bits} \qquad (6.9)$$

For a frame of size 80 and 8 pulses per frame $c_1 = 34.75$ bits and $c_2 = 50.58$ bits, which is approximately 16 bits more than the theoretical minimum of Eq.(6.8). In the remainder of this chapter we discuss

methods that approximate this theoretical minimum.

6.3.1 *Differential Coding*

If we code the position of the current pulse relative to the position of the previous pulse, and the position of the first pulse relative to the frame boundary we can speak of differential coding. In theory the bit reduction is low, because the maximum number of bits required for the coding of the differences is then

$$c_3 = k \log_2(L-k) \qquad (6.10)$$

In practical situations the reduction can be more effective. The histogram of Fig. 6.11 shows the average number of zeros between the pulses within a frame obtained from a large database. Each frame of 80 samples contained 8 pulses.

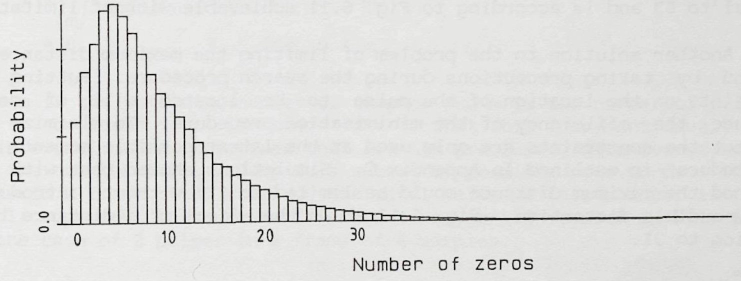


Figure 6.11. Probability of zero-string lengths.

From this figure we see that a distance greater than 64 samples never occurs. If we do not want to use more than 5 bits for each differential position, we can code distances up to a length of 31. However, the probability of finding distances greater than 31 is small but not zero. One solution for this problem is to allow only odd or even positions of the pulses. Thus pulses are, for example, only allowed on positions 1,3,5 etc. This operation also reduces the complexity of the search procedure. Listening tests showed that the degradation by this limitation was very small. Fig. 6.12 shows the effect of such a limitation on the segmental SNR.

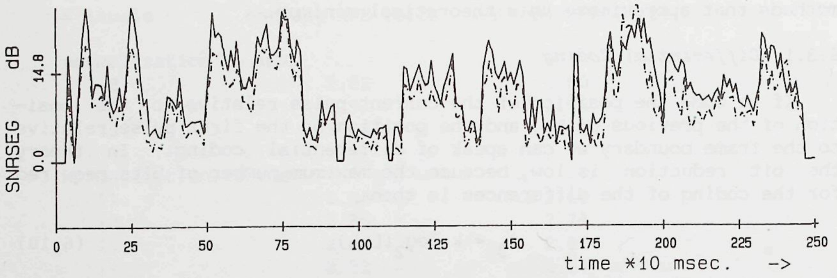


Figure 6.12. Segmental SNR of MPE coder with (dashed line) and without (solid line) limitation on the possible pulse locations.

The effective maximum distance that can now be coded with 5 bits is equal to 63 and is according to Fig. 6.11 achievable without limitation.

Another solution to the problem of limiting the maximum distance was found by taking precautions during the search procedure. Putting constraints on the location of the pulse to be located will of course reduce the efficiency of the minimization procedure. To minimize this effect the constraints are only used at the latest possible moment. The procedure is outlined in Appendix C. Simulations showed that with this method the maximum distance could be limited to 21 without introducing any audible distortion. Fig. 6.13 shows the effect of a distance limitation to 31.

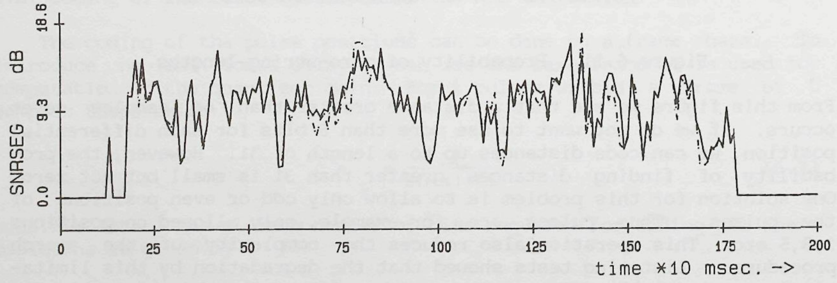


Figure 6.13. Segmental SNR of MPE coder with (dashed line) and without (solid line) distance limitation.

From these figures and listening tests it was clear that the distance limitation procedure performs better than the one using a sub-set of possible pulse positions.

6.3.2 *Enumerative Coding*

In [Schalkwijk, 1972] [Cover, 1973] a coding algorithm was proposed which can be used for coding the pulse positions. All possible position sets [Eq.(6.7)] can be placed on an imaginary list. To code a set of positions, the list is searched for the corresponding entry and the index is transmitted. At the receiver this index is used to select the appropriate position set. This procedure is optimal if the total number of position sets is equal to a power of two and all pulse locations have the same probability. The encoding can be accomplished by traversing a binary tree in which a "1" branch indicates the occurrence of a pulse and a "0" branch indicates that there is no pulse at the current position. Whenever a "1" is encountered, a counter is incremented, whereby the final value of this counter is the code word to be sent. The increment is given by:

$$I = \binom{n}{m} = \frac{n!}{m!(n-m)!} \quad \text{for } n \geq m \qquad (6.11)$$
$$= 0 \quad \text{for } n < m$$

where n is the remaining number of samples and m is the number of pulses yet to be found + 1. The basic idea is that this increment equals all possible excitation structures that are now precluded; with every increment the counter makes a jump to a subset of possible code words. For 8 pulses in a frame of 80 samples, the final value of this counter is the 35 bit code word to be sent. This procedure is illustrated in Fig. 6.14 for the case of 2 pulses in a frame of 4 samples.

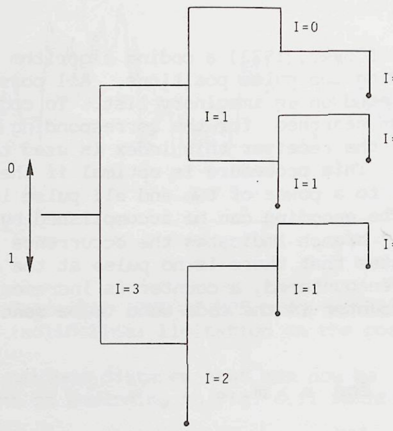


Figure 6.14. Example of enumerative position coding of the sequence 1010.

Thus, suppose we have the sequence 1010. The occurrence of a "1" in the first sample excludes the three excitation sequences which start with a zero, namely 0011, 0101 and 0110. Hence an increment of I=3 is obtained. The second sample is zero and therefore no increment is required. The third sample is a "1" and hence an additional increment equal to I=1 results. All positions are now detected and the resulting code word C is equal to 4. A possible coding algorithm is given by:

```
Initialization:
        L     = size of excitation frame
        k     = number of pulses
        IP(n) = pulse positions

Coding loop:
        m := k; C :=0;
        for n = 1 to k do
        begin
           i :=   L-IP(n)
           C := C + $\binom{i}{m}$
           m := m - 1
        end
```

Figure 6.15. Algorithm 1: Position coding.

The decoding procedure is complementary to the encoding procedure. For every possible position we check whether the corresponding increment is

comprehended in the code word. If this test is positive we subtract this increment factor from the code word and repeat the testing for the remaining positions. Using the same example as before, we will decode the code word C=4. For the first sample, a test is needed to find out whether the code word is greater than or equal to the increment I=3 due to a "1" on the first position. This test is true, which means that the first pulse is located on the first position, and the increment factor I is subtracted from the code word C (C=4-3=1). For the second position we check whether the residual code word is greater than or equal to the increment I=2. This is not true, which means that the second location contains a zero. In the third test a "1" is decoded and C is decremented with I=1 (C=1-1=0). At the receiver a final zero can be allocated since all k=2 pulses have been decoded. An algorithm for the decoding procedure is given by:

```
Initialization:
        C    = code word
        L    = size of excitation frame
        k    = number of pulses
        v(n) = array with pulse markers

Decoding loop:
        m := k; i := L;
        for n = 1 to L do
        begin
           i := i - 1
           if C $\geq \binom{i}{m}$ then
           begin
              v(n) := 1
              C := C - $\binom{i}{m}$
              m := m - 1
           end
           else v(n) := 0
        end
```

Figure 6.16. Algorithm 2: Position decoding.

Using the enumeration procedure each possible excitation string will be assigned an unique code word C with a value between 0 and c-1, where c was defined in Eq. 6.7. A proof of the validity of this procedure can be found in [Schalkwijk, 1972].

6.3.3 *Complexity Of The Coding Algorithms*

The complexity of the differential coding algorithm is not high. The use of the distance limiting procedure results in an additional complexity of the MPE analysis but is partly compensated by the reduction in size of the search frame. The encoding procedure is further straightforward.

The enumerative encoding procedure does not increase the complexity of the MPE analysis but the encoding procedure is more complex than the differential encoding procedure. In an excitation sequence of size L with k pulses there are (k-1)(L-k+1) combinatorial terms which have to be stored. For k=8 and L=80, 511 combinatorial terms consisting of 35 bits each have to be stored. At the encoder for every pulse, a memory reference and an addition of a combinatorial term have to be made.

At the receiver, L comparisons have to be made between the code word and a combinatorial term which has to be retrieved from memory.

If the memory space available is limited, the required storage space can be reduced by an on-line computation of the combinatorial terms by making use of the recursive relation:

$$\binom{n}{m}=\binom{n-1}{m}+\binom{n-1}{m-1} \qquad n \geq m \geq 1 \tag{6.12}$$

This recursion can be repeated until combinatorial terms are found in appropriate forms such as:

$$\binom{n}{n}= 1 \quad \text{or} \quad \binom{n}{1}= n \tag{6.13}$$

An addition of all these values gives the value of the desired combinatorial term.

6.3.4 *Combined Encoding Of Position And Amplitude*

For the encoding of one single pulse we need for an 8-level quantizer 3 bits for the amplitude and approximately 5 bits for the position, resulting in 8 bits/pulse. If the number of bits required for the position and amplitude coding are not integers we can save bits by the combined encoding of amplitude and position.

6.3.4.1 *Enumerative Coding Of Positions And Amplitudes* For an excitation sequence of size L with k pulses, each having one out of N possible amplitude values, the number of possible combinations is given by:

$$\binom{L}{k}N^k \tag{6.14}$$

Similar to the enumerative coding of the positions, indices can be used

to represent these possibilities. With the index as code word and assuming that all possibilities have an equal probability of occurrence, the minimum number of bits needed to represent an excitation sequence is given by:

$$H = \log_2\binom{L}{k}N^k = \log_2\binom{L}{k}+ k \log_2 N \quad \text{bits} \tag{6.15}$$

It can be seen that both the pulse positions (first term) and the amplitudes (second term) can then be coded optimally. For 8 pulses in a frame of 80 samples and a 6 level quantizer the entropy equals H = 55.39 bits, so that every frame can be encoded with 56 bits. The position encoding procedure of Section 6.3.2 can be modified to incorporate the amplitude encoding. For this purpose the "1"'s in the binary tree are replaced by the different amplitude values, and the increment then depends on the selected amplitude value. Having N amplitudes a(0), a(1), ..., a(N-1) the increment can be represented by:

$$I =\binom{n}{m}N^m + j\binom{n}{m-1}N^{m-1} \quad \text{, where} \tag{6.16}$$

n = remaining number of samples
m = number of pulses yet to be found + 1
j = index of amplitude a(j)
N = number of amplitude values

The first term in Eq.(6.16) represents the number of possible excitation sequences having a zero at that position. The second term gives an additional offset, which depends on the amplitude: if amplitude a(j) is detected, the counter is incremented with the number of branches connected to the nodes a(0) to a(j-1). In Fig. 6.17 this procedure is illustrated for 2 pulses in a frame of 4 samples. The pulses can have either amplitude a(0) or a(1).

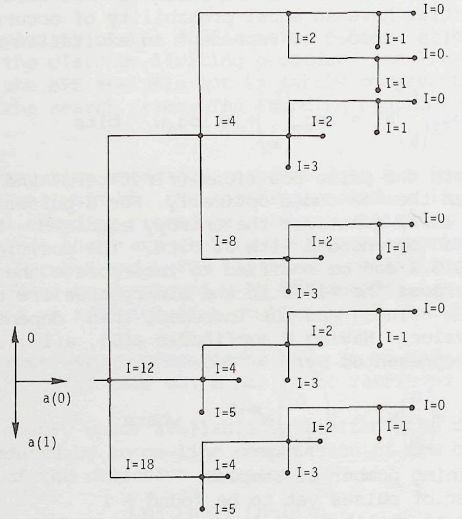


Figure 6.17. Example of enumerative encoding procedure for position and amplitude.

The coding procedure is given by the next algorithm:

Initialization:

L	=	size of excitation frame
k	=	number of pulses
IP(n)	=	pulse positions
b(n)	=	pulse amplitudes
N	=	number of amplitude values

```
Coding loop:
   m := k; C :=0;
   for n = 1 to k do
     begin
       i:= N-IP(n)
       for j = 0 to N-1 do
         begin
           if b(n) = a(j) then C := C + $\binom{i}{m}N^{m} + j\binom{i}{m-1}N^{m-1}$
         end
       m := m - 1
     end
```

Figure 6.18. Algorithm 3: Pulse position and amplitude coding.

The complementary decoding procedure is similar to algorithm 1 where only positions were detected; first a test is necessary to determine whether a zero or a pulse is to be decoded. Then additional tests are required to find out what particular amplitude has to be selected. This procedure is given by the following algorithm:

Initialization:

C	=	code word
L	=	size of excitation frame
k	=	number of pulses
v(n)	=	excitation array
N	=	number of amplitude values

```
Decoding loop:
     m := k; i := L;
     for n = 1 to L do
       begin
         i := i - 1
         if $C \geq \binom{i}{m} N^{m}$ then
           begin
             $C := C - \binom{i}{m} N^{m}$
             found := false
             for j = 0 to N-1 while found = false do
               begin
                 if $C \geq j \binom{i}{m-1} N^{m-1}$ then
                   begin
                     v(n)  := a(j)
                     C     := $C - j \binom{i}{m-1} N^{m-1}$
                     m     := m - 1
                     found := true
                   end
               end
           end
           else v(n):=0
         end
```

Figure 6.19. Algorithm 4: Pulse amplitude and position decoding.

6.3.4.2 *Differential Coding With Amplitude Coding* With a combined encoding of differential position and amplitude we get the so-called variable-length-to-block code, a generalized run-length code [Golomb, 1966], which was extensively described in [Jelinek and Schneider, 1972]. The described algorithm gives a set of source words consisting of a string of zeros followed by an amplitude value that need not be different from zero. Each source word is numbered and the number, in binary representation, is used as code word.

As an example of this algorithm we will construct a set of 9 source words for the amplitude values -1,0 and +1, and we assume that the probabilities P(.) of these variables are P(0)=0.9 and P(+1)=P(-1)=0.05. Let T denote the number of source words, W(T) the set of source words, w a source word of W(T), c the number of amplitude values and J the set of amplitudes values J={-1,0,+1}.

The first step is to set T=c and W(T)=J. Then choose that w of W(T) having the greatest probability, and make new source words by extending w

with the words of J. Combine the new words with the words of W(T), thereby excluding w. Next a new set of size T=c+1(c-1) has been formed:

W(T)={-1,+1,0-1,0+1,00},

T=c+(c-1)=5.

The same procedure applied to W(5) gives:

W(T)={-1,+1,0-1,0+1,00-1,00+1,000},

T=c+2(c-1)=7.

W(9) is found by using W(7) as input and gives:

W(T)={-1,+1,0-1,0+1,00-1,00+1,000-1,000+1,0000},

T=c+3(c-1)=9.

It can be shown [Jelinek and Schneider, 1972] that T=c+n(c-1), where n denotes the number of extensions. The value n actually represents the maximum number of zeros preceding a pulse. The procedure above can be repeated until an appropriate value of T has been reached. Using b bits per pulse, this value is limited by

$$T = c + n(c-1) < 2^{b} \qquad (6.17)$$

The value of c is equal to the number of different amplitudes within an excitation sequence. This means that for symmetrical quantizers with 2n and 2n+1 (n=1,2,..) levels the value of c for a particular n remains the same. Hence, using this code, a 7 level quantizer gives the same bit rate as a 6 level quantizer. In Table 6.4 the number of source words T and the maximum length n of the string of zeros are listed as a function of b and the number of quantizer levels N.

	N=6,7			N=8,9		
bits b :	6	7	8	6	7	8
T :	61	127	253	57	121	249
n :	9	20	41	6	14	30

TABLE 6.4. T and n as functions of the number of bits b and the number of quantizer levels l.

The code words representing the source words can be found by ordering the source words and using the index as code word. If the source words are ordered according to an increasing number of zeros in front of the non-zero amplitude values, then any code word C is given by:

$$C = Z*N + j \quad , \text{where} \qquad (6.18)$$

Z = number of zeros
j = index of amplitude a(j)
N = number of non-zero amplitude values

The encoder counts the number of zeros between the pulses and whenever a non-zero amplitude value is detected, a code word is computed using Eq.(6.18). In case a maximum number of zeros is not followed by an non-zero amplitude, a code word C is computed from:

$$C = (NZM + 1)*N \qquad (6.19)$$

where NZM = maximum number of zeros (n in Table 6.4). For 7 bits/pulse this happens whenever 21 successive zeros occur.

The decoding procedure could be accomplished using two memory references: the code word extended with a logical 0, being the address of the number of zeros to be outputted,and the code word extended with a logical 1, the address of the pulse amplitude. At the receiver the coding table must be stored as a two-dimensional table, representing the number of zeros and the amplitude. In Table 6.5 this is illustrated using the source words obtained in the previous example.

source word	code-word	number of zeros	ampli-tude
-1	000	0	-1
+1	001	0	+1
0-1	010	1	-1
0+1	011	1	+1
00-1	100	2	-1
00+1	101	2	+1
000	111	3	0

TABLE 6.5. Example of coding and decoding procedure.

6.3.5 *Computational Complexity*

If the pulse amplitude value and pulse position are jointly coded with the enumerative encoding procedure, two memory references to combinatorial terms have to be made, after which the increment can be found by three multiplications and an addition. This increment has to be added to the increment already found.

At the receiver, L comparisons have to be made between the code word and a combinatorial term multiplied by a factor which depends on the value of N and the number of pulses which are yet to be found. If a pulse is detected then another memory reference to a combinatorial term is necessary. For amplitude decoding, a multiplication and several additions are required. The total number of additions will depend on the actual amplitude and will be L in the worst case.

For the variable-length-to-block code no terms have to be stored at the encoder since all code words can be computed from the multi-pulse signal. At the encoder it is first necessary to test whether a sample is a pulse or a zero. If it is a non-zero amplitude value, a multiplication followed by an addition gives the code word.

At the receiver two memory references per pulse have to be made; the first gives its number of preceding zeros,and the second the amplitude value. The decoded number of zeros is outputted and followed by a pulse of the decoded amplitude.

6.4 Quantizing And Encoding The Filter Parameters

The p-th order all-pole synthesis filter 1/A(z) can be uniquely specified by different parameter sets [Rabiner and Schafer, 1978, p. 441]. For example, the auto-correlation coefficients $\{c(k)\}$, the filter coefficients $\{a_k\}$, the reflection coefficients $\{\rho_k\}$ and the poles of 1/A(z) all specify the same transfer function. In selecting a parameter set suitable for quantization we require 1) filter stability upon quantization and 2) a natural ordering of the parameters. The second requirement is not mandatory but enables the use of parameter statistics for coder design. Reflection coefficients or PARCOR's (PARtial CORrelation coefficients) satisfy both requirements. In [Viswanathan and Makhoul, 1975] a comparison was made between various filter parameter sets and it was shown that the reflection coefficients are the most effective parameters to be used for transmission. Another coefficient set that also possesses the required properties is called the Line Spectral Pair (LSP) representation [Soong and Juang, 1984] [Wakita, 1981]. This coding scheme transforms the set of p poles inside the unit circle into a set of p poles on the unit circle. The new poles then become automatically ordered by frequency.

The synthesis filter coefficients parametrize the spectral envelope of the speech signal and for a good quality it is necessary to maintain the shape of this envelope upon quantization. A distance between the log power spectra of the unquantized and the quantized filters can be used as a measure for the spectral match. Let the spectrum of the all-pole model be represented by $P(\omega) = |1/A(e^{j\omega})|$ and the spectrum of the quantized filter by $\hat{P}(\omega)$, where ω is the normalized frequency. The log spectral difference between $P(\omega)$ and $\hat{P}(\omega)$ is defined as

$$\Delta V(\omega) = \ln [P(\omega) - \ln [\hat{P}(\omega)] \qquad (6.20)$$

The distance d between the log spectra can be defined in terms of an L_q norm , i.e.,

$$d = [\int_{-\pi}^{\pi} |\Delta V(\omega)|^q \, d\frac{\omega}{2\pi}]^{\frac{1}{q}} \qquad (6.21)$$

where q is a positive integer. For the L_1 norm, d represents the average of the absolute value of the difference (spectral deviation) between the two log spectra $P(\omega)$ and $P(\omega)$. For $q=2$, d represents the root mean square (rms) value of the spectral deviation.

Quantization changes the values of the reflection coefficients. To investigate the effect of these perturbations, it is necessary to determine the sensitivity of the all-pole model spectrum to small changes in the values of the reflection coefficients. The spectral sensitivity for the PARCOR coefficient $\rho(i)$ is defined as

$$\frac{\partial S}{\partial \rho(i)} = \lim_{\Delta\rho(i)\to 0} \left|\frac{\Delta S}{\Delta \rho(i)}\right| \qquad (6.22)$$

where ΔS is the deviation in the all-pole spectrum due to a change $\Delta\rho(i)$ in the coefficient $\rho(i)$. As a measure of spectral deviation any of the L_q norms can be used. The sensitivity can be determined experimentally for the L_1 norm [Viswanathan and Makhoul, 1975], or analytically for the L_2 norm [Gray and Markel, 1976]. It appears that the sensitivity is high for reflection coefficients with a magnitude close to one and that the corresponding sensitivity curve is U shaped. Due to this non-flat sensitivity curve, linear quantization of the PARCOR parameters is not optimal. One approach to overcoming this difficulty is to reduce the sensitivity by pre-emphasizing and windowing the data prior to spectral analysis [Tohkura and Itakura, 1979]. Another possibility is to apply a non-linear transformation to the reflection coefficients, such that the transformed parameters have a flat or constant spectral sensitivity behavior. Two proposed transformations are the log area transformation [Viswanathan and Makhoul, 1975],

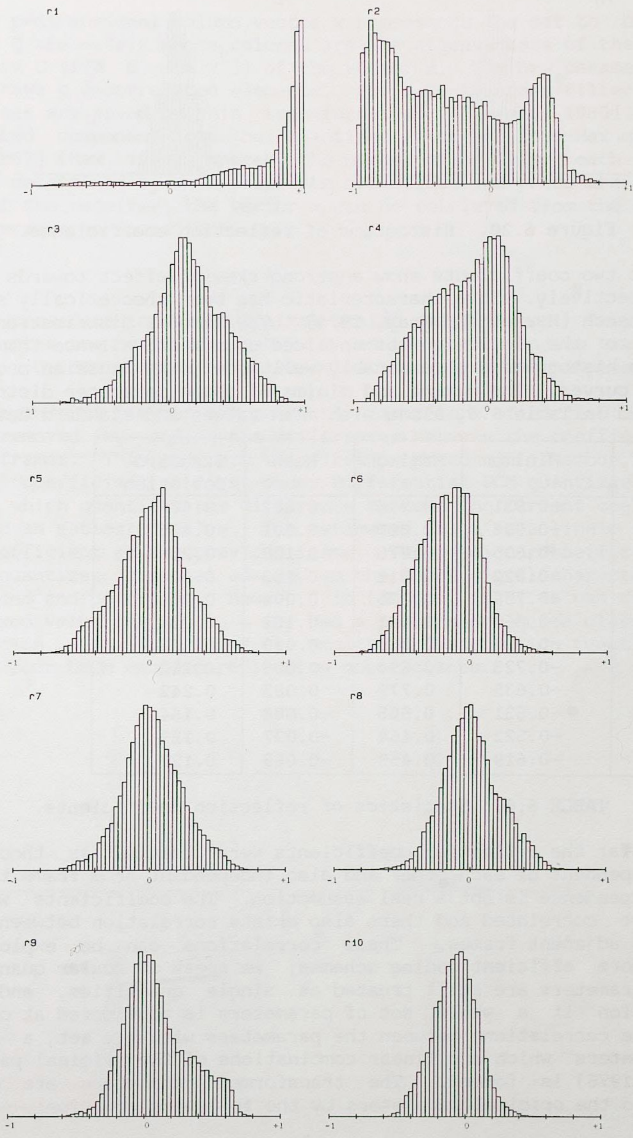
$$g(i) = \log\left[\frac{1+\rho(i)}{1-\rho(i)}\right] \qquad (6.23)$$

that yields a set of LAR parameters, and the inverse sine transformation [Gray and Markel, 1976] [Gray et al., 1977]

$$\theta(i) = \sin^{-1}[\rho(i)] \qquad (6.24)$$

that yields a set of theta parameters. The transformed parameters are then quantized with a uniform or non-uniform quantizer. The quantizer features such as number of levels per coefficient and the input range are determined by optimization of an appropriate design criterion, such as minimizing the maximum spectral deviation [Viswanathan and Makhoul, 1975] [Gray and Markel, 1976] [Gray et al., 1977] or minimizing the expected value or mean of the spectral deviation [Gray et al., 1977]. In [Gray and Markel, 1980] the quantizer design procedures for both criteria were described.

The distributions of the reflection coefficients are shown in the histograms of Fig. 6.20. These histograms show the amplitude distributions of the different reflection coefficients obtained from speech samples in a representative database. The coefficients were obtained with the auto-correlation method with a 25 ms Hamming window and update rates of 10,15 and 20 ms.



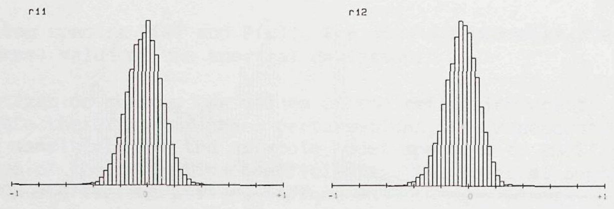


Figure 6.20. Histograms of reflection coefficients.

The first two coefficients show a strong skewing effect towards +1 and -1, respectively. This characteristic has been theoretically shown for voiced speech [Markel and Gray, 1974]. The bimodal distribution of $\rho(2)$ is due to the inclusion of unvoiced sounds and silence frames. The remaining histograms are reasonably well modeled by Gaussian probability density curves. The maxima and minima of these parameter distributions are listed in Table 6.6, along with mean values and standard deviations.

Coef.	Minimum	Maximum	Mean	Standard Deviation
1	-0.931	0.996	0.682	0.407
2	-0.996	0.800	-0.301	0.465
3	-0.805	0.876	0.128	0.286
4	-0.922	0.811	-0.163	0.308
5	-0.756	0.770	0.004	0.254
6	-0.869	0.591	-0.182	0.221
7	-0.762	0.762	0.010	0.223
8	-0.723	0.694	-0.003	0.212
9	-0.635	0.772	0.083	0.242
10	-0.631	0.505	-0.088	0.164
11	-0.522	0.464	-0.037	0.129
12	-0.619	0.459	-0.069	0.123

TABLE 6.6. Statistics of reflection coefficients.

Thus far the reflection coefficients were treated as though they were independent of each other and also independent from frame to frame. This independence is not a real assumption. The coefficients within a frame are correlated and there also exists correlation between coefficients of adjacent frames. These correlations can be exploited to obtain more efficient coding schemes. We speak of scalar quantization if the parameters are still treated as single quantities, and vector quantization if a whole set of parameters is considered at once. To remove the correlations between the parameters within a set, a new set of parameters which are linear combinations of the original parameters [Segall, 1976] is formed. The transformed parameters are directly related to the original parameters by the following equation:

$$y = Q^t x \qquad (6.25)$$

where the p-dimensional column vector x represents the set to be quantized and Q the matrix whose columns are the eigenvectors of the covariance matrix C (i.e. $C = [x\, x^t]$) of the vector x. The new parameter vector will have p uncorrelated elements. For a 10-th order filter about 3 bits per set are saved by this transformation [Fussel, 1980]. Every decorrelated component can be quantized with a Lloyd-Max quantizer [Lloyd, 1957] [Max, 1960], whereby the number of bits for each coefficient is determined by an appropriate bit allocation procedure [Segall, 1976]. At the receiver, the vector x can be retrieved from the vector y by an inverse transformation

$$\hat{x} = Q\,\hat{y} \qquad (6.26)$$

The whole procedure is called optimal scalar quantization. In practice, one estimates the covariance matrix C and the corresponding *transformation* matrix Q from a representative collection of speech samples.

The dependence between the predictor parameters of succeeding frames can be removed by coding the differences between the coefficients of adjacent frames. This approach was first proposed in [Sambur, 1975]. For each coefficient a first-order Differential PCM quantizer is constructed, which quantizes the difference between the current coefficient value and an estimated value. The estimated value is a linear function of the coefficient of the previous frame. The predictor coefficient a_1 of the quantizer can be adapted to the input signal characteristics (e.g. voiced and unvoiced). However, in [Sambur, 1975] it was concluded that a fixed value for a_1 ($a_1 = -1$) had a performance quite close to the adaptive case. Fig. 6.21 shows the amplitude distribution functions of the first four DPCM encoded reflection coefficients.

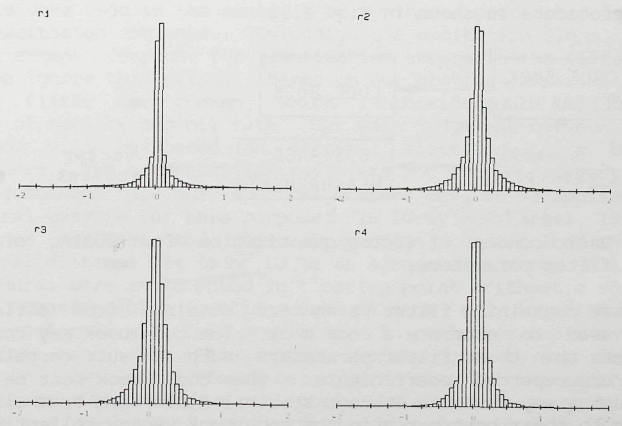


Figure 6.21. Histograms of DPCM encoded reflection coefficients.

A fixed predictor with $a_1=-1.0$ was used. The maxima and minima of these distributions are listed in Table 6.7 along with mean values and standard deviations.

Coef.	Minimum	Maximum	Mean	Standard Deviation
1	-1.713	1.625	0.002	0.234
2	-1.482	1.403	0.002	0.253
3	-1.211	1.317	0.000	0.222
4	-1.242	1.253	0.000	0.202
5	-1.181	1.114	0.000	0.194
6	-0.945	0.937	0.000	0.179
7	-0.834	1.020	0.000	0.169
8	-0.887	0.916	0.000	0.165
9	-0.856	0.884	0.000	0.159
10	-0.703	0.591	0.000	0.142
11	-0.572	0.662	0.000	0.128
12	-0.603	0.658	0.000	0.124

TABLE 6.7. Statistics of DPCM encoded reflection coefficients.

The amplitude distribution histograms for the differential values are easily modeled by a gamma density for the first coefficient and Laplace densities for the higher order coefficients. For these distribution functions optimal quantizer characteristics are available [Paez and Glisson, 1972].

Application of vector quantization (VQ) to the coding of LPC filter parameters was first described in [Buzo et al., 1980]. The basic concept of this procedure is shown in Fig. 6.22

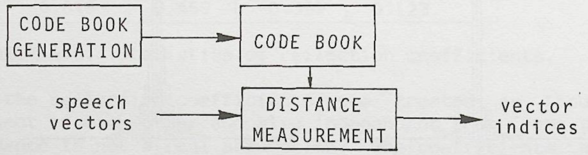


Figure 6.22. Basic concept of vector quantization for coding of the filter parameters.

A large database containing filter parameters obtained from different speakers is used to generate a code book. The code book may contain other parameters than these filter parameters, such as auto-correlation coefficients or cepstral coefficients. The code book must be constructed in such a way that the average spectral distortion from all the input vectors to their best match in the code book vector collection is minimized (see also Appendix D). The degree of matching is measured by an appropriate distortion measure such as the Itakura measure or the

likelihood ratio [Gray et al., 1980]. The quantizer uses the same distortion measure as was used during the design of the code book, to select the code book vector that has the best match to the input vector. The corresponding code book index is used as the transmission or storage parameter. An objective and subjective evaluation of scalar and vector quantization procedures for reflection coefficients can be found in [Juang et al., 1982]. The scalar method was based on the minimum deviation method [Gray et al., 1977] [Gray and Markel, 1980]. There it was stated that 10 bit vector quantization had a performance similar to 24 bit scalar quantization. In [Roucos et al., 1982] a similar comparison between VQ and optimal scalar quantization was made, and it was concluded that the deviations were less dramatic. Vector quantization with 10 bits/set produced the same mean square error as 15 bits/set optimal scalar quantization. Despite the savings in bits, two drawbacks exist for the application of vector quantization. First the enormous complexity of the search procedure and second the performance on speech samples not used for code book generation. The complexity can be reduced by using sub-optimal search procedures (e.g. tree-search) at the expense of a slight decrease in performance [Wong et al., 1982]. The effects of decreased performance for outside training data can only be minimized by using a very large training set that requires enormous computational effort.

6.4.1 *Quantization Of The Filter Parameters For The RPE And MPE Coders*

To investigate the effect of filter parameter quantization on the synthetic speech quality of MPE and RPE coders, we performed computer simulations with different quantization procedures. In these experiments, the excitation signals were not quantized. After LPC analysis, the reflection coefficients were quantized and these quantized coefficients were used in the analysis by synthesis procedures for determining the excitation sequence. Obviously, the excitation signal can to a certain extent correct for quantization errors in the filter parameters, but we ignore this effect. Based on our previous experiments, a 12-th order filter was chosen, which is considered to be a good choice in terms of quality and bit rate. Our main judgement between the different methods will be based on informal listening tests by experienced listeners. The segmental SNR ratio was not considered very useful for the judgement of the spectral differences, so we applied the rms log spectral measure for this purpose. In [Gray and Markel, 1976] different distance measures and their properties are described and the log rms spectral distance was found to be an adequate measure. All quantization procedures were implemented in floating point arithmetic so that coefficient transformation functions such as sine and logarithmic functions were performed with full precision. In real-time implementations these functions have to be approximated, thereby possibly introducing degradation in quantizer performance; however some initial experiments showed that these effects can be neglected. As a reference quantization procedure we took the quantization scheme described in [Sluyter et al., 1984], and listed in Table 6.8.

Coef.	Minimum	Maximum	Number of bits
1,2	-0.999	0.999	6
3	-0.580	0.850	5
4	-0.850	0.580	5
5,6,7,8,9,10	-0.700	0.650	4
11,12	-0.400	0.300	3

TABLE 6.8. Reference quantization procedure (RP).

The total number of bits for a twelfth-order filter is 52, resulting in a bit rate of 2600 b/s for a 20 ms update rate. A comparison of the RP scheme to other quantization schemes found in literature [Tremain, 1982] [Atal, 1982] revealed that this number of bits is amply sufficient. In our experiments we investigated other quantization procedures with a performance equal to the RP scheme but that requiring fewer bits.

In [Gray and Markel, 1980] two procedures for designing scalar quantizers were described. Both methods minimize the spectral deviation or L_q distance between the unquantized and quantized log spectra [Eq.(6.21)]. The first method attempts to minimize the maximum spectral deviation (D_m) and is called Uniform Sensitivity (US) quantization. The second method attempts to minimize the expected value or mean (D_a) of the spectral deviation and is called Minimum Deviation (MD) quantization. Both minimizations are done over all frames of speech in a representative database. The derivative of the spectral deviation with respect to a single parameter has a magnitude which has been called the sensitivity for that parameter [Eq.(6.22)]. The sensitivity curve of the parameters is made uniform by transforming the reflection coefficients to inverse sine coefficients [Eq.(6.24)]. The sensitivity of the resulting theta coefficients is determined empirically from a large data base. Fig. 6.23 shows the evaluated mean sensitivity, the maximum sensitivity and the mean plus two standard deviation sensitivity.

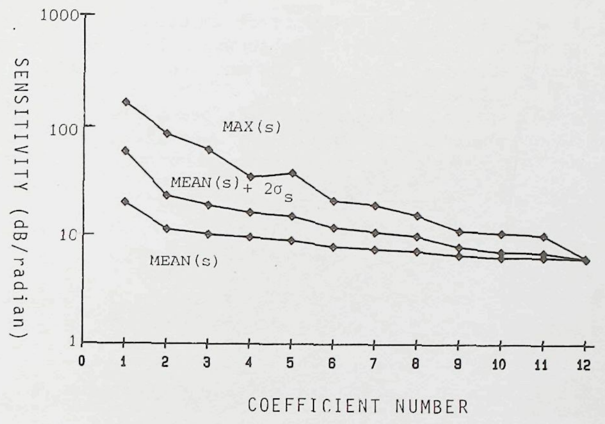


Figure 6.23. Theta coefficient sensitivity.

The reflection coefficients were obtained from 12-th order autocorrelation analysis on 25 ms frames with a Hamming window, and update rates of 10, 15 and 20 ms. To obtain a good estimation of the sensitivity, the mean sensitivity plus two times the standard deviation sensitivity was used. The number of levels assigned to each theta coefficient is a function of the sensitivity of that coefficient and the coefficient statistics. For uniform sensitivity quantization, the theta coefficients are uniformly quantized over the range of all values from the absolute minimum to the absolute maximum and parameter statistics are ignored. In [Gray and Markel, 1980] it was suggested that a more reasonable approach is to choose statistically meaningful extremes such as two standard deviations from the mean value. For the Minimum Deviation quantization, the theta coefficients are *non-uniformly* quantized and the amplitude probability function is explicitly used. To design US and MD quantizers, the deviation (D_m for US and D_a for MD) is specified and the design procedures minimize the number of levels/coefficient for the given deviation. To obtain a specified number of bits/set the design deviation has to be adjusted until the required number of bits is obtained.

For the design of optimal scalar quantizers we specify the total number of bits per frame, and distribute the available bits over the different parameters. The bit allocation procedure is based on the assumption that each parameter has a uniform sensitivity and a Gaussian amplitude distribution (which is not true for the first two reflection coefficients). Once the number of bits for each parameter is known, optimal uniform or non-uniform quantizers can be designed.

We used the design procedures described to obtain different quantizer characteristics for the encoding of 12 inverse sine coefficients with 44 bits per set. For the DPCM quantizer we used 33 bits per set. The bit allocation for each quantizer is shown in Table 6.9. The spectral deviation per coefficient (D), was .68 and .40, respectively, for the US and the MD quantizer design procedure. The optimum scalar quantizers (OS) were implemented without parameter decorrelation. The suffices u and nu denote uniform and non-uniform quantizers, respectively.

coef.	1	2	3	4	5	6	7	8	9	10	11	12	SUM
RP-u	6	6	5	5	4	4	4	4	4	4	3	3	52
US-u	7	6	5	4	4	3	3	3	3	4	4	4	44
MD-nu	7	5	5	4	4	4	3	3	3	2	2	2	44
OS-u	5	5	4	4	4	3	3	3	4	3	3	3	44
OS-nu	5	5	4	4	4	3	3	3	4	3	3	3	44
DPCM-nu	5	5	4	4	3	3	2	2	2	1	1	1	33

TABLE 6.9. Bit allocation for different quantizers.

Fig. 6.24 shows the log spectral distance for the different methods as a function of time for both a female and a male speaker.

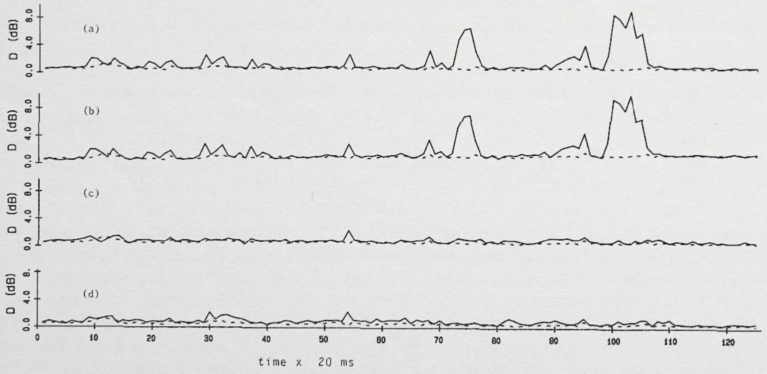


Figure 6.24. Log spectral distance as a function of time for US (a), OS-u (b), MD (c), OS-nu (d) and RP (dashed line) quantizers for a female speaker.

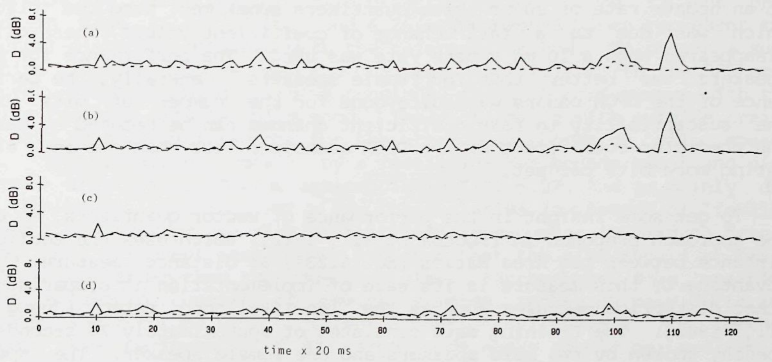


Figure 6.25. Log spectral distance as a function of time for US (a), OS-u (b), MD (c), OS-nu (d) and RP (dashed line) quantizers for a male speaker.

The update rate of the coefficients is 20 ms and the distance is computed for every set of coefficients. In each figure the RP results are given as a reference (dashed line). From this figure we see that US and OS-u produce similar results, which was expected because both minimize the maximum spectral deviation. The same is true for MD and the OS-nu quantizers. In listening tests the quantizers that produce a constant distance such as the RP, MD and OS-nu quantizers were preferred over the US and OS-u quantizers. For the female speaker the large deviations in spectral distance for the US and OS-u quantizers were clearly perceived as distortions. For male speakers the differences between the various quantizers were less noticeable. Informal listening tests revealed that the performance of the MD quantizer was equal to that of the RP quantizer.

The spectral distance function of the DPCM coder is shown in Fig. 6.26 for 10 and 20 ms update rates.

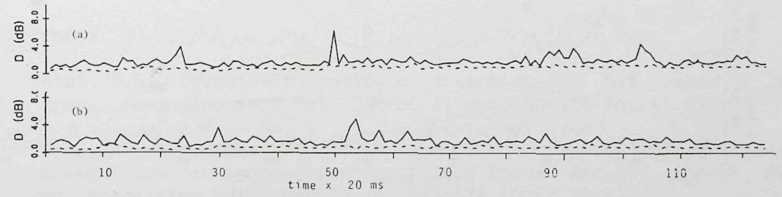


Figure 6.26. Log spectral distance as function of time for the DPCM coder at a 10 ms (a) and a 20 ms (b) update rate.

At an update rate of 20 ms these quantizers sometimes produced clicks, which was due to a fast change of coefficient values. These clicks disappeared when a 10 ms update rate was used. The performance for male speakers was better than for female speakers. Generally, the performance of the DPCM coders was quite good for the number of bits used. The susceptibility to fast coefficient changes can be reduced by making the predictor coefficient of the DPCM quantizer less than one and allocating more bits per set.

To get some insight in the performance of vector quantizers, we used the approach proposed in [Roucos et al., 1982], which uses the Euclidian distance between Log Area Ratios [Eq.(6.23)] as distance measure. The advantage of this measure is its ease of implementation in comparison to other distortion measures such as the log-likelihood ratio [Juang et al., 1982]. The training data consisted of approximately 70 seconds of speech, spoken by two male speakers and one female speaker. The speech material was analyzed with the auto-correlation method using 25 ms Hamming windows and an update rate of 10 ms. The resulting 7000 sets of 12 LAR coefficients each, were used to generate a 10-bit code book (1024 vectors). The binary split algorithm was used with a perturbation factor δ=0.01 and a convergence threshold ϵ=0.005 (see also Appendix D). Two utterances spoken by the same speakers as used for the training data were used for evaluation. One of the utterances was actually used for training (inside training data), while the other was not (outside training data). The reflection coefficients were quantized using a full search through the code book and the LAR distance measure. Fig. 6.26 shows the spectral distance function for inside and outside training data for both a male and a female speaker.

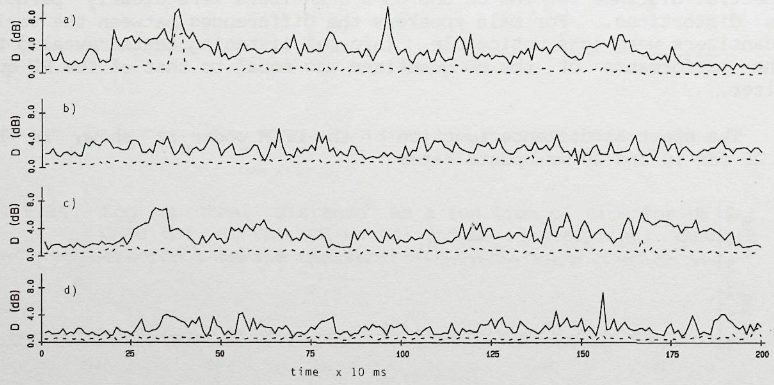


Figure 6.27. Log spectral distance as a function of time for a female speaker (a,b) and a male speaker (c,d), with outside training data (a,c) and inside training data (b,d).

Note that the inside training data utterances correspond to the ones used in Figs. 6.24, 6.25, and 6.26. The dashed lines represent the spectral distances obtained with the reference method. To obtain a subjective impression, we used the quantized coefficients in the RPE procedure. From listening tests we concluded that the performance with the inside training data is quite good. The performance with outside training data is only satisfactory for the utterances spoken by male speakers. The utterance spoken by a female speaker sounds rough and distorted. This effect is also noticeable in Fig. 6.27, and is mainly due to the small training set, and the relative low amount of "female" speech. The appropriate size of the training sequence is hard to determine, but different researchers proposed that the size should at least be 50 times larger than the code book size. Furthermore, the training sequence should consist of speech samples spoken by many different speakers and recorded with different acoustic environments. To further improve the performance of VQ, larger code books should be used, which requires for sub-optimal search procedures to enable practical implementations.

6.5 Summary

In this chapter we described efficient procedures for encoding the positions of the MPE excitation sequence. These procedures possess a low complexity and require approximately 5 bits per pulse position. We further examined the effects of quantization of the parameters of the MPE and RPE coders. It was shown that the predictor parameters can be efficiently coded with 44 bits per set of 12 coefficients. This results in a bit rate of 2200 b/s for a 20 ms rate, which leaves us with approximately 7000 b/s for the encoding of the excitation signal. The excitation signal can be coded with an APCM coder which normalizes to the maximum absolute value within a frame. The use of a Lloyd-Max quantizer improves the quality of the synthetic signal but good results were also obtained with a uniform quantizer. The use of a pitch predictor has a positive result on the perceived quality for both the MPE and the RPE coder, although the latter also performs equally well without a pitch predictor. Using 3 bits/pulse, the excitation signal can be encoded with 6400 b/s, so that the total number of bits will not exceed 9600 b/s.

References

Atal, B.S., "Predictive coding of speech at low bit rates," *IEEE Trans. Communications* Vol. **COM-30**(4) pp. 600-614 (April 1982).

Buzo, A., A.H. Gray, R.M. Gray, and J.D. Markel, "Speech coding based upon vector quantization," *IEEE Trans. Acoust., Speech, Signal Processing* Vol. **ASSP-28** pp. 562-574 (Oct. 1980).

Cover, T.M., "Enumerative source encoding," *IEEE Trans. Inform. Theory* Vol. **IT-19**(1) pp. 73-77 (Jan. 1973).

Fleischer, P., "Sufficient conditions for achieving minimum distortion in a quantizer," *IEEE Int. Conv. Rec.*, pp. 104-111 (1964).

Fussel, J.W., "The Karhunen-Loeve transform applied to the log area ratios of a linear predictive speech coder," *Proc. IEEE Int. Conf. Acoust., Speech and Signal Processing*, pp. 462-465 (1980).

Gish, H. and J.N. Pierce, "Asymptotically efficient quantization," *IEEE Trans. Inform. Theory* Vol. **IT-14** pp. 676-681 (1968).

Golomb, S.W., "Run-length encodings," *IEEE Trans. Inform. Theory* Vol. **IT-12** pp. 399-401 (July 1966).

Gray, A.H. and J.D. Markel, "Quantization and bit allocation in speech processing," *IEEE Trans. Acoust., Speech, Signal Processing* Vol. **ASSP-24** pp. 459-473 (Dec. 1976).

Gray, A.H. and J.D. Markel, "Distance measures for speech processing," *IEEE Trans. Acoust., Speech, Signal Processing* Vol. **ASSP-24** pp. 380-391 (Oct. 1976).

Gray, A.H., J.D. Markel, and R.M. Gray, "Comparison of optimal quantizations of speech reflection coefficients," *IEEE Trans. Acoust., Speech, Signal Processing* Vol. **ASSP-25** pp. 9-23 (Feb. 1977).

Gray, A.H. and J.D. Markel, "Implementation and comparison of two transformed reflection coefficient scalar quantization methods," *IEEE Trans. Acoust., Speech, Signal Processing* Vol. **ASSP-28** pp. 575-583 (Oct. 1980).

Gray, R.M., A. Buzo, A.H. Gray, and Y. Matsuyama, "Distortion measures for speech processing," *IEEE Trans. Acoust., Speech, Signal Processing* Vol. **ASSP-28** pp. 367-376 (August 1980).

Huffman, D.A., "A method for the construction of minimum-redundancy codes," *Proc. IRE* Vol. **40** pp. 1098-1101 (1966).

Jelinek, F. and K.S. Schneider, "On variable-length-to-block coding," *IEEE Trans. Inform. Theory* Vol. **IT-18** pp. 765-774 (Nov. 1972).

Juang, B.H., D.Y. Wong, and A.H. Gray, "Distortion performance of vector quantization for LPC voice coding," *IEEE Trans. Acoust., Speech, Signal Processing* Vol. **ASSP-30** pp. 294-304 (1982).

Lloyd, S.P., "Least squares quantization in PCM," *IEEE Trans. Inform. Theory* Vol. **IT-28** pp. 129-137 (1957). reprint March 1982

Makhoul, J. and M. Berouti, "Adaptive noise spectral shaping and entropy coding in predictive coding of speech," *IEEE Trans.*

Acoust., Speech, Signal Processing Vol. **ASSP-27** pp. 247-254 (1979).

Markel, J.D. and A.H. Gray, "A linear prediction vocoder simulation based upon the auto-correlation method," *IEEE Trans. Acoust., Speech, Signal Processing* Vol. **ASSP-22** pp. 124-134 (April 1974).

Max, J., "Quantizing for minimum distortion," *IRE Trans. Inform. Theory* Vol. **IT-6** pp. 7-12 (March 1960).

Noll, P., "Adaptive quantizing in speech coding systems," *Proc. Int. Zurich Seminar*, pp. B3(1)-B3(6) (March 1974).

Paez, M.D. and T.H. Glisson, "Minimum mean squared error quantization in speech PCM and DPCM systems," *IEEE Trans. Communications* Vol. **COM-20** pp. 225-230 (April 1972).

Rabiner, L.R. and R.W. Schafer, *Digital processing of speech signals*, Prentice Hall, Englewood Cliffs, New Jersey (1978).

Roucos, S., R.M. Schwartz, and J. Makhoul, "Vector quantization for very-low-rate coding of speech," *Proc. GLOBCOM*, pp. 1074-1078 (1982).

Sambur, M.R., "An efficient linear-prediction vocoder," *Bell Syst. Tech. J.* Vol. **54**(10) pp. 1693-1723 (Dec. 1975).

Schalkwijk, J.P.M., "An algorithm for source coding," *IEEE Trans. Inform. Theory* Vol. **IT-18**(3) pp. 395-399 (May 1972).

Segall, A., "Bit allocation and encoding for vector sources," *IEEE Trans. Inform. Theory* Vol. **IT-22** pp. 162-169 (March 1976).

Sluyter, R.J., G.J. Bosscha, and H.M.P.T. Schmitz, "A 9.6 kbit/s speech coder for mobile radio applications," *Proc. IEEE Int. Conf. Communications*, (May 1984).

Soong, F.K. and B.H. Juang, "Line spectrum pair (LSP) and speech data compression," *Proc. IEEE Int. Conf. Acoust., Speech and Signal Processing*, pp. 1.10.1-1.10.4 (March 1984).

Tohkura, Y. and F. Itakura, "Spectral Sensitivity Analysis of PARCOR parameters for speech data compression," *IEEE Trans. Acoust., Speech, Signal Processing* Vol. **ASSP-27**(3) pp. 273-280 (June 1979).

Tremain, T.E., "The government standard linear predictive coding algorithm:LPC-10," *Speech Technology*, pp. 40-49 (April 1982).

Viswanathan, R. and J. Makhoul, "Quantization properties of transmission parameters in linear predictive systems," *IEEE Trans. Acoust., Speech, Signal Processing* Vol. **ASSP-23** pp. 124-321 (June 1975).

Wakita, H., "Linear prediction voice synthesizers: line-spectrum pairs (LSP) is the newest of several techniques," *Speech Technology* Vol. 1(1) pp. 17-23 (Fall 1981).

Wong, D.Y., B.H. Juang, and A.H. Gray, "An 800 bits/s vector quantization LPC vocoder," *IEEE Trans. Acoust., Speech, Signal Processing* Vol. ASSP-30 pp. 770-780 (Oct. 1982).

7. CODER PERFORMANCE EVALUATION

7.1 Introduction

In the previous chapter we described efficient quantization and encoding procedures for the RPE and MPE coder parameters. In this chapter we evaluate the coder performance at various bit rates. We start with evaluating the performance at bit rates between 10 and 16 kb/s, in which range we aim to encode (near) toll quality speech. We further investigate the performance of the coders at lower bit rates (around 5 kb/s). To enable an adequate coder performance at these low rates, vector quantization techniques are used for encoding the excitation sequence. Another interesting aspect is the performance of the RPE and MPE coders at higher sampling rates (16 kHz), to provide wide band speech coding (7 kHz bandwidth), which has gained increasingly more interest with the advent of additional telephone services such as teleconferencing. Furthermore, we tested the coders using speech samples corrupted by background noise, and by using multi-speaker inputs.

7.2 Performance At 9.6 Kb/s

In Chapter 6 we separately studied the effects of the quantization of the filter coefficients and the excitation sequence. In this chapter we report on the results if both types of quantization are combined, as will be the case for normal coder operation. The quantization parameters are chosen such that the total bit rate of 9600 b/s is not exceeded. The parameter values used are listed in Table 7.1.

MPE and RPE filter coefficients		b/s
update rate	20 ms	
filter order	12	
quantization method	minimum deviation	
bits/set	44	
		2200
MPE excitation		
method	MPEQ1	
pulses/second	800	
bits/amplitude	3	
bits/position	5	
quantization method	Lloyd-Max non-uniform	
		6400
gain normalization	maximum abs. value	
gain update rate	10 ms	
bits/gain coefficient	6	
		600
TOTAL NUMBER OF BITS/SECOND MPE		9200

RPE excitation		
method	RPEQ2	
pulses/second	2000	
bits/amplitude	3	
bits/shift factor	2	
quantization method	Lloyd-Max non-uniform	
		6400
gain normalization	maximum abs. value	
gain update rate	10 ms	
bits/gain coefficient	6	
		600
TOTAL NUMBER OF BITS/SECOND RPE		9200

TABLE 7.1. Parameter values for MPE and RPE coders operating at 9600 b/s

From this table we see that both excitation signals require the same amount of bits. However, the RPEQ1 method makes it necessary to update the gain for every search frame. It is assumed that by sub-segment gain factors or differential coding (see Section 6.2.6) the given rate can be obtained. The addition of a pitch predictor will require an additional 500 bits per second and the 9600 b/s rate will be exceeded. By applying variable length coding techniques the MPE excitation signal can be coded with 7 bits per pulse, thereby enabling the use of a pitch predictor. From informal listening tests we conclude that the overall degradation of the synthetic speech signal caused by quantization is small but noticeable. This degradation was more perceptible for speech samples that were also difficult to encode without quantization. By adding a pitch predictor to the MPE coder and using 7 bits/pulse the quality of the MPE encoded signals was judged to be comparable to the RPE encoded signals (without pitch prediction).

7.3 Performance At 16 Kb/s

We described the coder performance at 9600 b/s, and remarked that the synthetic speech quality was very close to the 12-bits PCM encoded original signal, but that differences could still be heard. If the allowed bit rate is 16 kb/s, we expect that coder transparency, i.e. no perceptible difference between the original and the synthetic signal, can be achieved. If more bits than 9600 b/s are available, a decision has to be made about how to spend these bits. For example, the update rate of the predictor coefficients may be increased, or the number of pulses/second. Obviously, there exists a trade-off among these parameters. In the following we investigate the trade-off between the number of pulses/second and the number of quantizer levels. The pulse rate of the MPE coder can be varied more gradually than that of the RPE coder, where a change in the pulse spacing NS strongly affects the pulse rate. This makes it more difficult to adjust the RPE coder to the desired bit rate. In Table 7.2 two possible parameter sets are listed for the various coders, each resulting in approximately the same bit rate.

coder	bits/pulse	pulses/s	b/s
MPE1			
pred. coefficients (20 ms rate)			2200
gain (10 ms rate)			600
8 level quantizer	3+5	1600	12800
TOTAL			15600
MPE2			
pred. coefficients (20 ms rate)			2200
gain (10 ms rate)			600
16 level quantizer	4+5	1400	12600
TOTAL			15400
RPE1 ND=40, NS=2			
pred. coefficients (20 ms rate)			2200
gain (5 ms rate)			1200
8 level quantizer	3	4000	12200
TOTAL			15600
RPE2 ND=30, NS=3			
pred. coefficients (15 ms rate)			2930
gain (3.75 ms rate)			1600
16 level quantizer	4	2667	11200
TOTAL			15730

TABLE 7.2. Possible parameter values for MPE and RPE coders operating at 16 kb/s.

Obviously, many other possibilities exist. For example, the coders can be extended with a pitch predictor or the update rate of the predictor coefficients can be increased. For the parameter settings listed in Table 7.2 the versions with 16 level quantizers were preferred over those with an 8 level quantizer. We also found that the difference between 3 and 4 bit quantizers is more noticeable for the RPE coder, which is due to the wide distribution of the pulse amplitudes.

A comparison between the synthetic speech produced by the coders with 4 bit quantizers and the original PCM encoded signals revealed that both signals were almost undistinguishable.

7.4 Performance At Low Bit Rates

We have shown that the RPE and MPE coders can provide (near) toll quality speech at rates around 10 kb/s. In this section we describe the performance of the RPE coder at lower rates. Although it will be obvious that the quality of the synthetic signal will decrease, we aim to keep this degradation as low as possible. To reduce the bit rate we have three possibilities, which can be applied simultaneously or separately. First, the number of coder parameters can be reduced, and, second, the coder parameters can be quantized more coarsely. Another possibility is to reduce the sampling frequency, but this rather trivial solution is not considered. From the previous chapter we know that a considerable amount of the available bits is spent on encoding the excitation sequence. As said before we can reduce the number of parameters by decreasing the number of pulses per second. Some resulting bit rates for the RPE coder are shown in Table 7.3. Notice that the gain is not considered.

number of bits/pulse	4	3	2	1
pulse spacing = 4	8000	6000	4000	2000
pulse spacing = 5	6400	4800	3200	1600

TABLE 7.3. Bit rates for different pulse spacings and different number of bits/pulse.

From this table we see that the largest reduction is obtained by reducing the number of bits/pulse. From Chapter 3 we know that a further increase in the pulse spacing (NS > 5) leads to rough distorted sounds. Hence we stick to a pulse spacing NS=4, and try to improve the quantization procedures by the use of Vector Quantization (Appendix D) and pitch prediction. The vector quantization procedure proceeds as follows: once an amplitude vector b_k (Section 3.3.1) has been found, it is normalized to either its maximum absolute value or to its rms value. The normalized excitation vector b_k is compared to each code book vector y_i, i=1,N. The code book vector that has the smallest distance to b_k is selected, and its index is transmitted. As distance or distortion measure the squared error is used, that is

$$d(x,y_i) = (x-y_i)(x-y_i)^t = |x-y_i|^2 \qquad (7.1)$$

Note that the same criterion is used in the case of scalar quantization. When considering the complete excitation vector one could argue that we should continue to use the weighted distance between the original and the synthetic speech signals. However, this means that we would obtain a procedure similar to the multi-path search code book approach (Chapter 5). By using the mean squared error criterion for the amplitude vectors, a much lower complexity is obtained.

About 70 seconds of speech spoken by two male speakers and one female speaker was analyzed with the RPE method using a pulse spacing NS=4 and a search frame size NSSIZE=40. The resulting excitation vectors of dimension 10 were normalized to their maximum absolute value and used as a training sequence in the code book design procedure. To inhibit the inclusion of silent intervals, only excitation vectors whose rms value exceeded 7 dB were used. A code book of size 1024 was designed with the binary split algorithm and a convergence threshold ϵ=0.005 (Appendix D). The code book size of 1024 results in a bit rate of 1 bit/ pulse or 1/4 bit sample. To illustrate the effect of vector quantization, we show in Fig. 7.1 the average distortion per pulse when L adjacent pulses are quantized as a vector. The parameter L is called the vector length.

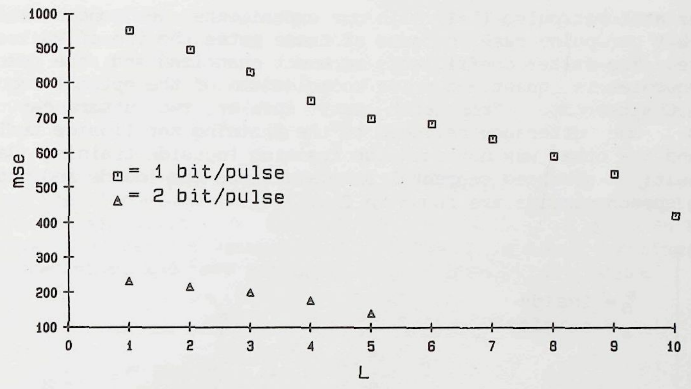


Figure 7.1. Average distortion per pulse for increasing values of L.

For L=1, we obtain the scalar case. Note that the dimension of the amplitude vector remains constant, but that for different values of L, groups of L adjacent pulses are considered as a vector. From Fig. 7.1 we see that by quantizing the vector at once, a large reduction in error is obtained compared to scalar quantization, but that scalar quantization with 2 bits/pulse still provides better results. The effect of the code book size is shown in Fig. 7.2

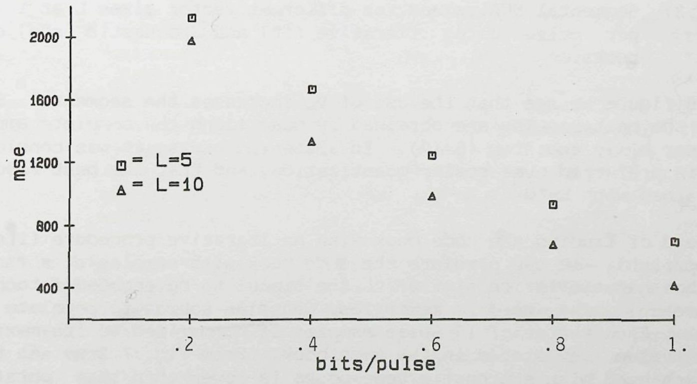


Figure 7.2. Average distortion per pulse for different code book sizes and different values of L.

A comparison of Figs. 7.1 and 7.2 reveals that the performance of a vector quantizer with 6/10 bit pulse (L=10) is equal to that of a scalar quantizer at 1 bit/pulse (L=1). In our experiments we concentrate on the one-bit per pulse case, because at these rates the use of VQ is more effective. The filter coefficients were not quantized and the excitation sequence is quantized prior to selection of the optimal sequence (RPEQ2, Chapter 3). From each test speaker two utterances were selected. One utterance belonged to the training set (inside training data), and the other was not used for training (outside training data). The resulting averaged segmental SNR values for the inside and outside training speech samples are shown in Fig. 7.3.

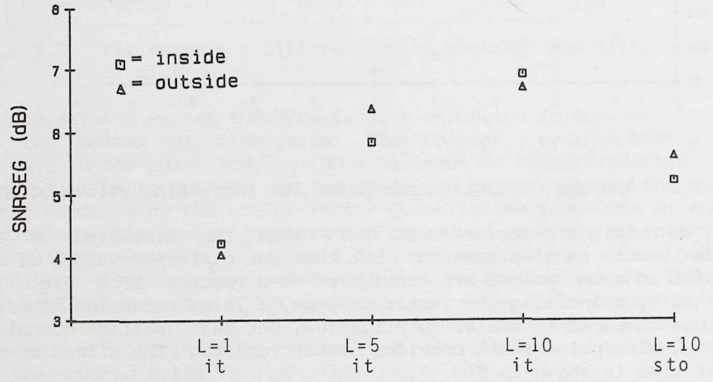


Figure 7.3. Segmental SNR values for different vector sizes L at 1 bit per pulse using iterative (it) and stochastic (sto) code books.

From this figure we see that the use of VQ increases the segmental SNR, and that the best results are obtained by quantizing the complete amplitude vector b_k in one step (L=10). In listening tests, it was concluded that VQ is preferred over scalar quantization, and that the best results were obtained with L=10.

Instead of finding the code book with an iterative procedure (iterative approach), we can populate the code book with samples of a random source that has statistics similar to the signal to be encoded (stochastic approach). We used a memoryless Gaussian source to populate the code book. Each vector of 10 noise samples is normalized to its maximum absolute value and stored in the code book. From Fig. 7.3 we see that the SNR achieved with stochastic code books is lower than that obtained with iterative code books, but that there is still a significant gain over the scalar case. To correctly interpret the differences between the inside and outside training data, we note that without quantization the average SNRSEG for the outside data is approximately 2 dB higher than for the inside training data (15.10 dB versus 13.21 dB). This difference is visible when a stochastic code book is used, and no

distinction between inside and outside training data can be made. For the scalar quantizer and VQ (L=10), the performance for the inside training data is much better than for the outside training data. For VQ with L=5 this is not the case, and this is mainly due to the inefficiency of normalizing a vector of size 10 to the maximum absolute value, and then quantizing the vector in two steps.

In previous chapters we discussed the use of pitch prediction to remove the long-term correlation in the excitation signals. As a result, the signal is less "peaked" and easier to quantize. In Fig. 7.4 we show the results obtained with different parameter settings of the pitch predictor, and both stochastic and iterative code books.

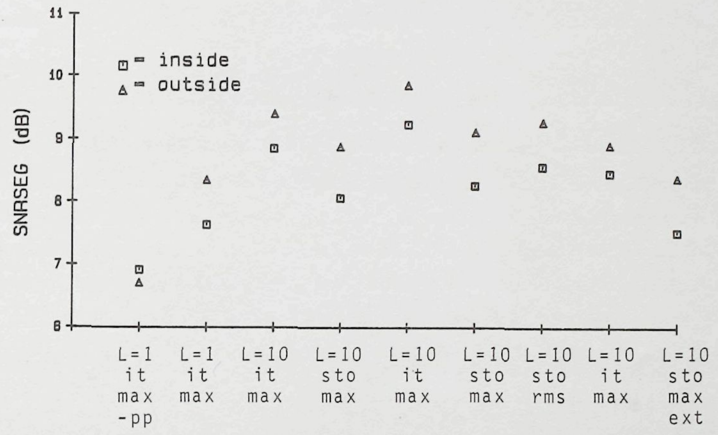


Figure 7.4. Averaged segmental SNR values for quantization of the excitation sequence with 1 bit per pulse. Code books are iterative (it) or stochastic (sto), and normalization is done either to the absolute maximum value (max) or the rms value.

From this figure we see that at these low rates, the use of pitch prediction dramatically increases the SNR. A one-tap pitch predictor is used and the value of ß is quantized with 4 bits. Two ranges of M have been used: M=16:80 (6 bits), and M=16:144 (7 bits). The largest range gives the best results in terms of SNR values and listening tests. The resulting excitation signal is more easily modeled by a Gaussian signal after pitch prediction, and we see from Fig. 7.4 that the performance of the stochastic code books is quite close to that of the iterative code books. The improvement due to pitch prediction is confirmed by listening tests. In Fig. 7.4 we also shown the results obtained with normalization to the rms value, and with a stochastic code book. We see that compared to normalization to the maximum absolute value there is a slight increase in SNR, but in listening tests no differences could be heard. From the same figure we see that when the pitch predictor parameters are determined from the residual (ext, see also Chapter 2), we loose approximately 0.5 dB in SNR. The best results are obtained with an iterative code book and M=16,144, but we were impressed by the results obtained with the stochastic code book. The SNRSEG as a function of time for the utterance "a lathe is a big tool" for the procedure with the iterative code book and M=16:144 is shown in Fig. 7.5 for a male and a female speaker. The dashed line represents the results obtained with a scalar Lloyd-Max quantizer. We see that with VQ the segmental SNR shows a smooth behavior, and results in a somewhat higher SNR than obtained with scalar quantization.

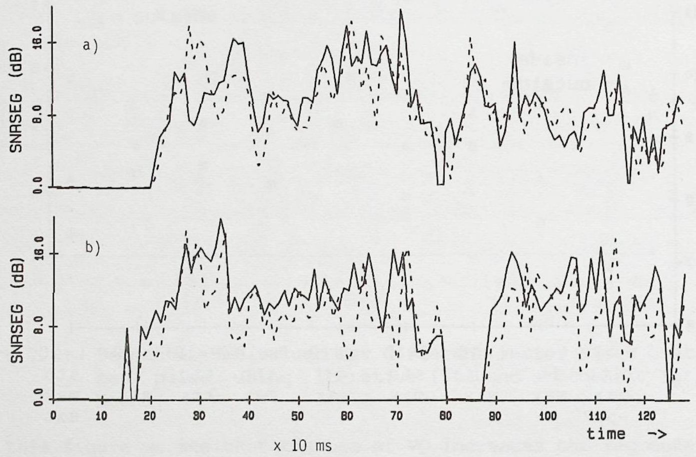


Figure 7.5. SNRSEG obtained with pitch prediction (M:16,144) and 1 bit per pulse for (a) female and (b) male speaker using a scalar Lloyd-Max quantizer (dashed line) or vector quantization with iterative code book (solid line is)

At 2 bits/pulse the effect of VQ is less dramatic. Fig. 7.6 shows the averaged SNRSEG obtained with and without pitch prediction, for both scalar and vector quantizers. The range of M was chosen between 16 and 144.

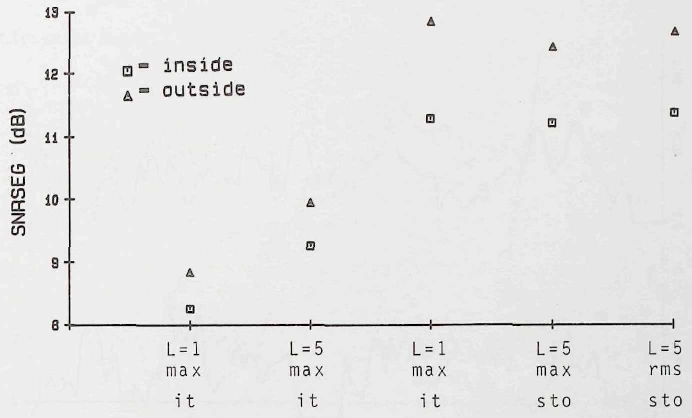


Figure 7.6. Averaged SNRSEG values using different quantization procedures at 2 bits/pulse.

Without a pitch predictor the use of VQ improves the performance, but when a pitch predictor is used, the effect of VQ is not significant. Note that we used only stochastic code books for the last case. An iterative code book may give some improvement, but we think that this improvement does not justify the resulting complexity of the code book generation procedure.

In Chapter 5 we described a multi-path search procedure using a stochastic code book approach that also operates at 1/4 bit sample. The only difference is that in this chapter the mean squared error between the computed excitation vector and the code book vector is minimized. A comparison between the SNR values for the utterance "a lathe is a big tool" is shown in Fig. 7.7

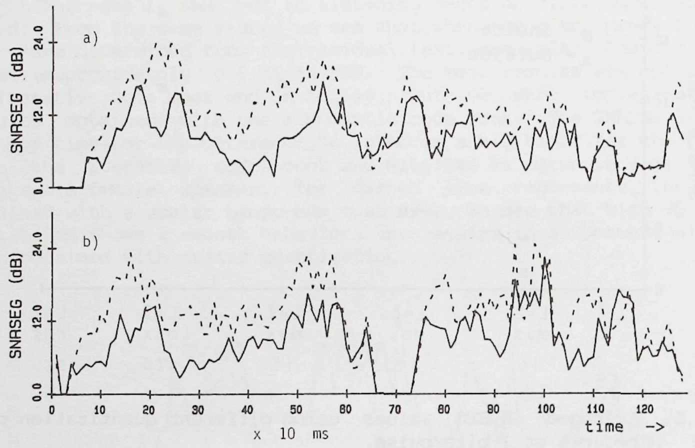


Figure 7.7. Segmental SNR values using a code book approach with either minimization between the excitation sequences (solid line) or between the speech sequences (dashed line). a) female, b) male speaker.

The differences are due to the effects caused by the predictor filters, which cannot be taken into account if the distortion between excitation vectors is minimized.

The use of VQ gives an improvement in SNR but increases the complexity of the coder. This increase is due to the additional memory requirements for code book storage, and the computational effort of finding the appropriate code book vector. An alternative is to use more structured code books, to enable a more efficient use of memory and the use of fast search procedures. Lattice quantizers [Gersho, 1982] have such a regular structure and can be described as a quantizer whose output set is a sub set of a lattice (see also Appendix D). Many interesting lattices have been published [Sloane, 1981], and efficient coding and decoding algorithms exist for these lattices [Conway and Sloane, 1982]. Generally, a lattice quantizer can be considered as a uniform vector quantizer. By an appropriate choice of a lattice and a particular subset of it, the code book need not be stored, and the encoding and decoding procedures can be implemented very efficiently [Adoul et al., 1984]. In Fig. 7.8 we show some results obtained with the first sphere of lattice E_8 (Appendix D). The RPE excitation was computed every 32 samples with a pulse spacing NS=8, such that the remaining excitation vector of size 8 could be quantized by one of the 240 code book vectors (LQ2). In [Adoul et al., 1984] a somewhat different definition was given of these 240 vectors, and 16 additional vectors were added to yield a size 256 code book (LQ1). For comparison, we also give the results obtained with a

stochastic code book.

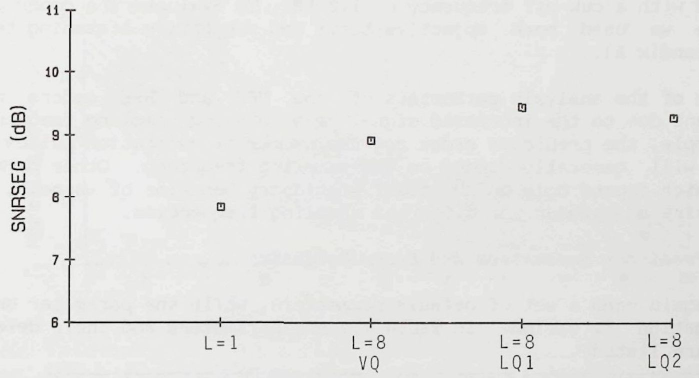


Figure 7.8. Segmental SNR values obtained with 1 bit/pulse lattice quantizers.

As can be seen, the performance of the three coders is almost equal, which was confirmed by listening tests. In each case a pitch predictor is used, and the excitation vector is normalized to the rms value. This result makes it very attractive to use lattice quantizers. Moreover, a lattice quantizer enables the use of large code books, which need not be stored, and may still provide a gain over scalar quantization.

Using a 1 bit per pulse vector quantizer in the bit allocation scheme of Table 7.1, we obtain a 5.2 kb/s rate for the RPE coder without a pitch predictor, and a 6.2 kb/s bit rate for the RPE coder with a pitch predictor. The use of a pitch predictor ensures that the coder provides a good communication quality at approximately 6 kb/s.

7.5 Wide band Speech Coding

Recently, several techniques have been developed to handle the problem of transmitting high quality audio signals [Johnston and Goodman, 1979] [Bertorello et al., 1982] [Johnston and Chrochiere, 1979] or wide band speech [Combescure et al., 1982] [Snyder, 1984] at a bit rate of 64 kb/s. The evolution towards an integrated digital services network [Decina, 1982], however, has created an interest in the development of coders which can provide commentary grade quality speech at a rate less than 64 kb/s. Also in applications such as tele or video conferencing a rate of 64 kb/s is too high and needs further reduction. In this section we will examine the suitability of the MPE and RPE coders for wide-band speech coding at a rate less than 32 kb/s.

speech signals. This result was confirmed by listening tests.

The *noise weighting filter* $W(z)= A(z)/A(z/\gamma)$ is determined by the parameter γ. Decreasing the value of γ increases the bandwidth of the poles of $W(z)$. In Chapters 3, 4 and 5 we concluded that a value of γ between 0.80 and 0.85 gives the best results. This corresponds to an increase in bandwidth of approximately 500 Hz. For a sampling frequency of 16 kHz the same increase in bandwidth necessitates a value of approximately 0.95. As noted before, noise-shaping reduces the SNR, but improves the perceived speech quality. Fig. 7.11 shows the results of subjective listening tests for different values of gamma.

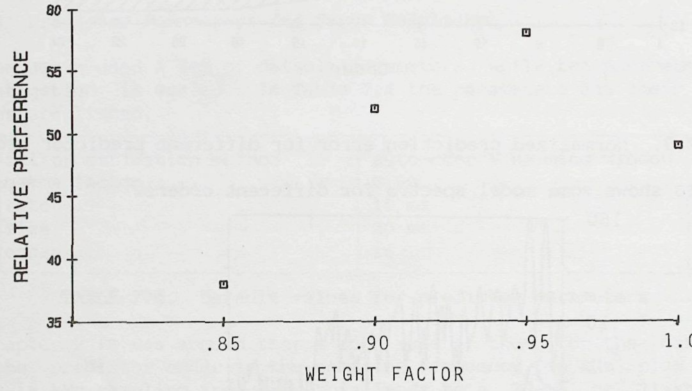


Figure 7.11. Results of listening tests for different values of γ.

These listening tests revealed that the effect of noise shaping is small but can be heard. An optimal value for γ was found to be between 0.90 and 0.95. A value of 1.0 is not optimal but the difference in perceived speech quality is very small, and because of the reduction in complexity this value might be used in real-time implementations.

7.5.2 *Excitation Parameters*

7.5.2.1 *Multi-Pulse Excitation Coder* As was shown in Chapter 4 the best results are obtained when the search frame size is equal to the minimization frame size. In the same chapter it was explained why small search frames (less than 5 ms) are not optimal. Due to computational reasons the search frame size must be chosen as small as possible. So the search frame size was chosen to be equal to 5 ms. The number of pulses was varied from 8 to 12 and we used the sub-optimal procedure SUB2 (Section 4.3), to determine the excitation sequence. The segmental SNR increases with these values (see Fig. 7.12), but listening tests showed no significant improvement in the perceived speech quality among 10, 11, and 12 pulses/ 5 ms.

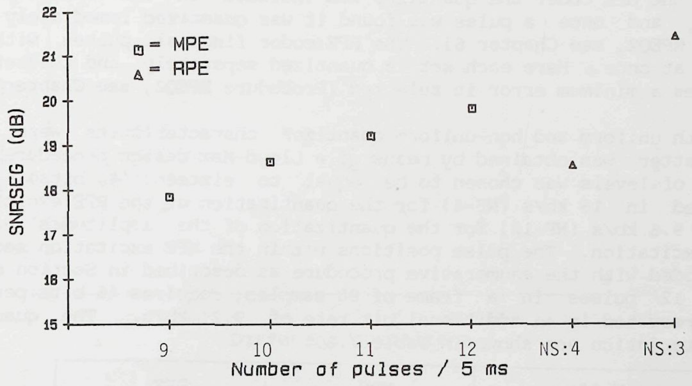


Figure 7.12. Segmental SNR for different pulse rates in the MPE and RPE sequences.

7.5.2.2 *Regular-Pulse Excitation Coder* In order to compare both coders and due to the same computational and qualitative reasons as mentioned above, the search frame was chosen to be equal to 5 ms. The pulse spacing was varied from 4 to 3. The segmental SNR increases with decreasing pulse spacing (Fig. 7.12). However, in listening tests it was hard to distinguish between the synthetic speech samples produced with different pulse spacings.

7.5.3 *Quantization And Encoding Of The Excitation Parameters*

The amplitude distribution functions of the normalized excitation signals of both the MPE and RPE coders are shown in Fig. 7.13. The excitation signal was normalized to the maximum absolute value within a frame of 5 ms.

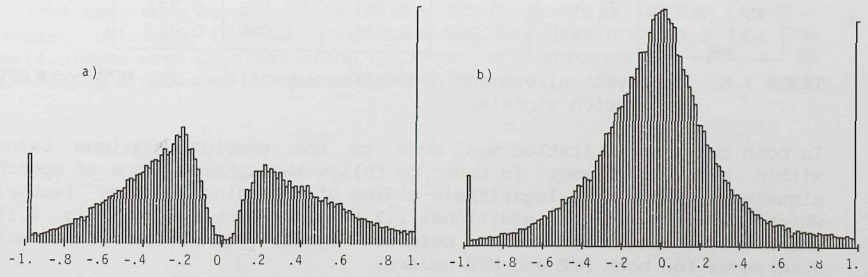


Figure 7.13. Amplitude distribution histograms (a) MPE, (b) RPE.

In the MPE coder the quantizer was included in the analysis procedure, and once a pulse was found it was quantized immediately (Procedure MPEQ2, see Chapter 6). The RPE coder finds all pulses within a frame at once. Here each set is quantized separately, and the set that produces a minimum error is selected (Procedure RPEQ2, see Chapter 6).

Both uniform and non-uniform quantizer characteristics were used, the latter was obtained by means of a Lloyd-Max design procedure. The number of levels was chosen to be equal to sixteen (4 bits), which resulted in 16 kb/s (NS=4) for the quantization of the RPE excitation, and in 9.6 kb/s (NP=12) for the quantization of the amplitudes of the MPE excitation. The pulse positions within the MPE excitation sequence were coded with the enumerative procedure as described in Section 6.3.2. Using 12 pulses in a frame of 80 samples, requires 46 bits per set, which resulted in an additional bit rate of 9.2 kb/s. The quantizer characteristics are shown in Table 7.5.

	Uniform		MPE		RPE	
i	x	y	x	y	x	y
0	-1.000	-	-1.000	-	-1.000	-
1	-0.875	-0.938	-0.870	-0.960	-0.839	-0.950
2	-0.750	-0.813	-0.708	-0.778	-0.645	-0.728
3	-0.625	-0.688	-0.575	-0.636	-0.494	-0.561
4	-0.500	-0.563	-0.461	-0.515	-0.369	-0.426
5	-0.375	-0.438	-0.359	-0.408	-0.263	-0.311
6	-0.250	-0.313	-0.262	-0.310	-0.172	-0.216
7	-0.125	-0.188	-0.166	-0.215	-0.087	-0.128
8	-0.000	-0.063	0.001	-0.117	-0.007	-0.046
9	0.125	0.063	0.170	0.120	0.071	0.030
10	0.250	0.188	0.270	0.220	0.157	0.113
11	0.375	0.313	0.375	0.320	0.254	0.202
12	0.500	0.438	0.488	0.430	0.367	0.305
13	0.625	0.563	0.608	0.547	0.506	0.430
14	0.750	0.688	0.740	0.670	0.679	0.582
15	0.875	0.813	0.896	0.811	0.876	0.774
16	1.000	0.938	1.000	0.980	1.000	0.977

TABLE 7.5. 16-level uniform and non-uniform quantizers for MPE and RPE excitation signals.

In both cases normalization was done to the absolute maximum value within a 5 ms frame. In order to follow the dynamic range of speech signals we used 6 bits logarithmic coding of the gain (scaling factor) which resulted in an additional 1.2 kb/s for both coders. Fig. 7.14 shows the segmental SNR values obtained by averaging four different utterances for both RPE and MPE coders.

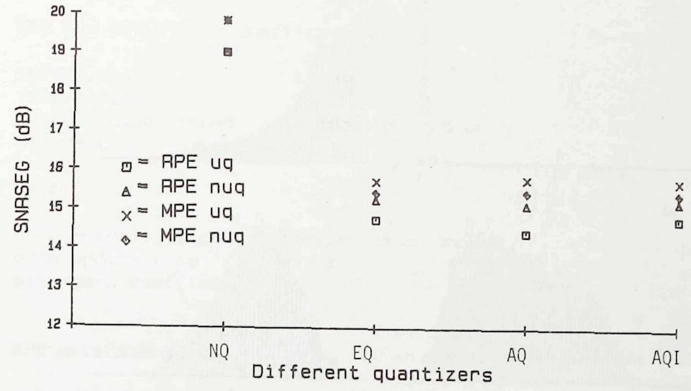


Figure 7.14. Segmental SNR values obtained with different quantization procedures, NQ = Not Quantized; EQ = Only the excitation quantized; AQ = Excitation and reflection coefficients quantized, rate 10 msec; AQI = AQ with linear interpolation of the reflection coefficients (5 ms rate).

This figure shows that for the RPE coder the segmental SNR increases when non-uniform quantization is used. However, for the MPE coder this value decreases slightly. Listening tests revealed that for the RPE excitation non-uniform quantization was preferred over uniform quantization, and that these differences were significant. Non-uniform quantization of the MPE sequence also gave a better perceived quality. The differences with uniform quantization, however, were very small though noticeable.

7.5.4 *Quantization And Encoding Of The Filter Parameters*

The amplitude distributions of the first four reflection coefficients, obtained form a large database, are shown in Fig. 7.15. The coefficients were obtained using the auto-correlation method with a 20 ms Hamming window and an update rate of 10 ms.

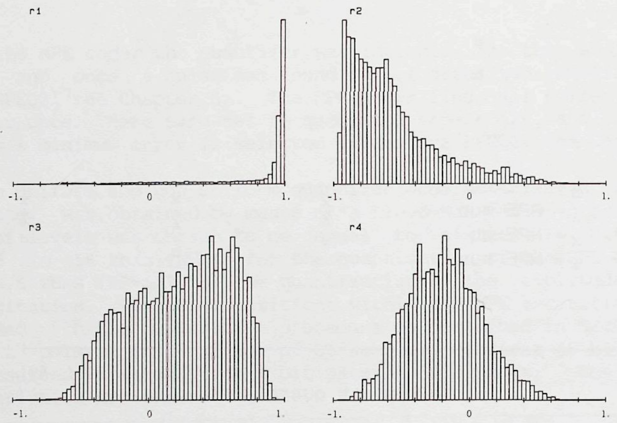


Figure 7.15. Histograms of the first four reflection coefficients.

The first two coefficients show an even greater skewing effect towards +1 and -1, respectively, than in the 8 kHz case. The distributions of the remaining coefficients are reasonably well modeled by Gaussian probability density curves.

Before the reflection coefficients were quantized, they were transformed into inverse sine coefficients to be able to use the Minimum Deviation quantizer design procedure (Section 6.4). The quantizer was calculated with a design limit of 0.29 dB, which resulted in 68 bits/set, and requires an additional 6.8 kb/s for both coders. The bit allocation is shown in Table 7.6.

coeff.	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
bits	7	6	6	5	5	5	4	4	4	4	3	3	3	3	3	3

TABLE 7.6. Bit allocation using an MD quantizer.

7.5.5 *Performance At Rates Below 32 Kb/second*

To enable encoding of commentary grade speech below 32 kb/s, we used the parameter settings as listed in Table 7.7.

MPE and RPE filter coefficients		b/s
update rate	10 ms	
filter order	16	
quantization method	Minimum deviation	
bits/set	68	
		6800
Gain		
gain normalization	maximum abs. value	
gain update rate	5 ms	
bits/gain coefficient	6	
		1200
MPE excitation		
method	MPEQ2	
pulses/second	2400	
bits/amplitude	4	
position bits/set	46	
quantization method	Lloyd-Max non-uniform	
		18800
TOTAL NUMBER OF BITS/SECOND MPE		26800
RPE excitation		
method	RPEQ2	
pulses/second	4000	
bits/amplitude	4	
bits/shift factor	2	
quantization method	Lloyd-Max non-uniform	
		16800
TOTAL NUMBER OF BITS/SECOND RPE		24800

TABLE 7.7. Parameter settings for MPE and RPE coders for the encoding of wide band speech below 32 kb/s.

Although both coders operate approximately at the same bit rate, listening tests revealed that most listeners had a slight preference for the RPE encoded signals. A comparison between PCM encoded signals and the RPE encoded signals, revealed that in most cases almost no differences could be heard, but that for some test sentences some distortion could be heard. Spending more bits on the encoding of the filter coefficients or increasing the filter order will possibly improve the quality.

7.6 Performance For Degraded Speech

Thus far, all simulations have been performed with non-degraded speech signals, where non-degraded means that these samples were recorded under ideal conditions without any background noise. In practice, the input signals for speech coding systems will be more or less degraded by background noises due to, for example, background conversation, air conditioning units, typewriters, etc. Another cause of signal degradation can be found when two or more coding systems are connected in cascade (tandem operation). In that case, the coder input consists of the speech signal corrupted by the encoding distortion of previous encoding stages.

To reduce the effects of background noise special care can be taken during the recording procedure by using special microphones, etc. Another possibility is to apply pre-processing techniques to reduce the noise contents in the speech signal [Lim and Oppenheim, 1978] but these techniques produce generally undesired distortions.

Background noises will change the shape of the signal spectrum and will thereby affect the performance of the predictor. In [Sambur and Jayant, 1976] it was reported that in the case of additive white noise, an SNR greater than 17.5 dB is needed to preserve the spectral characteristics of the original signal. In the class of coders described so far, the predictor order was selected to be optimal for speech signals. A spectrum rather different from a speech spectrum makes the predictor less efficient, and a decrease in performance can be expected.

Some noise sources have their energy concentrated in a specific frequency band (e.g. telephone ringing), but most background noises have their energy equally distributed over the speech frequency band and are additive in nature.

In our simulations, background noise is simulated by adding samples of a memoryless Gaussian noise source with zero mean and variance σ^2 to the original speech samples. The SNR of the resulting composite signal is given by

$$\mathrm{SNR} = \sum_n x^2(n) \;/\; (\sigma^2 . n) \qquad (7.2)$$

where x(n) represent the speech samples and n represents the number of samples in an utterance. RPE and MPE analysis is performed on degraded speech samples spoken by a male and a female speaker. Fig. 7.16 shows the results for different SNR noise levels.

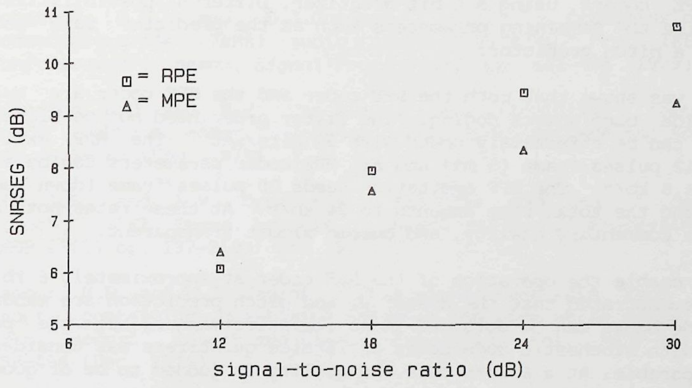


Figure 7.16. SNRSEG values for speech degraded by additive white noise.

All simulations were performed without quantization of the coder parameters. From this figure it is clear that the RPE coder is more affected by the noise than the MPE coder is. This result was confirmed by listening tests. Especially for a 12 dB SNR, the RPE encoded signals exhibited tonal noise effects. For decreasing SNR values, the differences between the two coders were less perceivable and for the highest SNR value a slight preference existed for the RPE coder. The low performance of the RPE coder at a 12 dB SNR is ascribed to the fact that, if the predictor is not effective (due to the flat input spectrum), the RPE encoding procedure is just a straight down-sampling procedure of the input signal, in which the aliasing components introduce the perceivable tonal noises. The MPE has the advantage in that case that there is more irregularity in the excitation signal, which makes the aliasing components less perceivable.

Another interesting form of degraded speech input is the case of multiple speaker input, where two or more speakers are talking at the same time. As a result the short-time spectra are more complex than for a single speaker, and thus more difficult to model with a low-order predictor. We simulated multi-speaker input by adding two test utterances of a male and a female speaker. In listening tests no significant deterioration could be heard when the signals were encoded with the MPE and RPE coders.

7.7 Summary

In this chapter we demonstrated that the coders provide (near) toll quality at a bit rate of 9600 b/s. The excitation sequence is quantized with an 8 level quantizer and requires approximately 6400 b/s. The remaining bits are used for the encoding of the filter parameters. If a 16 kb/s rate is allowed, toll quality speech can be obtained by both RPE and MPE coders, using a 4 bit quantizer. Different possibilities exist to define the remaining parameters such as the predictor rate and the use of a pitch predictor.

It was shown that both the RPE coder and the MPE coder are suitable for wide band speech coding. The filter order need not be higher than 16 and can be effectively coded with 68 bits/set. The MPE excitation needs 12 pulses/frame (5 ms) and all MPE coder parameters can be encoded with 26.8 kb/s. The RPE excitation needs 20 pulses/frame (down sampling of 4) and the total rate amounts to 24 kb/s. At these rates both coders provide commentary quality, and become almost transparent.

To enable the operation of the RPE coder at approximately 6 kb/s, it was demonstrated that the use of VQ and pitch prediction are mandatory. The code books can be obtained in an iterative manner, but the performance with stochastic code books or lattice quantizers was considered to be comparable. At a 6 kb/s rate the quality is judged to be of good communication quality.

Some preliminary experiments revealed that both the RPE and MPE coders operate quite well for speech corrupted by white noise or for multi-speaker inputs.

References

Adoul, J.P., C. Lamblin, and A. Leguyader, "Baseband speech coding at 2400 bps using spherical vector quantization," *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing*, pp. 10.4.1-10.4.4 (March 1984).

Bertorello, L., M. Copperi, G. Pirani, and F. Rusina, "Broadcasting-quality transmission of audio signals at 64 kbps," *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing*, pp. 1972-1975 (1982).

Combescure, P., A. le Guyader, and M. Haghiri, "ADPCM algorithms applied to wideband speech encoding (64 kbit/s, 0-7 khz)," *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing*, pp. 1976-1979 (1982).

Conway, J.H. and N.J.A. Sloane, "Fast quantizing and decoding algorithms for lattice quantizers and codes," *IEEE Trans. Inform. Theory* Vol. **IT-28** pp. 227-232 (March 1982).

Decina, M., "Managing ISDN through international standard activities," *IEEE Communications Magazine*, pp. 19-25 (Sep. 1982).

Gersho, A., "On the structure of vector quantizers," *IEEE Trans. Inform. Theory* Vol. **IT-28** pp. 157-166 (March 1982).

Johnston, J.D. and D.J.Goodman, "Digital transmission of commentary-grade (7kHz) audio at 56 or 64 kb/s," *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing*, pp. 442-444 (1979).

Johnston, J.D. and R.E. Crochiere, "An all-digital 'commentary grade' subband coder," *Journal of the Audio Engineering Society*, pp. 855-865 (Nov. 1979).

Lim, J.S. and A.V. Oppenheim, "All-pole modeling of degraded speech," *IEEE Trans. Acoust., Speech and Signal Processing* Vol. **ASSP-26**(3) pp. 197-210 (June 1978).

Sambur, M.R. and N.S. Jayant, "LPC analysis/synthesis from speech inputs containing quantizing noise or additive white noise," *IEEE Trans. Acoust., Speech and Signal Processing* Vol. **ASSP-24**(6) pp. 488-494 (Dec. 1976).

Sloane, N.J.A., "Tables of sphere packings and spherical codes," *IEEE Trans. Inform. Theory* Vol. **IT-27**(3) pp. 327-338 (May 1981).

Snyder, J.H., "Experimental hardware for real-time wideband speech coding," *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing*, pp. 19.2.1-19.2.3 (1984).

8. EFFICIENT ALGORITHMS

8.1 Introduction

In the previous chapters we have described techniques for time-domain coding of (near) toll quality speech. The given description of the coders was adequate for non-real time implementations in a high-level language on a general purpose computer. However, for practical implementations, a more detailed description of the encoding and decoding algorithms is required. Moreover, when these algorithms have to be executed in real time with signal processors using fixed point arithmetic, the following additional requirements are placed on the algorithm:

- computational efficiency,
- good numerical properties,
- suitability for the target hardware.

The choice of a particular hardware architecture will have consequences for the structure of the algorithm, leading to an interaction between software efficiency and hardware efficiency. A conventional approach is to use a configuration of one or more (general purpose) signal processors. The analysis procedure is partitioned into different tasks, which will be distributed among the available processors. Within each processor, the task is executed in a sequential fashion. Although this approach can be quite successful, there are several drawbacks. For example, it is very difficult to exploit all the parallel capabilities of a given algorithm, which will result in a loss of efficiency. The sequential execution within each processor inhibits the use of high-throughput pipeline implementations. When it is possible to find an algorithmic structure that has parallel and pipeline properties it might be advantageous to design (special purpose) hardware, tailored to the execution of one or more specific algorithms. With the aid of modern VLSI techniques such a mapping of an algorithm onto silicon might be highly feasible, and could be more cost effective than an implementation with general purpose processors [Kung et al.,1985]. It is important for such a mapping to recognize the basic primitive operations of the underlying algorithm. The algorithm is decomposed into a set of those primitive operations, which will then be directly mapped onto silicon. In this chapter we define the fundamental tasks of a given coder, and we describe efficient decompositions of these tasks.

8.2 Solving A Set Of Linear Equations

A typical basic task to be performed in speech processing is the problem of solving a set of linear equations

$$pA = q \qquad (8.1)$$

where A is an (n+1) by (n+1) real matrix and p and q are (n+1)-dimensional row vectors. For example, the determination of the filter coefficients of the prediction filter A(z) incorporates the solving of such a system. Another example is the computation of a set of pulse amplitudes for either the MPE or RPE coder. This system can be solved with a Gaussian elimination procedure with a complexity of $O(n^3)$. However, for the given examples the matrix A will have a special structure, which enables us to solve the system in fewer than $O(n^3)$ operations. A commonly encountered structure is the symmetric positive Toeplitz structure

$$a_{ij} = a_{|i-j|} = a_k \qquad 0 \le k \le n \tag{8.2}$$

where a_{ij} is the (i,j)-th element of the matrix A. By making use of the Toeplitz structure, Eq. (8.1) can be solved in $O(n^2)$ operations or even less. For example, the Levinson algorithm [Levinson, 1947] recursively solves the n-th order equations in $O(n^2)$ operations. The Levinson algorithm constitutes an LU decomposition of the matrix A^{-1}, namely,

$$A^{-1} = BB^* \tag{8.3}$$

where B is a lower triangular matrix and * denotes complex conjugate transpose. However, the Levinson algorithm is not very suitable for parallel execution, as was pointed out in [Kung and Hu, 1983], and we have to use a different scheme.

An appealing algorithm is the Schur algorithm which has been introduced in [Schur, 1917] and in its current version in [Dewilde et al., 1978]. Unaware of these developments [LeRoux and Gueguen, 1979],, an equivalent algorithm , and the Schur algorithm is sometimes referred to as the LeRoux-Gueguen algorithm. The Schur algorithm constitutes a decomposition of the matrix A

$$A = LL^* \tag{8.4}$$

where L is a lower triangular matrix. Then the solution p of Eq. (8.1) can be solved explicitly via back substitution

$$q = \rho L^* \tag{8.5a}$$

$$p = \rho L^{-1} \tag{8.5b}$$

If we denote the elements of the first row of the matrix A by $\{a_{0i}, i=0,..,n\}$ and a unit delay by z^{-1}, we can write the Schur algorithm as follows:

Schur Algorithm

Initialization:

$$G_0^* = \begin{bmatrix} \Gamma_0 \\ \Delta_0 \end{bmatrix} = \begin{bmatrix} 0 & \gamma_1^{(0)} & . & . & \gamma_n^{(0)} \\ \delta_0^{(0)} & \delta_1^{(0)} & . & . & \delta_n^{(0)} \end{bmatrix} \tag{8.6a}$$

where

$$\gamma_i^{(0)} = \delta_i^{(0)} = \hat{a}_{0i} = a_{0i}/[a_{00}]^{\frac{1}{2}} \qquad i=0,..,n \tag{8.6b}$$

For i = 1 to n do

$$G_i^* = \begin{bmatrix} \Gamma_i \\ \Delta_i \end{bmatrix} = \begin{bmatrix} O_{1xi} & & \gamma_1^{(i)} & . & \gamma_n^{(i)} \\ O_{1x(i-1)} & \delta_0^{(i)} & \delta_1^{(i)} & . & \delta_n^{(i)} \end{bmatrix} = \Theta_i \begin{bmatrix} I & 0 \\ 0 & z^{-1} \end{bmatrix} \begin{bmatrix} \Gamma_{i-1} \\ \Delta_{i-1} \end{bmatrix}$$

where

$$\Theta_i = (1-\rho_i)^{-\frac{1}{2}} \begin{bmatrix} 1 & -\rho_i \\ -\rho_i & 1 \end{bmatrix} \tag{8.7}$$

and

$$\rho_i = \gamma_1^{(i-1)}/\delta_0^{(i-1)} \tag{8.8}$$

The quantities $\{\rho_i\}$ are called reflection coefficients which have to nice property that their absolute value is bounded by one. The matrix Θ_i represents a J-rotation or a hyperbolic rotation and has the property that it is J-orthogonal, that is,

$$\Theta_i J \Theta_i^* = J \qquad J = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} \tag{8.9}$$

which implies that it preserves the norm in the J-metric (where $|x|_j^2 = xJx^*$). The matrix L is given by

$$L = \Delta = \begin{bmatrix} \delta_0^{(0)} & 0 & . & 0 \\ \delta_1^{(0)} & \delta_0^{(1)} & . & 0 \\ . & . & . & . \\ \delta_n^{(0)} & \delta_{n-1}^{(1)} & . & \delta_0^{(n)} \end{bmatrix} \tag{8.10}$$

It is convenient to use a lattice filter representation of the Schur algorithm (Fig. 8.1).

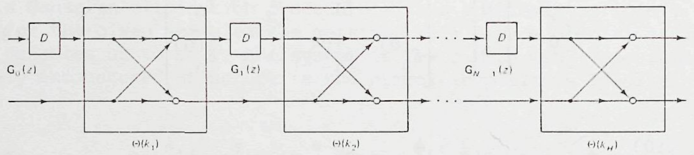


Figure 8.1. Lattice representation of the Schur algorithm.

Each section Θ_i performs a hyperbolic rotation as defined by Eq. (8.7), and for a Toeplitz matrix each section has only two input and two output connections. The input is the generator matrix G_0^*.

The Schur algorithm performs a decomposition of the matrix A and not of the matrix A^{-1} [Dewilde et al., 1978]. However, for stationary linear prediction problems and the realization of analysis and synthesis filters it is sufficient to have the reflection coefficients, and we do not need to compute the Cholesky factors of A^{-1}. Nevertheless, the inverse Cholesky factor L^{-1} can be easily computed with the lattice structure of Fig. 8.1. Once the $\{\rho_i\}$ have been computed, the lattice is inputted with $[1/\hat{a}_{00}\ 1/\hat{a}_{00}]$ (where $\hat{a}_{00}$ was defined in Eq. (8.6b)), and the rows of L^{-1} are obtained at the delay elements. Hyperbolic rotations can easily be implemented with CORDIC modules [Volder, 1959] [Walther, 1971] (see also Appendix E). Therefore, an architecture based on these modules will provide an efficient realization. However, if the rotations have to be implemented on a conventional signal processor (e.g. TMS320) the square root operations may be very cumbersome. By normalizing the initial vector Γ_{02} and Δ_0 to a_{00}, changing the factor $(1-\rho_i^2)^{\frac{1}{2}}$ in Eq. (8.6c) into $(1-\rho_i^2)$, and dropping the factor $(1-\rho_i^2)^{-\frac{1}{2}}$ in Eq. (8.6d), we obtain a non-normalized version of the Schur algorithm that still possesses good numerical properties [Rajasekaran and Hansens, 1982].

Thus far we have assumed the matrix A to be a Toeplitz matrix. It would be disappointing if the algorithms derived could not be used for matrices that are not Toeplitz. However, many matrices that arise from practical covariance approximation problems will have a structure "close" to Toeplitz. In [Kailath et al., 1979] the concept of displacement rank α (or α-stationarity) was introduced. The displacement rank α of a matrix A is defined as

$$\alpha = \text{rank}\{]A\} \quad , \text{with} \quad]A = A - ZAZ^* \qquad (8.11)$$

where the operator ']' is called a box operator and Z is the lower shift matrix (with ones on the first sub-diagonal and zeros elsewhere). It can be shown that it requires $O(\alpha n^2)$ operations to invert a matrix A with displacement rank α. For the solution of equation sets with displacement rank α we can use one of the procedures described in [Lev-Ari and Kailath, 1984]. In the next section we discuss how these procedures can be applied to solve the RPE coder equations.

8.3 Solution Of The RPE Equations

As described in Chapter 3 the RPE coder requires the solution of N=NS sets of linear equations of the form of Eq (8.1). The dimension of the matrices A_l (l=1,2,..,N) will be n+1 = NSSIZE/NS, i.e. the number of pulses within a search frame. In [Deprettere and Jainandunsing, 1985] an efficient procedure that inverts the N matrices with a complexity of $O((N+2)n^2)$ was developed. The matrix A_l is defined as

$$A_l = M_l HW(M_l HW)^* \qquad (8.12)$$

where the matrices M_l, H and W were defined in Section 3.3. The special structure of A_l will allow a simplified version of the procedures described in [Lev-Ari and Kailath, 1984] to be used. Furthermore, the matrices A_l, are very close to each other, in the sense that their rank distance, defined by

$$d = \text{rank}(A_{l+1} - A_l) \quad , l=1,N \qquad (8.13)$$

is equal to one.

8.3.1 *Fast Cholesky Decomposition of* A_l

The displacement rank of the matrix A_l is according to Eq. (8.11) equal to N+2. This means that by using the algorithm described in [Lev-Ari and Kailath, 1984] A_l can be inverted with $O((N+2)n^2)$ elementary operations. This algorithm computes L in a first "pass" and L^{-1} in a second one. In [Deprettere and Jainandunsing, 1985] it was shown that $]A_l$ can be factorized as

$$]A_l = G_l \Sigma_l G_l^*, \ l=1,\ldots,N, \qquad (8.14)$$

where G_l is the (N+2) by (n+1) matrix:

$$G_1^* = \begin{bmatrix} 0 & \hat{a}_{01}^{(1)^*} & \ldots & \hat{a}_{0n}^{(1)^*} \\ . & & & \\ . & & Q_1^* & \\ 0 & & & \\ \hat{a}_{00}^{(1)^*} & \hat{a}_{01}^{(1)^*} & \ldots & \hat{a}_{0n}^{(1)^*} \end{bmatrix} \qquad (8.15a)$$

with $\hat{a}_{0i}^{(1)} = \dfrac{a_{0i}^{(1)}}{(a_{00}^{(1)})^{\frac{1}{2}}}$, $i=0,1,\ldots,n$, and

$$Q_1^* = \{q_{ij}^{(1)} = h(L-i-jN-1+1) \mid 0 \le i \le (N-1),\ 0 \le j \le n-1)\} \qquad (8.15b)$$

$$\Sigma_1 = \Sigma = (-I_{N+1}) \oplus 1 \qquad (8.15c)$$

where I_{N+1} is the identity matrix of dimension N+1. The proof is obtained straightforwardly by direct calculation. When we drop the index "l" the computation of L^* is as follows:

Initialize:

$$G_0^* = \begin{bmatrix} \Gamma_0 \\ \hline \Delta_0 \end{bmatrix} = \begin{bmatrix} O_{(N+1)x1} & \gamma_1^{(0)} & \ldots & \gamma_n^{(0)} \\ \hline \delta_0^{(0)} & \delta_1^{(0)} & \ldots & \delta_n^{(0)} \end{bmatrix} = \begin{bmatrix} 0 & \hat{a}_{01}^* & \ldots & \hat{a}_{0n}^* \\ . & & Q^* & \\ 0 & & & \\ \hline \hat{a}_{00}^* & \hat{a}_{01}^* & \ldots & \hat{a}_{0n}^* \end{bmatrix} \qquad (8.16a)$$

For $i=1,\ldots,n$

$$G_i^* = \begin{bmatrix} \Gamma_i \\ \hline \Delta_i \end{bmatrix} = \begin{bmatrix} O_{(N+1)xi} & & \gamma_1^{(i)} & . & \gamma_n^{(i)} \\ \hline O_{1x(i-1)} & \delta_0^{(i)} & \delta_1^{(i)} & . & \delta_n^{(i)} \end{bmatrix} = \Theta_i \begin{bmatrix} I_{N+1} & 0 \\ 0 & z^{-1} \end{bmatrix} \begin{bmatrix} \Gamma_{i-1} \\ \hline \Delta_{i-1} \end{bmatrix} \qquad (8.16b)$$

where

$$\Theta_i = \begin{bmatrix} (I_{N+1}-\rho_i^*\rho_i)^{-\frac{1}{2}} & 0 \\ 0 & (1-\rho_i\rho_i^*)^{-\frac{1}{2}} \end{bmatrix} \begin{bmatrix} I_{N+1} & -\rho_i^* \\ -\rho_i & 1 \end{bmatrix} \qquad (8.16c)$$

and

$$\rho_i^* = \gamma_1^{(i-1)} / \delta_0^{(i-1)} \qquad (8.16d)$$

From the positivity of A it follows that $|\rho|^2 \le 1$. This procedure is shown in Fig. 8.2, where the Δ_i $(i=0,1,\ldots,n)$ are the rows of L^*.

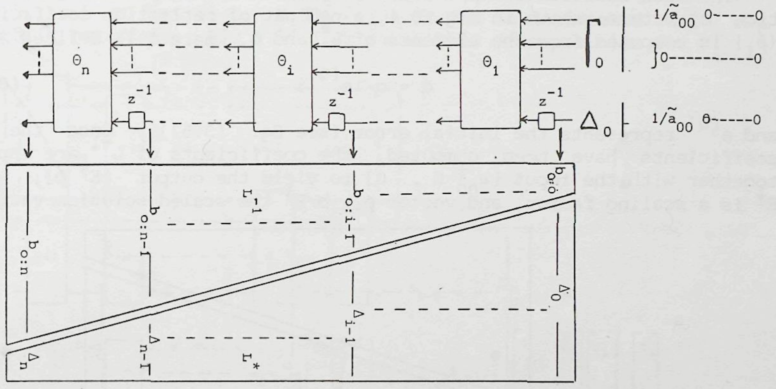


Figure 8.2. Lattice structure for Cholesky factorization of $A=LL^*$ and the computation of L^{-1}.

L^{-1} is computed as follows: Let $\nu = [-1/\hat{a}_{00},0,\ldots,0,1/\hat{a}_{00}]$; then $\nu G = [1\ 0\ldots.0]$, and

For i=1,...,n

$$\begin{bmatrix} C_{0:i} \\ b_{0:i} \end{bmatrix} = \Theta_i \begin{bmatrix} I_{N+1} & 0 \\ 0 & z^{-1} \end{bmatrix} \begin{bmatrix} C_{0:i-1} \\ b_{0:i-1} \end{bmatrix} \qquad (8.17)$$

with

$$\begin{bmatrix} C_{0:0} \\ \\ \\ \\ b_{0:0} \end{bmatrix} = \Sigma\nu' = \begin{bmatrix} 1/\hat{a}_{00} \\ 0 \\ . \\ 0 \\ 1/\hat{a}_{00} \end{bmatrix} \qquad (8.18)$$

The $b_{0:i}$ $(i=0,1,\ldots,n)$ are the rows of L^{-1} (see Fig. 8.2) We refer to [Lev-Ari and Kailath, 1984] for a proof of this result.

8.3.2 *Back Substitution*

The back-substitution procedure is done in two steps. Using the lattice structure shown in Fig. 8.4, a new set of reflection coefficients $\{\hat{\rho}_i\}$ is computed from the elements of L^* and $\hat{q}$, where $\hat{q}$ is defined as

$$\hat{q} = q/|e^{(0)}| \qquad (8.19)$$

and $e^{(0)}$ represents the initial error (see Eq. (3.5)). Once the new coefficients have been computed, the coefficients of L^{-1} are applied together with the input $[a_{00}^{-\frac{1}{2}}\ 0\ ..\ 0]$ to yield the output $[E^{\frac{1}{2}}\ \hat{p}]$, where $E^{\frac{1}{2}}$ is a scaling factor, and vector $\hat{p} = p/E^{\frac{1}{2}}$ the scaled solution vector.

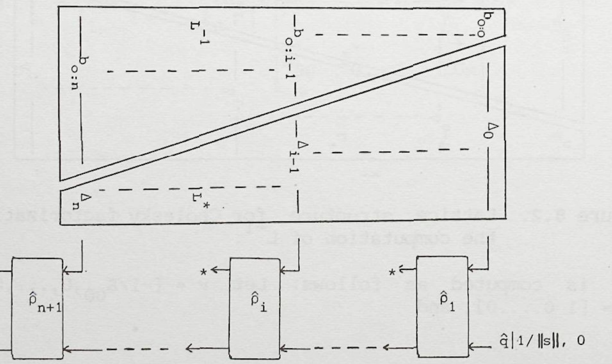


Figure 8.3. Lattice structure for the back-substitution procedure.

Note that the scaling factor $E^{\frac{1}{2}}$ is nothing else than the square root of the error $E^{(I)}$ (see Eq. (3.10)).

8.3.3 *Decomposition of* Θ_i

Fig. 8.4 shows the complete structure for Cholesky factorization and back substitution. The matrix Θ_i [Eq. (8.16c)] can be written as a product of elementary rotations via a process similar to diagonalization by means of Givens rotations.

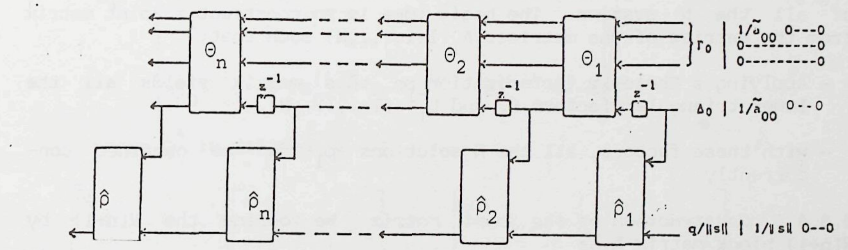
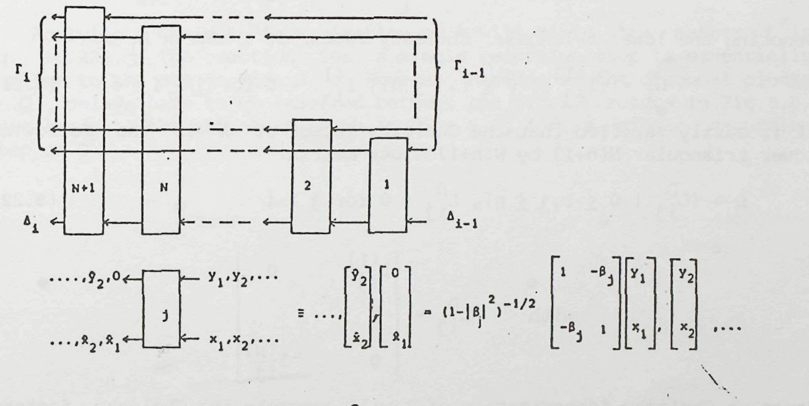


Figure 8.4. Combined cascade for the Cholesky decomposition and back substitution.

Since Θ_i is Σ-unitary (i.e. $\Theta_i\Sigma\Theta_i^* = \Sigma$), it can be decomposed into N+1 hyperbolic rotations. Fig 8.5 illustrates this decomposition for Θ_i



Γ_i
Γ_{i-1}
N+1
N
2
1
Δ_i
Δ_{i-1}
$\ldots,\hat{y}_2,0$
$y_1,y_2,\ldots$
j
$\ldots,\hat{x}_2,\hat{x}_1$
$x_1,x_2,\ldots$
$\begin{bmatrix}\hat{y}_2\\\hat{x}_2\end{bmatrix}\begin{bmatrix}0\\\hat{x}_1\end{bmatrix} = (1-|\beta_j|^2)^{-1/2}\begin{bmatrix}1 & -\beta_j\\-\beta_j & 1\end{bmatrix}\begin{bmatrix}y_1\\x_1\end{bmatrix},\begin{bmatrix}y_2\\x_2\end{bmatrix}\ldots$

Figure 8.5. Decomposition of Θ_i into a cascade of hyperbolic plane rotations.

It was noted that the basic lattice structure is based on (hyperbolic) rotations. These rotations can be realized very efficiently with CORDIC processors [Volder, 1959] [Walther, 1971] (see also Appendix E). In [Deprettere et al., 1984] it was shown that even pipeline CORDIC structures can be constructed.

8.3.4 *The Concurrent Solution Of A Set Of N Equations*

In the previous section we have shown how to apply the algorithm of Lev-Ari & Kailath for solving one system in Eq. (8.1). The next step is to show how this algorithm can be extended for the simultaneous solution of all the N systems. The basic idea is to construct a joint matrix from the entries of the matrices $A_l (l=1,\dots,N)$ such that:

- applying a Cholesky factorization on this matrix yields all the lower triangular factors L_l^* and L_l^{-1} $(l=1,\dots,N)$.
- with these factors, all the N solutions p_l can be obtained concurrently.

8.3.4.1 *Construction of the joint matrix* We define the $N(n+1)$ by $N(n+1)$ block matrix **A** as

$$\mathbf{A} = \{A^D_{ij} \mid 0 \le i,j \le n\} \tag{8.20}$$

$$\text{with} \quad A^D_{ij} = \begin{bmatrix} a^{(1)}_{ij} & & 0 \\ & \cdot & \\ 0 & & a^{(N)}_{ij} \end{bmatrix}$$

Denoting the lower triangular Cholesky factor of a matrix A_l by

$$L_l = \{l^{(l)}_{ij} \mid 0 \le i,j \le n\},\ l^{(l)}_{ij} = 0 \text{ for } j > i \tag{8.21}$$

it is easily verified that the Cholesky factor of **A** is the following lower triangular $N(n+1)$ by $N(n+1)$ block matrix:

$$\mathbf{L} = \{L^D_{ij} \mid 0 \le i,j \le n\},\ L^D_{ij} = 0 \text{ for } j > i \tag{8.22}$$

$$\text{with} \quad L^D_{ij} = \begin{bmatrix} l^{(1)}_{ij} & & 0 \\ & \cdot & \\ 0 & & l^{(N)}_{ij} \end{bmatrix}$$

Hence, a Cholesky factorization of **A** will generate the Cholesky factors of all A_l.

8.3.4.2 *Cholesky Factorization And Back Substitution.* Analogous to the $G_l \Sigma G_l^*$ decomposition of a matrix A_l, we can find a $\mathbf{G}\Sigma\mathbf{G}^*$ decomposition of **A**, in order to obtain $\mathbf{L}^*$ and $\mathbf{L}^{-1}$ from **G**. In [Deprettere and Jainandunsing, 1985] it was derived that the following decomposition holds:

(i) the matrix $\mathbf{G}^*$ is the following $(N+2)$ by $N(n+1)$ matrix:

$$\mathbf{G}_* = \begin{bmatrix} 0 & 0\text{—}0 & a^{(1)*}_{01} & 0\text{—}0 & . & a^{(1)*}_{0n} & 0\text{—}0 \\ 0 & & & & & & \\ | & & & & Q^* & & \\ 0 & & & & & & \\ a^{(1)*}_{00} & 0\text{—}0 & a^{(1)*}_{01} & 0\text{—}0 & . & a^{(1)*}_{0n} & 0\text{—}0 \end{bmatrix} \tag{8.23}$$

where Q^* is defined as

$$Q^* = \left[\begin{bmatrix} \dot{h}(L) & & & 0 \\ & \dot{h}(L-1) & & \\ & & \ddots & \\ 0 & & & \dot{h}(L-N+1) \end{bmatrix} \begin{bmatrix} \dot{h}(L-N) & & & 0 \\ & \dot{h}(L-N-1) & & \\ & & \ddots & \\ 0 & & & \dot{h}(L-2N+1) \end{bmatrix} \cdots\cdots \begin{bmatrix} \dot{h}(2N) & & & 0 \\ & \dot{h}(2N-1) & & \\ & & \ddots & \\ 0 & & & \dot{h}(N+1) \end{bmatrix} \begin{bmatrix} \dot{h}(N) & & & 0 \\ & \dot{h}(N-1) & & \\ & & \ddots & \\ 0 & & & \dot{h}(2) \\ & & & 0 \end{bmatrix} \right] \tag{8.24}$$

(ii) the matrix Σ has rank $(N+2)$ and is of the form

$$\Sigma = (-I_{N+1}) \oplus 1. \tag{8.25}$$

Applying a Cholesky factorization on **A** will yield the matrix $\mathbf{L}^*$ in Eq. (8.22). The section for a single recursion step is essentially similar to the one in Fig. 8.5. However, because of the diagonal blocks in Q^*, delays have to be inserted between the first N rotors in Fig. 8.5. This leads to the structure shown in Fig. 8.6, for a single recursion step on $\mathbf{G}^*$.

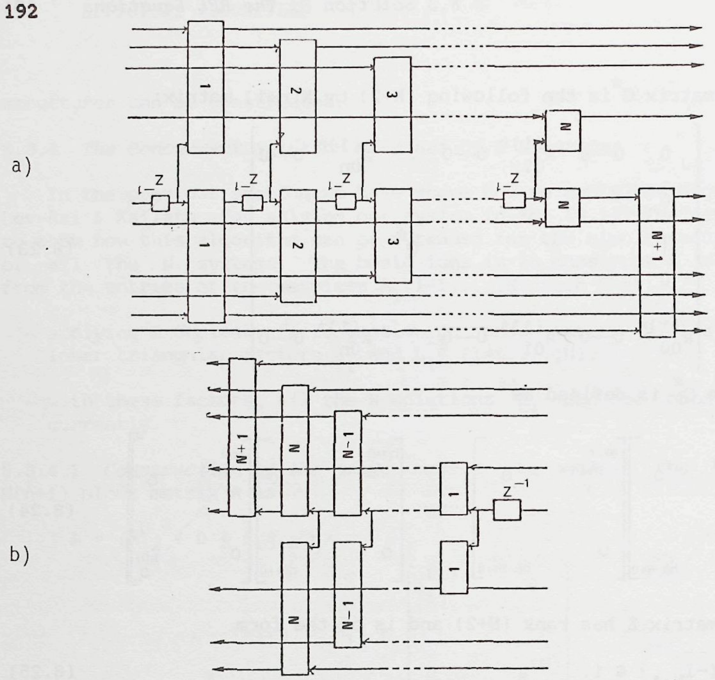


Figure 8.6. *(a) Section for a single recursion step on G^*; (b) section for a recursion step on G_c^*.*

The rows of the L_1 matrices, which are required for the computation of the N solution vectors p_1, appear at the taps. Fig. 8.6(a) can be simplified to Fig. 8.6(b) by making the following two observations.

1. The delay elements - separating the rotors in a section - can be deleted, without affecting the final result. Deleting the delays corresponds to compressing each of the n blocks of N columns (starting the count from the second column) of G^* including the *remaining block of (N-1) columns, into a single column.*

2. It is allowed to permute the rows of Q^* (and the corresponding columns of Q) since the product QQ^* will not change.

From 1) and 2), reversing the row order of Q and compressing G^* leads to the following equivalent form of G^*:

$$G_c^* = \begin{bmatrix} 0 & \hat{a}_{01}^{(1)*} & .\ \hat{a}_{0n}^{(1)*} & 0 \\ 0 & h^*(L-N+1) & .\ h^*(N+1) & 0 \\ . & . & .. & h^*(2) \\ 0 & h^*(L) & h^*(2N) & h^*(N) \\ \hat{a}_{00}^{(1)*} & \hat{a}_{01}^{(1)*} & \hat{a}_{0n}^{(1)*} & 0 \end{bmatrix} \qquad (8.26)$$

resulting in the structure of Fig.8.6(b). The complete cascade is shown in Fig. 8.7 with the matrix G_c^* as the input data for the Cholesky factorization of A and the matrix $\hat{D}$:

$$D = \begin{bmatrix} 0\text{——} & \hat{q}_1 & \text{——} \\ & . & \\ 0\text{——} & \hat{q}_N & \text{——} \end{bmatrix} \qquad (8.27)$$

for the calculation of the $\{\hat{\beta}_i\}_l$ $(l=1,\dots,N)$.

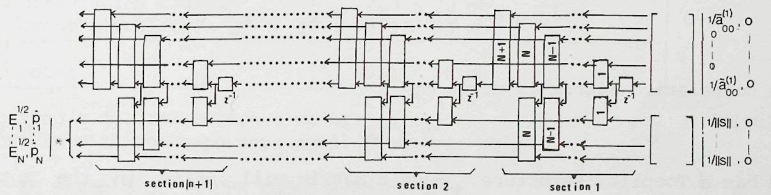


Figure 8.7. Complete cascade for the inversion of A and the computation of the N solutions $p_1,\dots,p_N$.

To obtain all solutions p_l, $l=1,2,\dots,N$, the vector

$$\underbrace{[1/\hat{a}_{00}^{(1)}\ 0\ \dots\ 0\ 1/\hat{a}_{00}^{(1)}}_{N+2}\ \underbrace{1/|e^{(0)}|\ \dots 1/|e^{(0)}|]}_{N} \qquad (8.28)$$

is applied at the input of the cascade of Fig. 8.7. Observe that this cascade is very similar to the one in Fig. 8.4. The only increase in complexity has been due to the addition of the back-substitution structure for $p_2,\dots,p_N$. The total complexity is therefore almost twice, instead of N times, the complexity for a single system of equations. The algorithm exhibits pipelining on several levels. For example, the pipeline can be implemented on the cascade level, but a hyperbolic rotation can also be implemented in a pipeline fashion [Deprettere et al., 1984]. Parallelism is also exhibited in the sense that Cholesky decomposition and the back substitution are done in parallel. In [Deprettere and Jainandunsing, 1985] a pipeline CORDIC structure that implements the algorithm described above was proposed. The configuration consists of two pipelined CORDIC processors and implements the pipelining on the rotor level with hyperbolic rotations. The CORDIC pipe consists of 13 stages and implements one macro hyperbolic rotation. The chip requires approximately 6.5x6.5 mm^2 in a 4 μ NMOS process.

8.4 Correlation Computations

In the previous sections we described efficient structures for the solution of a set of equations of the form of Eq. (8.1). In this section we examine how these equations are constructed, as well as the computational effort required for this construction. The matrix A and the vector q are both a result of correlation computations. Given a data set $\{x_1, x_2, \ldots, x_L\}$, and a corresponding (n+1) by 2L matrix

$$X = \left[\begin{array}{ccccc:cccc} x_1 & x_2 & x_3 & . & x_L & 0 & . & 0 & 0 \\ 0 & x_1 & x_2 & . & x_{L-1} & x_L & . & 0 & 0 \\ . & . & . & . & . & . & . & 0 & 0 \\ 0 & 0 & 0 & . & x_1 & x_2 & . & x_L & 0 \end{array}\right] \qquad (8.29)$$

the correlation matrix is given by

$$A = XX^* \qquad (8.30)$$

and has a Toeplitz structure. Such a matrix will arise in the autocorrelation procedure for determining the filter coefficients. If the matrix of Eq. (8.30) is reduced to an (n+1) by L matrix by dropping the elements on the right-hand side of the dashed line, the resulting correlation matrix will be symmetric but not Toeplitz. Such a matrix structure is encountered in the RPE equations. The vector x is then equal to the impulse response of the cascade of the synthesis filter and error weighting filter. The vector q is in the RPE method the result of a cross-correlation between the initial error $e^{(0)}$ and the product matrix $M_l HW$

$$q_l = e^{(0)}(M_l HW)^* \qquad l=1,..,N \qquad (8.31)$$

Note that by computing the product $e^{(0)}(HW)^*$, all the N vectors q are available at once. Suppose that a processor unit that performs a multiply and add operation in one step is available. The number of operations required for the construction of an (n+1) by (n+1) Toeplitz correlation matrix is given by

$$\tfrac{1}{2}(n+1)(2L-n) \qquad (8.32)$$

The elements of the first row of this matrix are defined by

$$c(j) = a_{0j} = a_{j0} = \sum_{k=1}^{L-j} x(k)x(k+j) \qquad j=0,..,n \qquad (8.33)$$

The computation of the non-Toeplitz matrix requires the same number of operations. The truth of this can be seen by recognizing the fact that the elements on a diagonal are formed by the partial sums of Eq. (8.33). Note that for the RPE coder only the correlation coefficients c(0), c(NS), ..,C(nNS) are required, resulting in a total number of operations

$$\tfrac{1}{2}(n+1)(n+2)NS \qquad (8.34)$$

The computation of the cross-correlation of Eq. (8.31) requires the same number of operations as the computation of the correlation matrix. To get an estimate of the required multiplication time we consider the RPE coder using 10 pulses/5 ms and a 12-th order filter. We assume that the coefficients are determined from 25 ms frames and are updated every 20 ms:

matrix	# operations /time interval
auto-correlation matrix	2522 / 20 ms
correlation matrix	220 / 20 ms
cross-correlation	820 / 5 ms
total	6022 / 5 ms

requiring a multiplication time of 3.4 μs, which is not very difficult to obtain with contemporary signal processor chips.

8.5 An example of an RPE coder architecture

In the previous sections we found that by decomposing the basic algorithms into elementary hyperbolic operations, the resulting structure could be easily implemented with CORDIC modules. The fact that the lattice analysis filter A(z) and lattice synthesis filter 1/A(z) can also be expressed in terms of hyperbolic and circular rotations, respectively, will make the use of CORDIC modules even more attractive. Correlation computations are more efficiently implemented with a processor that enables a fast execution of multiply-add operations. A general purpose signal processor (e.g TMS 320) will be very suitable. Such a processor will also be useful for tasks such as selecting and encoding the excitation structure. Provided that the CORDIC chips are able to perform the LPC analysis, the filter operations and the RPE analysis in a sequential manner, and that the signal processor performs all the correlation computations, the whole coder consists, in addition to external memory, of only two processor chips.

8.6 Summary

In this chapter we considered some efficient algorithms for solving equations of the RPE coder. It was shown that an algorithm can be found that has a lot of parallelism, thereby enabling efficient structures with a high throughput. An example was given of the decomposition of the RPE algorithm into hyperbolic rotations, such that the coder can be implemented with a pipelined CORDIC architecture. An examination of the number of correlation computations revealed that a separate processor is required to perform these computations. Based on these results it was estimated that a complete RPE coder can be implemented with a special purpose CORDIC chip, a general purpose signal processor and some additional memory.

References

Deprettere, E.F., P. Dewilde, and R. Udo, "Pipelined cordic architectures for fast VLSI filtering and array processing," *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing,* pp. 41A6.1-41A6.4 (March 1984).

Deprettere, Ed.F. and K. Jainandunsing, "Design and VLSI implementation of a concurrent solver for N-coupled least-squares fitting problems," *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing,* pp. 6.3.1-6.3.4 (March 1985).

Dewilde, P., A. Viera, and T. Kailath, "On a generalized Szego-Levinson realization algorithm for optimal linear predictors based on a network synthesis approach," *IEEE Trans. Circuits Syst.* Vol. **CAS-25** pp. 663-675 (Sep. 1978).

Kailath, T., S.Y. Kung, and M. Morf, "Displacement ranks of matrices and linear equations," *J. of Mathematical Analysis and applications* Vol. **68**(2) pp. 395-407 (April 1979).

Kung, S.Y. and Y.H. Hu, "A highly concurrent algorithm and pipelined architecture for solving Toeplitz systems," *IEEE Trans. Acoust., Speech, Signal Processing* Vol. **ASSP-31**(1) pp. 66-76 (Feb. 1983).

Kung, S.Y., H.J. Whitehouse, and T. Kailath, *VLSI and modern signal processing*, Prentice Hall, New Jersey (1985).

LeRoux, J. and C. Gueguen, "A fixed point computation of partial correlation coefficients," *IEEE Trans. Acoust., Speech, Signal Processing* Vol. **ASSP-27** pp. 257-259 (1979).

Lev-Ari, H. and T. Kailath, "Lattice filter parametrization and modeling of nonstationary processes," *IEEE Trans. Inf. Theory* Vol. **IT-30**(1) pp. 2-16 (Jan. 1984).

Levinson, N., "The Wiener RMS (root-mean-square) error criterion in filter design and prediction," *J. Math. Phys.* Vol. **25** pp. 261-278 (Jan. 1947).

Rajasekaran, P.K. and J.C. Hansens, "Finite word length effects of the LeRoux-Gueguen algorithm in computing reflection coefficients," *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing,* pp. 1286-1290 (April 1982).

Schur, J., "Uber potenzreihen, die in Innern des einheitskreises beschrankt sind," *J. fur die Reine and Angewandte Mathematiek* Vol. **147** pp. 205-232 (1917).

Volder, J.E., "The CORDIC trigonometric computing technique," *IRE Trans. Electronic Computers* Vol. **EC-8** pp. 330-334 (Sep. 1959).

Walther, J.S., "A unified algorithm for elementary functions," *Proc. of the 1971 Spring Joint Computer Conference,* pp. 379-385 (1971).

9. A SOFTWARE ENVIRONMENT FOR SIGNAL PROCESSING

9.1 Introduction

Signal processing research, and more specifically, speech and image processing research, comprises many computer-based activities. Two major activities are the development of algorithmic solutions to problems and the simulation of these solutions using real-world data. During the algorithm development stage, conceptual systems that usually originate from both heuristic and mathematically founded roots are tried out. Once a conceptual algorithm has been defined, this structure is translated into an algorithmic structure suitable for VLSI implementation. During the simulation stage, both the conceptual and the practical systems are submitted to stringent performance studies using a large set of test data. To be able to concentrate on research activities rather than on software development, it is crucial to have a suitable software environment. The complexity of such an environment can vary from very simple (e.g. the availability of a small library with basic procedures), to very complex (e.g. an integrated software and hardware environment).

Historically, signal processing software design has not been of major concern. Of course, computer code for classical signal processing algorithms, such as filter design programs and Fourier transform procedures, can be found in literature, but a collection of such "callable" functions as can be found in [IEEE Press, 1979] does not provide a software environment for signal processing. It was no earlier than 1980 when the first "portable" commercial software package for signal processing (ILS) became available [Signal Technology, 1980]. ILS is an interactive environment consisting of many modules that each performs a single task. By combining these basic tasks more complicated operations can be performed. The communication between the different programs is maintained by a common memory area. A few years later similar packages appeared such as DT-DATS [Data Translation, 1982] and I*S*P [Bedford Research, 1983]. Obviously, many non-commercial packages have been developed, but these are in most cases difficult to obtain and suffer from a lack of support. All the packages mentioned will run on commonly used computer systems and operating systems without the requirement for special hardware (with the exception of data-acquisition equipment and graphic display systems). With the advance of sophisticated work stations using bit-mapped graphic displays and user-interfaces such as mouses, a more integrated approach becomes feasible. A good example of the latter is the Integrated Signal Processing System (ISP) [Kopec, 1984].

During the early stage of our research work, we recognized the need for a suitable software environment, and we started to develop our own package called SPPACK [Kroon, 1983] [Kroon, 1983]. At that time we were unaware of the existence of other systems, but later it became clear that the basic philosophy of our system was not really different from that used in other packages. Moreover, we find that the SPPACK system is better suited to our needs and has the advantage that it can easily be modified to provide further support of our research work. In this chapter we describe the general considerations used on the design of such an interactive system for signal processing and we describe how the desired features have been incorporated in SPPACK.

9.2 Design Considerations For A Software Package

As we stated in the introduction the package must provide a powerful experimental environment to allow the user to develop new programs, and to do preliminary experiments. This requires an interactive environment. On the other hand, once an analysis procedure has been established, simulation runs using a large collection of test data will require a batch mode. Besides these general characteristics, many other features such as portability and user friendliness are desirable. A portable system means that the package can be run on different computer or operating systems with little or no modification. The term "user friendliness" is a little vague and can be interpreted in many ways. In our opinion it means a system that can be easily used by both novice and expert users. The command language used for controlling the jobs should be easy to learn and use. On-line help should be available and additional documentation has to be provided. Further, an extensive error detection mechanism is required, which checks for illegal user responses and actions, and enables the users *to correct his or her mistakes*. Another important requirement is extensibility. The package should comprise many basic operations such as data acquisition, data manipulation, graphics etc. However, some operations are not available and have to be supplied by the user. If this task can be accomplished in a simple way, the package will become more valuable and will even suit future users.

9.3 Description Of SPPACK

The signal processing package SPPACK is a modular file oriented *interactive software system*. The package provides many basic signal processing tasks such as data acquisition, digital filter design, signal transformation, data manipulation etc. Each of these elementary functions in SPPACK is implemented as a single program, referred to as "module", which carries out only one definable function. Each SPPACK module produces output in a form suitable for use by other modules. Each module can be used separately, with interactive parameter selection by the user. Based on the results of the previous steps, the user may invoke other modules or commands with the parameters appropriately set. In this manner the system is capable of supporting a high level of man-machine interaction in signal processing. More complicated tasks can be performed by combining a set of modules in a user selected order. The program parameters will then be given by interaction or a control program. A simple control command language is provided to "glue" the modules together into a sequence of modules, executing a complicated task. This building block approach allows natural development of complex signal processing functions and programs. Most input and output

data is stored in standard disk files. Sometimes other devices such as the user terminal, AD and DA converters, or graphic displays are used as input/output device. Data files play an important part in the SPPACK system. Each file consists of a file header followed by the actual data. The file header contains information about the file contents such as data type and the procedure used for generation of the data. This header can be read and/or modified by both user and programs, thereby providing a method for inter-program communication. Each module or program can be invoked in several ways. Without any additional parameters the module will operate in the query mode, and will prompt the user for options. This mode is intended for inexperienced users. More experienced users can specify additional parameters to bypass the prompting mode. In both cases an extended error checking and recovery mechanism is active to protect the user against illegal procedures. To improve the accessibility of the system, it is to some extent integrated in the host operating system. For example, the SPPACK data files can usually be treated as any other file of the host system. This means that many standard file utilities such as copying a file or deleting a file can be used directly. Most operating systems provide some control language which can be used for "gluing" modules together. With this approach SPPACK may somewhat differ on different operating systems, but new SPPACK users will become familiar with the package more easily. The system was developed under the RSX-11M* operating systems and is almost completely written in the Fortran language. Versions that operate under RT-11*, VMS* and UNIX** are available but in our examples we stick to the RSX-11M version. In the following sub-sections we describe typical features of the system, and some of the design considerations. Finally, we give some examples. For a more detailed descriptions of the available modules the reader is referred to the SPPACK manuals [Kroon, 1983] [Kroon, 1983].

9.4 Command Format And Processing

As explained above, SPPACK consists of many modules that perform a specific operation. Each module is actually an independent program that runs under control of the operating system. The name of a module resembles as much as possible the operations that it performs. A module is invoked by specifying its name. For example, to perform a Fast Fourier Transform, the command FFT has to be issued. Most commands require the specification of additional parameters such as the data to be acted upon and optional modifications to the basic command. For example, the FFT command will require the specification of the input data and a

---

* RSX-11M, RT-11 and VMS are trademarks of the Digital Equipment Corporation.

** UNIX is a trademark of Bell Laboratories.

specification of the location in which to store the output data. Additional parameters define the size and the type of the data window to be used. The complete command can consist of the following items:

1. module name,
2. range specification,
3. options,
4. file specifications.

Only the first item is mandatory for each SPPACK command.

To accommodate both experienced and unexperienced users the following approach has been used. The most simple procedure for invoking a program is to specify its name. The program will then be in the prompting mode and will prompt the user to supply specific information. Each query also displays its default response, which will be used if the user responds with a carriage return <CR>. The defaults are selected to reflect the normal use of the program. The most efficient way to invoke a program, however, is to use the single line format. In this mode all command attributes such as parameters and file names are specified on one single line. The program will then start execution immediately without prompting. This single line format is also used when complex programs are built from many basic modules. Intermediate forms between the prompting format and the single line format are also possible. In that case a user can invoke a module by typing its name and file specifications on a single line and the program prompts for the missing specifications. All programs will prompt for the non-specified attributes rather than use the default responses. This approach ensures that the user always has to confirm the actions he or she wants to take. However, by the use of a DEFAULT option it is still possible to specify within the single line format the use of defaults.

From the description above it is hopefully be clear that a "module", a "program" and a "command" are synonyms. However, there are some exceptions. Most modules perform one single task and return to the monitor after execution. In this case a module is equivalent to a program. We call these programs 1-level programs. Sometimes, it is quite impractical to have a separate program for each task. For example, a graphical display program plots data. The basic plot command can be modified by options such as drawing mode, pen color or scaling factor. However, associated interactive commands such as drawing axes or displaying text are more easily incorporated as sub-commands than as separate programs. Another aspect is that these programs require a lot of user interaction, and it would be very inconvenient if the user had to specify the same command many times. For these kinds of programs we introduce the concept of 2-level programs. A 2-level program is invoked as a 1-level program, but rather than returning to the monitor after execution, it remains in the program and accepts commands on level 2.

We said above that complex programs can be built from basic modules by the use of a control language. Most operating systems provide such a control language (e.g. command file processor for DEC-systems, shell procedures for UNIX systems). The simplest use of control languages is to use them to specify a set of single line commands and store those lines in a control file. Upon execution of the control file, the commands are executed in the specified order. This approach works well for all 1-level commands. However, to be able to use a similar procedure for commands that have to be issued at the second level of a 2-level program, we used a mechanism called CDF (CommanD File). This mechanism allows the user to store level-2 commands in a file and execute them automatically when desired. Fig. 9.1 shows the different ways of interaction.

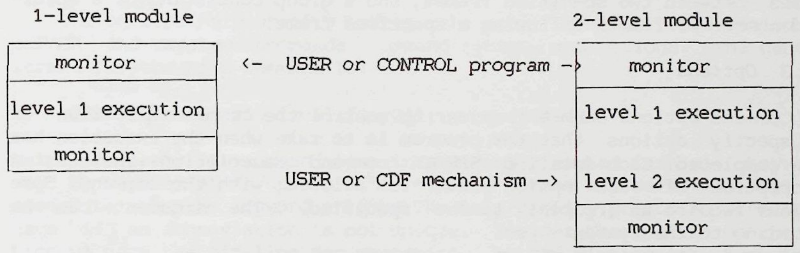


Figure 9.1. Interaction between user and 1- and 2-level programs.

A typical example of a single line command is:

```
FFT    1:5    /DEF/WINDOW:2/SIZE:128 P1 P2

name  range       options         files
```

As can be seen from this example the command and option names are English words that suggest the intended action. For the option words it is not necessary to type the full name. Any abbreviation that is not ambiguous will be accepted. For a description of the complete command syntax, the reader is referred to the SPPACK manuals.

9.4.1 *Input/Output Files*

"Input file" represents the file on which the action is to be taken. "Output file" represents the file that is to receive the results of the operation. If the file types are not specified the program defaults to an appropriate type. If only one file is specified while two files are expected the program assumes this one to be the input file.

9.4.2 *Range Specification*

For some commands, it is required to specify what portion of the file to use. This portion is called the "range". A range can be as large as an entire file or as small as a single frame, where a frame is a context or user defined number of adjacent data elements. There are three general types of range specifications: single, variable and compound range specifications. Single range specifications allow the specification of one frame of data. Variable range specifications allow the specification of an indeterminate number of contiguous frames. Examples are: all frames in the current file, and all frames between the current frame and the end of the file. Compound range specifications allow for the specification of a group of adjacent frames. Examples are: all the frames between two specified frames, and a group consisting of a specified number of frames following a specified frame.

9.4.3 *Options*

Command options enable the user to control the command execution or to specify actions that the program is to take when the execution has been completed. Each detailed SPPACK command description includes a description of each option that is allowed with the command. Some options require an argument to be specified. The argument can be according to the syntax:

```
an integer argument       e.g.    /option:32767
                                  /option:3
                                  /option:-172

a number argument         e.g     /option:173
                                  /option:173.
                                  /option:17.3E1

a string argument         e.g.    /option:text
(maximally 4 characters)          /option:ab1
```

9.5 **Error Checking And Recovery**

We stated in the beginning of this chapter that it is important to have an extended error checking and recovery system in each module. The simplest form of error detection is to check the user input for spelling and syntactical errors, and to produce appropriate diagnostic messages. Such a detection mechanism should be active during all times the user could possibly make an error. Once an error has been detected, facilities should be provided to correct this error. It is important that the error should be detected directly after its occurrence. It is very annoying to answer many prompts, and to have a message about an error in the first input line not appear until the last question. Error detection mechanisms should also detect, as far as possible, logical errors such as conflicting responses to options. The file header system allows additional logical checking. For example, the program FFT can check for a time-frequency transform, if the input file really contains time domain data. As a result the system will detect many possible errors and will warn the user at the time of the detection. A different form of error recovery is required during program execution. For example, execution errors such as data overflow or divide by zero will on most systems cause a program dump. In an environment that uses compiled code, continuation of a dumped program is difficult or impossible. However, to some extent it is possible to intercept dumps, and display an appropriate message. A difficulty is that these procedures might involve many operating system features, thereby reducing the portability of the system.

9.6 **Help Utility And Documentation**

The on-line help utility enables the user to list information on SPPACK and system commands, command syntax, and options. The general format for the help command is:

```
HELP [qualifier1] [qualifier2] [... qualifier9]
```

Help interprets its command line in a strictly nested fashion. Keywords may be abbreviated up to the minimum number of characters that are needed to make the keyword unique. The help utility prints an error message if an abbreviation is not unique. Besides the on-line documentation we have the off-line documentation. We can distinguish two forms of documentation:

1. Inter-program documentation, such as comments.

2. Documentation separated from the program itself.

Here we will consider only the first category. In this category program code and comments are stored in the same file, which will improve maintenance. Standards have been defined for the layout of the code and the use of comments. A set of names has been adopted for variables that are frequently used throughout the system, and the use of these names is mandatory so that there may be consistency among programs. This also serves the purpose of aiding programmers with software development and maintenance. A person familiar with the conventional names has an easier time reading programs. According to our conventions every program, subroutine or function begins with a number of comment lines, which we call the (program) header. The standard layout of this header improves readability and makes it possible to use software tools for documentation (e.g. automatic change of modification date or extraction of help files). The general structure of a program header is as follows:

1. Identification part,

2. Brief functional description,

3. Global and common parameter description,
4. List of external files and sub-programs,
5. Usage,
6. Process description.

The *identification part* lists attributes such as program name, programmer's name, modification date, implementation language, and host system. *Usage* explains the use of the program and this part will also be used by the help utility. *Process Description* gives additional details about the algorithm in plain text or an appropriate description language.

9.7 Data File Structures And Types

SPPACK distinguishes two general classes of files:

1. Text files,
2. Unformatted binary data files.

The first class of files contains readable text and can be created and modified by any conventional text editor. Files that belong to this class are: command files, directory files, help files and module description files.

Unformatted binary data files are used for storage of input and output data of the SPPACK programs. One of the main features of SPPACK is the high degree of standardization of data file structures. In the next sections the organization of the data files will be discussed in more detail.

9.7.1 *Data Files*

Data files are unformatted binary data files, each with a 256 (16 bit) word file header containing information that is global to the file. Conventions have been set down as to what gets stored in the header and where it is located. There is no restriction on the length of the file other than that it must fit on the disk. To access only certain portions (frames) of the data, software has been implemented to provide for data access in terms of variable length records referred to as "frames". The length of a frame can be defined by the user according to what is convenient for the application at hand. The location of data in a data file can now be specified in terms of its starting frame and number of frames (duration). Notice that for each frame length, frame 1 always refers to the beginning of the data immediately after the header.

9.7.2 *File Header Format*

The file header consists of 256 words, located at the beginning of a data file. The header contents serve as identification for the stored data. To cover a wide range of data types and their characteristic features, SPPACK provides the concept of header modules. The header consists of a fixed part containing global data features and a variable part (module) containing characteristic features of the stored data. These modules can be user defined to allow a maximum of flexibility for new applications. The global layout of a header is

```
Fixed header part (75% of the header space)
  text strings
  command line
  modification date
  statistical information
  file identification
  general data description

Modular header part (25% of the header space)
```

The first part of the header contains three *text strings* of 80 characters each. These strings can be filled by the user and could contain descriptive information about the data stored in the file. The contents of these text strings can be read and changed with the utility program HEADER.

Every program that creates or modifies a data file will write its *command line* in the data file header. The command will be stored as a single line command even in case it was issued in the prompting format. If the input file header already contained a command line, this line will be copied to the text string part before writing the new command line to the file. This facility enables the user to examine the "history" of the file up to a maximum of four command lines.

Usually, the operating system records the creation date of a data file. However, modifying the data within an existing file will not affect this creation date. To be able to find out the last time a modification has been made to the stored data, the header contains the last *modification date*.

Statistical information will be written into the header by SPPACK's statistical programs. This information comprises the maximum and minimum values and their corresponding locations within the file, average value and variance and the number of 256 word blocks used for computing these parameters. The information will be used by other programs such as plot modules.

General data descriptors will give information about the data domain (e.g. time or frequency domain), and the sampling frequency in Hz.

9.7.3 *File Types*

SPPACK defines different file types for different applications. For example the file type of a file storing filter coefficients will be different from the file type of a file used for the storage of sampled data. Before we describe the existing file types, we make a division between different data types. SPPACK supports four different data types.

1. INTEGER

2. REAL

3. COMPLEX

4. DOUBLE PRECISION

File types are represented by a negative integer number in the range -32000:-28000. Every data type is mapped into a small area of this range, according to table 9.1.

data type	file type range	user definable range
INTEGER	-28999:-28000	-28999:-28500
REAL	-29999:-29000	-29999:-29500
COMPLEX	-30999:-30000	-30999:-30500
DOUBLE PRECISION	-31999:-31000	-31999:-29500

TABLE 9.1. Data types and file type ranges.

A file type can be defined by choosing a number in the range corresponding to the data type used. For example, SPPACK uses file type -28000 for data files containing sampled data. To avoid conflicts between user defined file types and SPPACK file types, the user must stay within the range given in Table 9.1. Some commonly used file types are: sampled data files, analysis files and filter coefficient files. Sampled data files are used for storing sampled data coming from such sources as the analog-to-digital converter or a digital filter program. Analysis files store data from different speech analysis programs. Analysis outputs are stored as fixed length vectors with one vector for each consecutive window of analysis. Conventions have been established for the position of data within a vector, such as linear prediction coefficients, RMS energy and pitch frequency. Filter coefficient files store filter coefficients as polynomials of rational transfer functions. Filter coefficient files can be created with SPPACK's filter design programs or manually entered by the user.

The last part of the file header is modular and is called the *header module*. The contents of this module depend on the file type involved, and every file type has its own header module. A user can define new file types and header modules to accommodate his or her research activities. The layout of a header module can be described with the aid of a Header Module Description file (HMD). This file enables the HEADER command to display the contents of a file header module together with an adequate description.

9.8 Implementation Issues

SPPACK was developed on a PDP-11/34A computer under the RSX-11M and RT-11 operating systems. Although both operating systems have a lot in common, it was clear at an early stage that the package should hide operating system features as much as possible to facilitate portability. Portability has been achieved by using as much as possible a high-level language. Fortran-IV was chosen as the target language due to the already available programs and the availability of an efficient Fortran compiler. Only when no other possibility existed were programs written in assembly language (e.g. bit operations and data acquisition). Although Fortran is a non-structural language, it is very convenient for signal processing operations. However, for other tasks such as string and list processing it is a very cumbersome language and one might consider other languages such as C more suitable. The command interpretation procedures and the error detection mechanism will place a burden on memory and CPU time. However, we feel that these issues are very important for user-friendly systems, and should never be omitted for the sake of a faster execution time or a reduced memory use. Two practical issues that affect the portability of the system are the use of special *terminal features such as cursor control*, and *graphic displays*. The best solution is to have device independent software, but in most cases this possibility depends on the capabilities of the operating system. In our current implementation, the display program depends completely on the graphic device used. Future implementations, however, will use intermediate representations (e.g. GKS for graphics) and the use of different device drivers for different devices.

The command syntax allows only the specification of two files. This was not found to be a limitation. When a program requires more external files, we use fixed file names for these additional files.

9.9 Creation And Modification Of Modules

Despite the flexibility of the SPPACK system, many situations could arise in which a desired task is not available, and can not be constructed from the available modules either. In that case the user has to supply his or her own programs. Our basic philosophy is that if these programs will obey the SPPACK format (i.e. command syntax and file structure), the capability of the system will increase gradually. Another advantage is that the user can still use all the available SPPACK modules together with his own programs. Such an approach is only possible under the condition that it is easy to write programs in SPPACK style. We solved the problem as follows. Each module consists as much as possible of many sub-modules, which will usually correspond to sub-programs of the programming language in use. All sub-programs are stored in a library, which is accessible by all users. Furthermore, the basic module consists of many parts that are common to most programs, for example, command initialization, command processing, file handling. As a result, it is not to difficult to provide a source code that acts as a skeleton for the new program. Little modification is then required to make the new program. A further refinement would be a program generator that requires as input the names of commands and options, and that part of the code that is specific for the intended task. Such a program is currently implemented for the Unix system.

9.10 Examples

First we illustrate the difference between the prompting mode and the command line mode. Both examples will have the same result.

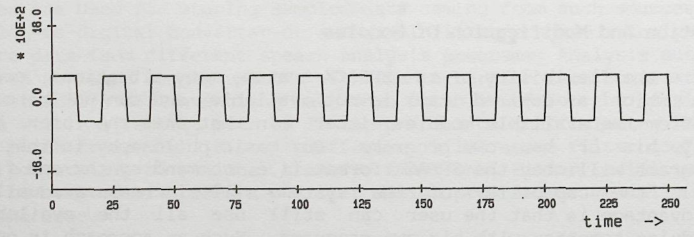
```
    FFT 1/SI:1024/RA:1024/W:0/M:3/RE:1024 R0 FR0
```

and

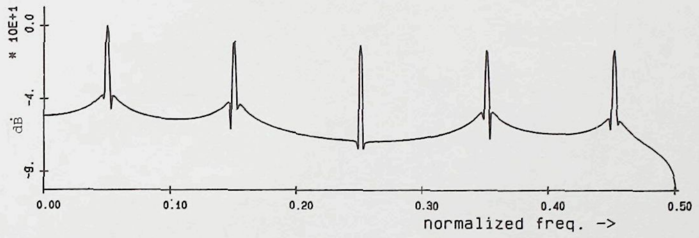
```
    FFT
    Input-file ? <*.DAT> :R0
    Output-file? <*.DAT> :FR0
    Start frame number       <        1> :
    End frame number?        <        1> :
    Window? 0=No,1=Han,2=Ham,3=Bar <     0> :
    Mode? 1=M, 2=LM, 3=NLM         <     3> :
    Frame size?                    <  1024> :
    Resolution?                    <  1024> :
    Frame rate?                    <  1024> :
```

A more complete example is the following illustration of a filter design procedure. A band-pass elliptic filter is designed to extract the second harmonic of a square wave. Comment is preceded by an exclamation mark.

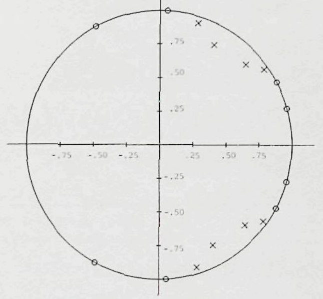
```
TFUN /M:7/D/PU:10/OF:-500/PER:20 DEMO1   !Create test signal
FFT 1/D/W:2 DEMO1 DEMO2                  !Compute spectrum
PLOT2 DEMO1                              !Plot time signal
```



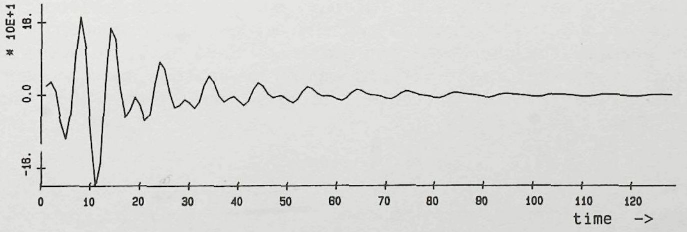
```
PLOT2 DEMO2                              !Plot spectrum
```



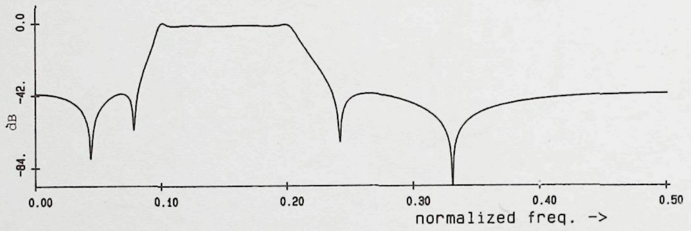
```
EFI 4,.5,10,.1,2,-40 FILTER              !Design filter
ZPLANE /D FILTER                         !Plot pole/zero diagram
```



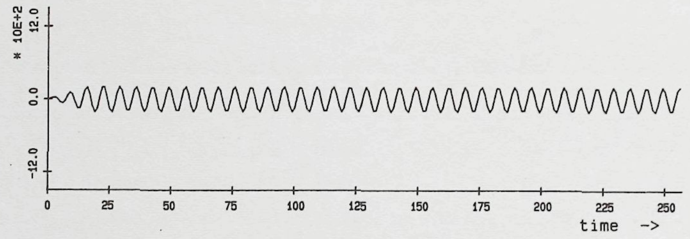
```
TFUN /M:7/PU:1/PER:1024/DE P0            !Create impulse
FILTER %WH/D P0 P1                       !Filter impulse
FFT 1/D P1 P2                            !Compute spectrum

PLOT2 P1                                 !Plot impulse response
```



```
PLOT2 P2                                  !Plot spectrum
```



```
FILTER %WH/D DEMO1 DEMO3                  !Filter test signal
PLOT2 DEMO3                               !Plot filtered signal
```



For other examples we refer to the SPPACK manuals.

9.11 Summary

In this chapter we described an interactive software environment suitable for speech research. The package was used for the development of the coders described in this thesis and it is considered to be very adequate. Some interesting features of the package include the capability of constructing new programs by "gluing" existing programs together, the different prompting formats, and the header module concept. This last feature enables the user to accommodate new data file headers to new programs. The development of programming generators and documentation tools will result in a suitable research environment for most users.

References

ASSP, Digital signal processing committee IEEE, *Programs for digital signal processing*, IEEE Press (1979).

Bedford Research,, *I*S*P- The interactive signal processor*, BR, Bedford, MA (1983).

Data Translation,, *DT-DATS*, DT, Marlboro, MA (1982).

Kopec, G.E., "The integrated signal processing system ISP," *IEEE Trans. Acoust., Speech, Signal Processing* Vol. ASSP-32(4) pp. 842-851 (August 1984).

Kroon, P., "SPPACK User's Guide," Report NT-37, Dept. of EE, Delft University of Technology, Delft, The Netherlands (April 1983).

Kroon, P., "SPPACK Programmer's Guide," Report NT-38, Dept. of EE, Delft University of Technology, Delft, The Netherlands (April 1983).

Signal Technology Inc.,, *ILS Interactive Laboratory Systems*, STI, Goleta, CA (1980).

process, quite good results were obtained with 1/4 bit per sample. As a result the coder enables the encoding of (near) toll quality speech at approximately 5 kb/s.

A performance evaluation of the investigated coders revealed that both the RPE and MPE coding techniques are very suitable for applications such as mobile telephony, where channel rates around 10 kb/s are available. The multi-path search approach produced good results but the complexity of the encoding procedure is enormous and not suitable for real-time implementations. We further examined the performance of the RPE coder for operations at bit rates around 6 kb/s, and we demonstrated that with the use of vector quantization techniques quite reasonable results can be obtained. Another interesting aspect is the performance of the coders for wide-band speech signals. Both RPE and MPE coders are able to obtain near transparency at approximately 25 kb/s.

The synthesis procedure for all coders is very straightforward, which makes them attractive for applications in voice-response systems. When the coders are used for wide-band speech coding applications, the lower complexity of the RPE coder makes it more attractive than the MPE coder.

A study has been made of the possibility of mapping the coder algorithm onto silicon. The RPE algorithm is particular suitable for VLSI implementation using pipeline and parallel structures. An advantage of the solution described using CORDIC processor elements is that other related algorithms such as LPC and lattice filters can be effectively implemented. Another advantage of the proposed structure is that a high throughput can be obtained, which is mandatory for wide-band applications.

Finally, we conclude with some suggestions for future research. First, we are of the opinion that the error weighting procedure should be investigated in more detail and that more and better perceptual solutions should be searched for. Another interesting point of research should focus on better code book structures for vector quantization. The use of, for example, lattice quantizers in a multi-path search approach enables the realization of large code books, with hardware of a moderate complexity. Further investigations in these directions will finally further push down the bit rates required for the encoding of toll quality speech.

A. APPENDIX A: SPEECH QUALITY EVALUATION

A.1 Introduction

The speech quality of the investigated coders was evaluated by using objective and subjective measures and real speech data. The speech material consists of ten English sentences spoken by both a male and female speaker and thirteen Dutch sentences spoken by both a male and a female speaker. Totally four different speakers were used. The sentences were recorded with professional equipment in a broadcast studio. The utterances were low-pass filtered (3400 Hz cut-off frequency) and sampled with a 12-bit linear converter at an 8 kHz sampling rate. For every utterance, the sound level was adjusted to use the full quantizer range without clipping. This procedure equalized the peak power level of the four speakers over all sentences.

A.2 Speech Material

The English and Dutch sentences were taken from [Goodman et al., 1976] and [Plomp and Mimpen, 1979], respectively. Each is a simple declarative sentence that can be spoken in about two seconds. The lists shown in tables A.1 and A.2 include all the phonemes of English and Dutch in initial, final, and intervocalic position.

1. A lathe is a big tool.
2. Grab every dish of sugar.
3. An icy wind raked the beach.
4. Her father failed many tests.
5. Joe brought a young girl.
6. The chairman cast three votes.
7. The boy was mute about his task.
8. Beige woodwork never clashes.
9. Both teams started from zero.
10. My cap is off for the judge.

TABLE A.1. The 10 English sentences spoken by 2 speakers each.

1. De bal vloog over de schutting.
2. Morgen wil ik maar een liter melk.
3. Deze kerk moet gesloopt worden.
4. De spoortrein was al gauw kapot.
5. De nieuwe fiets is gestolen.
6. Zijn manier van werken ligt mij niet.
7. Het slot van de voordeur is kapot.
8. Dat hotel heeft een slechte naam.
9. De jongen werd stevig aangepakt.
10. Het natte hout sist in het vuur.
11. Zijn fantasie kent geen grenzen.
12. De aardappels liggen in de schuur.
13. Alle prijzen waren verhoogd.

TABLE A.2. The 13 Dutch sentences spoken by 2 speakers each.

A.3 Objective Quality Measurement

A widely used quality measurement applied in waveform coders and related coders (e.g. hybrid coders) is the signal-to-noise ratio, evaluated over an utterance of speech. If $x(n)$ and $\hat{x}(n)$ represent the input and output signals of the coder, respectively, then the conventional SNR is

$$SNR = \sum_n \frac{x^2(n)}{[\hat{x}(n) - x(n)]^2} \text{ dB} \tag{A.1}$$

It has also long been known that in many situations the SNR measure does not correlate well with subjective performance [Goodman et al., 1976]. Another approach that correlates better with subjective performance is the segmental SNR (SNRSEG) [Noll, 1974] [Goodman et al., 1976]. The measure is the average of SNR measurements made up of segments of speech that are typically about 10-20 ms in duration. Thus

$$SNRSEG = \frac{1}{N} \sum_{i=1}^{N} (SNR)_i \text{ dB} \tag{A.2}$$

where $(SNR)_i$ corresponds to the signal-to-noise ratio in decibels for a segment i and N represents the number of segments in the utterance. To prevent any slight noise in low energy intervals (e.g. silent intervals) from causing large negative $(SNR)_i$, and thereby dominating the average in Eq. (A.2), we exclude these segments from this average by a threshold. A segment i will be included in the computation of SNRSEG if its energy p_i defined by

$$p_i = \frac{1}{k} \sum_{n=1}^{k} x^2(n) \tag{A.3}$$

exceeds a threshold T, where k represents the number of samples in the i-th segment. Furthermore, to prevent any one segment from dominating the average, we also limit the value of $(SNR)_i$ to a range of -10 to +80 dB. In our tests we set T to 5 and used 10 ms segments.

A.4 Subjective Quality Tests

To evaluate the subjective quality of the coders, listening tests have been used. The Diagnostic Rhyme Test (DRT) [Voiers, 1977] was not considered useful, due to the high quality of the synthetic signals. A more meaningful test is the Mean Opinion Score (MOS) procedure [Daumer, 1982] [Goodman and Nash, 1982], but a disadvantage is that it is very time consuming, and that many experienced listeners are required to provide reliable results. For our tests we used a more practical procedure, which we refer to as the Paired Comparison Test (PCT). The Paired Comparison Test is intended to rank a set of sentences processed in different ways. Each of the processed words and sentences is compared to the different versions by randomly playing pairs to the listener(s). The pairs are chosen so that all the permutations of the processed material are played exactly once. The partial score 1 is attributed to the preferred word and 0 to the other one. Finally, all these partial scores are summed, and averaged over the number of listeners and the number of test sentences. A final normalization is done by dividing them by the maximum attainable score. The systems under test are then listed in order of preference. Note that no absolute scores are obtained with this method, but that the PCT is very suitable for quickly determining the optimum values of specific system parameters.

References

Daumer, W.R., "Subjective evaluation of several efficient speech coders," *IEEE Trans. Communications* Vol. COM-30 pp. 655-662 (April 1982).

Goodman, D.J., B.J. McDermott, and L.H. Nakatani, "Subjective evaluation of PCM coded speech," *Bell Sys. Tech. J.* Vol. **55**(8) pp. 1089-1109 (Oct. 1976).

Goodman, D.J. and R.D. Nash, "Subjective quality of the same speech transmission conditions in seven different countries," *IEEE Trans. Communications* Vol. COM-30 pp. 642-654 (April 1982).

Noll, P., "Adaptive quantizing in speech coding systems," *Proc. Int. Zurich Seminar*, pp. B3(1)-B3(6) (March 1974).

Plomp, R. and A.M. Mimpen, "Improving the reliability of testing the speech reception threshold for sentences," *Audiology* Vol. 18 pp. 43-52 (1979).

Voiers, W.D., "Diagnostic evaluation of speech intelligibility," in *Speech intelligibility and speaker recognition*, ed. M.E. Hawley, Dowden Hutchinson Ross, Stroudsburg, PA (1977).

B. APPENDIX B: TIME-VARYING LINEAR FILTERS

The all-zero analysis filter A(z) and the all-pole synthesis filter 1/A(z), where A(z) is defined as

$$A(z) = 1 + \sum_{k=1}^{p} a_k z^{-k} \qquad (B.1)$$

can be implemented by different filter structures. Two well known structures commonly used in speech analysis and synthesis procedures are direct form filters [Oppenheim and Schafer, 1975], and lattice filters [Gray and Markel, 1973] [Makhoul, 1978]. Fig. B.1 shows the different structures for the all-zero synthesis filter A(z).

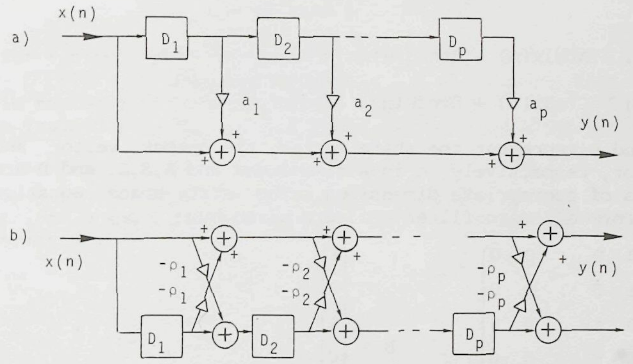


Figure B.1. Direct form (a) and lattice form (b) all-zero filters.

When disregarding finite word length effects, both types of filter structures realize the same transfer function; they are equivalent realizations. However, for speech synthesis applications, where the filter coefficients are refreshed at regular intervals, both filters show a different behavior, as we will demonstrate in this appendix.

The refreshment of the filter coefficients is organized as follows. For every frame of NSSIZE samples the filter coefficients are kept fixed. Before the samples of the next frame are computed, all the filter coefficients of the previous frame are replaced by the coefficients belonging to the new frame. To ensure continuous filtering, the states of the filters are not cleared when a new set of coefficients is dumped. The process of filtering a signal through such a time-varying filter is reversible by using the inverse filter with a similar structure and whose coefficients have the same time dependency (Fig. B.2).

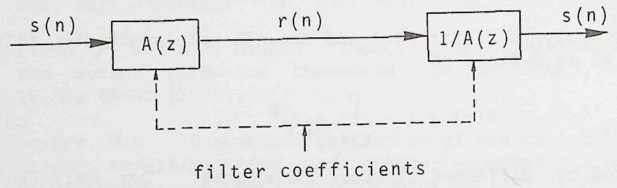


Figure B.2. Filtering and inverse filtering with time-varying filters.

We describe the filter transfer functions in the form of state-space equations:

$$x(n+1) = A(n)x(n) + B(n)u(n) \tag{B.2}$$

$$y(n) = C(n)x(n) + D(n)u(n)$$

where x,y and u represent the state vector, the output vector and the input vector, respectively, n is a time index and A,B,C, and D are system matrices of appropriate dimensions. The state-space equations of the direct form all-zero filter A(z) are given by:

$$A = \begin{bmatrix} 0 & 0 & . & 0 & 0 \\ 1 & 0 & . & 0 & 0 \\ 0 & 1 & . & 0 & 0 \\ 0 & 0 & . & 0 & 0 \\ . & . & . & . & . \\ 0 & 0 & . & 1 & 0 \end{bmatrix} \qquad B = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \\ . \\ 0 \end{bmatrix}$$

$$C = [\ a_1\ a_2\ .\ .\ a_p\ ] \qquad D = 1$$

The state-space equations of the lattice all-zero filter A(z) are given by

$$A = \begin{bmatrix} 0 & 0 & . & 0 & 0 \\ 1 & 0 & . & 0 & 0 \\ \rho_1\rho_2 & 1 & . & 0 & 0 \\ \rho_1\rho_3 & \rho_2\rho_3 & . & 0 & 0 \\ . & . & . & . & \\ \rho_1\rho_{p-1} & \rho_2\rho_{p-1} & . & 1 & 0 \end{bmatrix} \qquad B = \begin{bmatrix} 1 \\ -\rho_1 \\ -\rho_2 \\ -\rho_3 \\ . \\ -\rho_{p-1} \end{bmatrix}$$

$$C = [\ -\rho_1\ -\rho_2\ .\ .\ -\rho_p\ ] \qquad D = 1$$

The correspondence between the state-space matrices of the direct-form and lattice filter is given by the invertible matrix T.

direct form (A, B, C, D)

lattice form $(\ TAT^{-1}, TB, TC, D)$

The output for the current frame depends on the initial state of the filter and the current input. The relation between the initial state x_0 and the past data values y_- can be expressed as

$$x_0 = Cy_-$$

with

$$C = [B\ AB\ A^2B\ ..\ A^{p-1}B]$$

where for finite impulse response filters $A^p=0$. For the direct form filter C will be an identity matrix I_p, which means that the initial state is represented by the previous data p values. The output in the current frame due to the initial state can be related by the matrix O

$$y_+ = O\ x_0$$

The matrices O and C for the lattice filter can be obtained by the transformations

$$C' = TC = TI_p = T$$

$$O' = OT^{-1}$$

Hence we can write the output in the current frame due to past data as

$y_+ = Oy_-$ direct form

$y_+ = OT_+^{-1}T_-y_-$ lattice form

In the stationary case $T_- = T_+$, and both outputs will be the same. In the non-stationary case T_- and T_+ are not equal and the outputs will differ. The behavior of the lattice filter may cause undesired transition effects, which will affect the performance of coders employing these structures. However for practical situations, other aspects such as numerical stability will play a role, and the net result may be in favor of the lattice structure.

References

Gray, A.H. and J.D. Markel, "Digital lattice and ladder filter synthesis," *IEEE Trans. Audio and Electroacoustics* Vol. **AU-21** pp. 491-500 (1973).

Makhoul, J., "A class of all-zero lattice digital filters: properties and applications," *IEEE Trans. Acoust., Speech, Signal*

Processing Vol. **ASSP-26** pp. 304-314 (1978).

Oppenheim, A.V. and R.W. Schafer, *Digital signal processing*, Prentice Hall, Englewood Cliffs, New Jersey (1975).

C. APPENDIX C: DISTANCE LIMITATION ALGORITHM MPE CODER

C.1 Introduction

This appendix describes the algorithm which ensures that the distance between two succeeding pulses within a search frame does not exceed a certain maximum. This option can be useful for coding purposes. For example, when using variable-length-to-block coding to encode an excitation sequence of size NSSIZE=80 with NP=8 pulses, 7 bits/pulse are required if the maximum distance is limited to 20 successive zeros. Hence a critical distance of NZM=21 or more zeros has to be avoided. This is done by checking distances between pulses. If a critical distance is detected, a search interval is computed in which a pulse has to be located to cancel it. To reduce a possible sub-optimal performance of the search procedure, this position limitation is only imposed at the latest possible instant. Below we shall first describe a method for finding the search intervals and their boundary points. After that, the minimal number of pulses which are necessary to cancel every possible critical distance is determined. Then we will describe in detail when the limitation has to be imposed. In the last section, a Nassi-Schneiderman diagram of the algorithm is given.

C.2 Search Intervals And Their Boundary Points

The multi-pulse excitation sequence within a search frame is found sequentially. At every step a new pulse is located by trying all locations within the search frame, and selecting that location which yields a minimum error. When it is necessary to cancel a critical distance, search intervals in which pulses have to be located are computed. Suppose a critical distance is found between the positions 10 and 40. The boundary points of the search intervals, using NZM=21, are then given by:

```
GRP(1) = 40 - NZM = 19
GRP(2) = 10 + NZM = 31
```

If a pulse is located in the search interval (19,31) the critical distance is canceled. For a larger critical distance, two or more search intervals have to be used. Suppose a critical distance is found between the positions 10 and 60. Then the 4 boundary points are found by:

```
GRP(1) = 60 - 2*NZM = 18
GRP(2) = 10 +  NZM  = 31
GRP(3) = 60 -  NZM  = 39
GRP(4) = 10 + 2*NZM = 52
```

Hence 2 pulses have to be located in the intervals (18,31) and (39,52), respectively, to cancel this critical distance. In general, a distance of less than (k+1)*NZM zeros can be canceled using k pulses.

C.3 Moment Of Imposing Constraints

In Section C.2 a possible way to overcome a critical distance is given. To reduce the degradation introduced by this limitation procedure, we want to use it as late as possible. Suppose we want to use it when NSTART pulses still have to be located. Then the largest possible critical distance equals (NSSIZE - NP - NSTART) zeros. This critical distance can be canceled with NSTART pulses, if

$$(NSSIZE - NP - NSTART) < (NSTART + 1)*NZM \qquad (C.1)$$

since (k+1)*NZM zeros can be canceled by k pulses. Rewriting Eq. (C.1) gives for NSTART:

$$NSTART > (NSSIZE - NP - NZM) / (NZM + 1) \qquad (C.2)$$

The smallest integer value satisfying Eq. (C.2) equals the number of pulses which is sufficient to overcome every possible critical distance. For an excitation sequence of NSSIZE=80 samples and with NP=8 pulses to be located, it is necessary to consider distances only when NSTART=3 or less pulses still have to be located.

For every critical distance the number of pulses necessary to cancel it is computed. Once the total number of pulses NPNEC necessary for removing critical distances is known, it is checked whether the amount of pulses to be placed is greater than NPNEC. In that case it is not necessary to use a placing constraint, and care only has to be taken to prevent any critical distance from being created. For example, suppose NPNEC=1 while 2 pulses are to be located. Then it is not necessary to define a reduced search interval for the 7th pulse to be placed. However, if the position of the right-most pulse is, say, equal to 40, then locating the 7th pulse at position 70 would create a second critical distance. A boundary point at position 61 must then be used, reducing the search frame size to 61 samples.

C.4 Diagram Of The Algorithm

As was mentioned before, the location of a pulse is selected one pulse at a time in a search frame of size NSSIZE. If a distance limitation is necessary, the search interval is redefined. The search procedure can be represented by:

```
    SEARCH (GRP,NGRP)
```

GRP contains successively the starting point and the ending point of search frames, and NGRP defines the number of elements in GRP. For a search between the locations 1 and NSSIZE, we have to set

```
GRP(1) := 1
GRP(2) := NSSIZE
NGRP   := 2
```

The procedure LIMIT will redefine the search intervals if necessary. This procedure has the following parameters:

Global Input Parameters:

NSSIZE	=	Size of a search frame
NP	=	Total number of pulses which will be located
NPFND	=	Number of pulses already located
BEP(i)	=	Array of locations of the NPFND pulses, BEP(1)=0 is the beginning of a frame, BEP(2) is the position of the first pulse etc.
NZM	=	Smallest *critical distance*, i.e. a number of NZM or more zeros has to be avoided
NSTART	=	Remaining number of pulses, which after the algorithm LIMIT will be used.

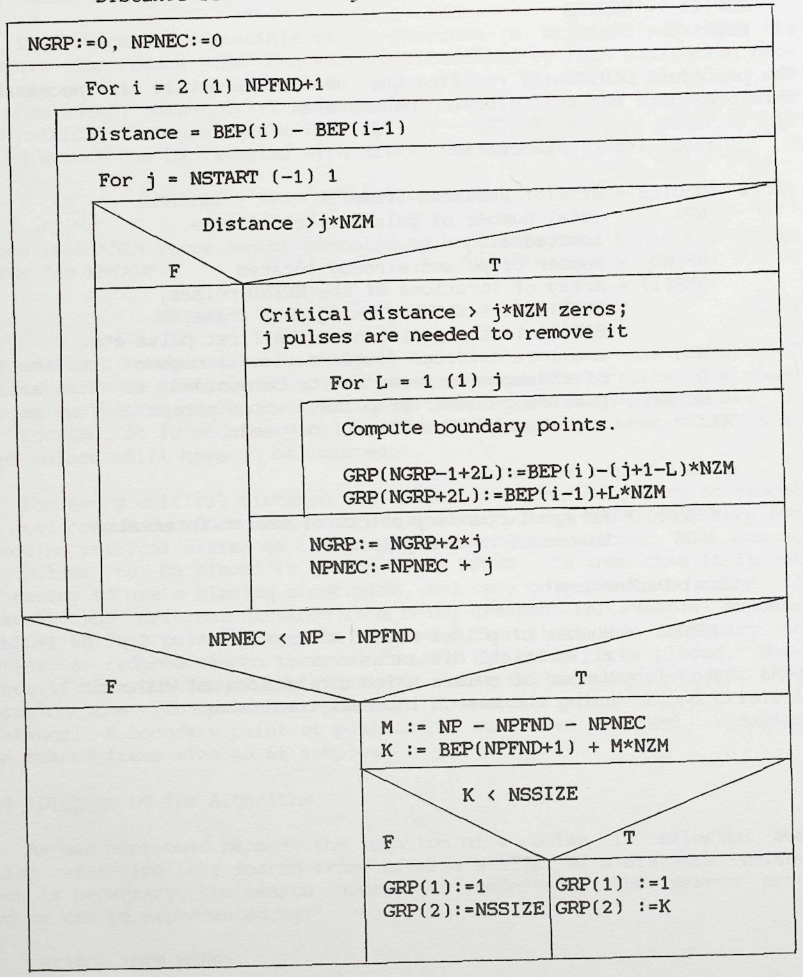
Global Output Parameters:

GRP(i)	=	Array of boundary points of search intervals
NGRP	=	Number of boundary points

Local Parameters:

NPNEC	=	Number of pulses needed to cancel all critical distances
M	=	Number of pulses which can be located without using the search interval constraints

Distance limitation algorithm:



D. APPENDIX D: VECTOR QUANTIZATION

D.1 Introduction

Vector quantization (VQ) [Gray, 1984] can be viewed as a generalization of Pulse Code Modulation techniques (PCM) and can be basically described as follows. An ordered set of k samples (k-dimensional vector) is mapped onto one of a finite set of representative (output) vectors. The index of the resulting output vector is used for transmission or storage purposes. At the decoder this index is used to reproduce the corresponding output vector, which approximately reconstructs the original input vector. Some difficulties associated with VQ are how to choose a representative set of output vectors and how to find the appropriate output vector as fast as possible.

D.2 Basic Structure Of A Vector Quantizer

A vector quantizer maps a k-dimensional Euclidean space R^k onto a finite subset Y of R^k.

$$Q:R^k \rightarrow Y \tag{D.1}$$

where $Y = \{y_1, y_2, \ldots, y_N\}$ and y_i, called an output point, is in R^k for each i. Associated with every N point quantizer in R^k is a partition

$$R_1, R_2, \ldots, R_N \tag{D.2}$$

where

$$R_i = Q^{-1}(y_i) = \{x \in R^k : Q(x) = y_i\} \tag{D.3}$$

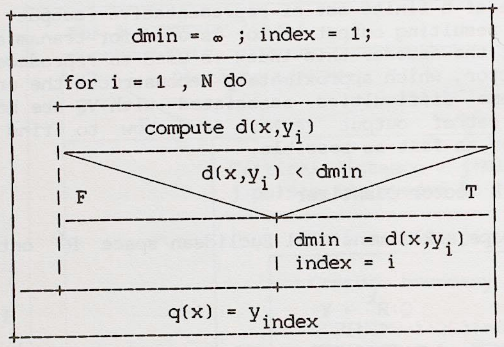
With this definition, it follows that

$$\bigcup_{i=1}^{N} R_i = R^k \text{ and } R_i \cap R_j = 0 \text{ for } i \neq j \tag{D.4}$$

The partitions $\{R_i\}$ in R^k are called Voronoi cells or regions and the set of N output vectors $\{y_i\}$ is called a code book of size N. An appropriate choice of the code book vectors and the corresponding Voronoi regions is crucial for the performance of a vector quantizer. The mapping of an arbitrary input vector onto one of a limited set of output vectors will introduce a distortion d(x,y), and we have to use a code book that minimizes the average distortion between input and output vectors. The design of optimal vector quantizers from empirical data was proposed in [Linde et al., 1980]. This algorithm uses a training set of random vectors generated from a source for which the quantizer is to be optimized. Vector quantizers that use a code book obtained from real samples (training data) usually lack any structure within the code book, and the quantizers are referred to as random quantizers. In the remainder we first examine this type of VQ's in more detail.

D.3 Code Book Generation Procedures For Vector Quantizers

The procedure for generating a code book depends on the search procedure used. We distinguish two search procedures: full search and binary search [Gray and Linde, 1982]. The flowchart for the full search procedure is given in Fig. D.1



```
dmin = ∞ ; index = 1;
for i = 1 , N do
    compute d(x,y_i)
    d(x,y_i) < dmin
    F:
    T:  dmin = d(x,y_i)
        index = i
q(x) = y_index
```

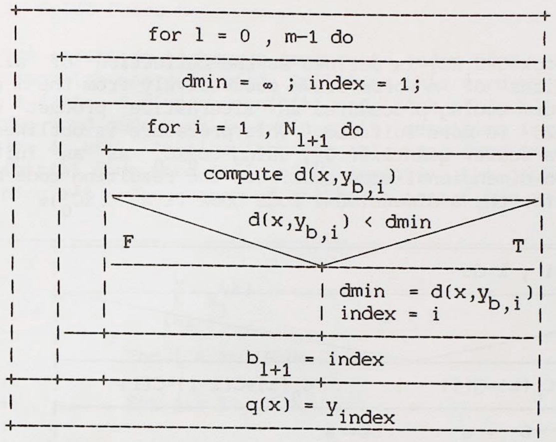
Figure D.1. Full search vector quantizer.

Due to the enormous complexity for large code books, a simplification of the search procedure is required. A useful technique is called tree searched VQ (TSVQ) [Buzo et al., 1980]. A TSVQ is most easily visualized as a tree which is labeled with vectors and is searched by the coder. The tree structure is completely specified by an m-dimensional rate vector $R_m = [r_1, r_2, \ldots, r_m]$. The number of code book vectors at a node on level l is given by $N_l = 2^{r_l}$. The search procedure selects among the vectors of the first level (m=1) the one that yields the lowest distortion. For the second level, only the vectors connected to the node of the previously selected vector are considered. Again, the vector that yields the minimum distortion is selected. A tree search reduces the number of comparisons at the expense of sub-optimality (not all vectors are searched), and an increase in storage space. The address of the selected vector is given by specifying the path vector b:

$$b = (b_1, b_2, \ldots, b_l) \qquad l > 0$$

$$b = \emptyset \text{ for } l = 0$$

The search procedure is shown in Fig. D.2.



```
for l = 0 , m-1 do
    dmin = ∞ ; index = 1;
    for i = 1 , N_{l+1} do
        compute d(x,y_{b,i})
        d(x,y_{b,i}) < dmin
        F:
        T:  dmin = d(x,y_{b,i})
            index = i
    b_{l+1} = index
q(x) = y_index
```

Figure D.2. Tree search vector quantizer.

D.4 Code Book Generation Procedures

Code book generation procedures that use a training set are all based on a generalization of the Lloyd algorithm.

Step 0: Given a training sequence and an initial decoder

Step 1: Encode the training sequence; if the average distortion is small enough then quit.

Step 2: Replace each reproduction code word by the centroid of all training vectors that mapped onto that code word. Goto step 1

Two basic approaches exist to generate the initial code book:

1. start with a simple (i.e. lower dimension) code book of the correct size ("random" codes, product codes);

2. start with a simple small code book (with the correct dimension) and recursively construct larger ones (splitting).

D.4.1 *Random Codes*

Use the first N vectors in the training sequence or select widely spaced vectors.

D.4.2 *Product Codes*

A product code book $C=C_i XC_m$ is defined as the collection of all M possible concatenations of m words drawn successively from the m code books C_i. For waveform coding procedures an alternative product code [Abut et al., 1982] is more suitable. This procedure is outlined as follows: start with a scalar quantizer C_0, using C_0XC_0 as an initial guess. Design a two-dimensional code book. Use the resulting code book as an initial guess for the 3-dimensional code book (i.e. C_2XC_0).

```
For i = 1 , k do
    if k > 1
        F: $C_0(k)=Q(1)$
        T: $C_0(k)=C(k-1)*C(1)$
    m = 0    $D_{-1} = \infty$    $DD = \infty$
    while $DD > \epsilon$
        if m > 0
            F:
            T: $C_m(k)=cen\{S_l\}$ ,l=1,N
        For l = 1 , N
            $S_l = \{x_j^{(k)}: d(x_j^{(k)},y_l) \le d(x_j^{(k)},y_n) ,l \ne n$
        $D_m = L^{-1} \sum_{i=0}^{L-1} d(x_i^{(k)},q(x_i^{(k)}))$
        $DD = (D_{m-1} - D_m)/D_m$
        m = m + 1
```

Figure D.3. Product method for the generation of a code book for full search VQ.

The design procedure is shown in Fig. D.3. The initial scalar quantizer is given by Q(1). The subscript m of the code book $C_m(k)$ represents the code book obtained after the m-th iteration. The training sequence consists of L k-dimensional vectors

$$x_i^{(k)} ,i=1,..,L \quad \text{with} \quad L=J/k \tag{D.5}$$

D.4.3 *Split Codes*

The splitting technique constructs codes of a chosen dimension by starting with one code word. Splitting this word by perturbing its value with a factor δ gives an initial code book of double size. The resulting two-word code book is again split, resulting in an initial code book of size 4, etc. An algorithm for this procedure is shown in Fig. D.4. The initial code word is found by computing the centroid of the complete training set. The number of splitting operations is given by $R=\log_2 N$.

```
N = 1
$C_0(k) = L^{-1} \sum_{i=1}^{L} x_i^{(k)}$
For i = 1 , R do
    For j = 1 , N do
        $y_{j+N} = y_j(1+\delta)$
    N = 2 * N
    m = 0    $D_{-1} = \infty$    $DD=\infty$
    while $DD > \epsilon$
        if m > 0
            F:
            T: $C_m(k)=cen\{S_l\}$ ,l=1,N
        For l = 1 , N
            $S_l = \{x_j^{(k)}: d(x_j^{(k)},y_l) \le d(x_j^{(k)},y_n) ,l \ne n$
        $D_m = L^{-1} \sum_{i=0}^{L-1} d(x_i^{(k)},q(x_i^{(k)}))$
        $DD = (D_{m-1} - D_m)/D_m$
        m = m + 1
```

Figure D.4. Splitting procedure for the generation of a code book for full search VQ.

To generate a code book suitable for tree search, we can use a modified split algorithm. The procedure is outlined as follows. Suppose there are m levels in a tree, and at each level there are r_j branches (j=1,..,m).

a. An r_1 full search code book is first generated.

b. The training data are partitioned into r_1 subsets, each containing all the training vectors matched to one of the r_1 code words.

c. For each subset, an r_2 size full search code book is generated.

d. Steps b) and c) are repeated for r_2 and r_3 and so on, until all of the code words are generated at the bottom of the tree.

The total number of code words stored for such a code book is

$$\sum_{i=1}^{m} \prod_{j=1}^{i} r_j \tag{D.6}$$

D.4.4 *Empty Cells*

During the design of a code book, we will sometimes encounter empty cells. This effect is often found if the training set is small in relation to the number of output values. But even for large training sets empty cells can still be encountered, which means a waste of the available bits. Here we describe some procedures to reduce or to avoid empty cells. For example, after the clustering operation the number of input vectors assigned to every cluster is checked. If a cell is empty, the corresponding output point is deleted and a new output point is defined by "splitting" the output point representing the cluster with the highest distortion. In [Rabiner et al., 1983] the binary split algorithm was used, and for an empty cluster, the "largest" cluster is split into two clusters. The convergence test is bypassed to ensure a reclassification in which each cluster is non-empty. Note that this check is done during the classification iterations. In [Linde et al., 1980] the binary split algorithm was used and for an empty cluster the new code word was set to the old code word. In [Gray and Linde, 1982] empty clusters are treated by choosing the new output value equal to a perturbed value of the centroid of all training data. For a second empty cluster the corresponding output value is set to another perturbed value of the initial centroid.

D.5 Vector Quantization Of LPC Filter Coefficients

In the previous sections we described various techniques for generating code books. In this section we consider the application of VQ for encoding LPC filter coefficients. The choice of an appropriate distortion measure is very important. For LPC quantization, the distortion measure should be consistent with the residual energy minimization concept of LPC. Three distortion measures having this property are [Gray et al., 1980] the Itakura-Saito measure, the Itakura measure, and the likelihood ratio measure. In [Wong et al., 1982] [Juang et al., 1982] the likelihood ratio was chosen as distortion measure. A significant advantage of this measure is that the gain term is separated from the filter, so it can be quantized separately, thereby reducing the size of

the code book. Let α_p represent the residual energy resulting from inverse filtering the speech signal X(z) with the optimal p-th order LPC filter $A_p(z)$. Also, let $1/A(z)$ be any p-th order all-pole model. Inverse filtering X(Z) with A(z) will result in a residual energy α, $\alpha \geq \alpha_p$ because α is minimized by $A_p(z)$. The likelihood ratio measure is defined for the two unity gain model spectra as

$$d_{LR}(1/A_p, 1/A) = \int_{-\pi}^{+\pi} |A(e^{j\phi})/A_p(e^{j\phi})|^2 \frac{d\phi}{2\pi} - 1 \tag{D.7}$$

$$= \frac{\alpha}{\alpha_p} - 1$$

The evaluation of D_{LR} can be carried out with a scalar product [Buzo et al., 1980]:

$$d_{LR}(1/A_p, 1/A) = \alpha_p^{-1}[r_x(0)r_a(0) + 2\sum_{i=1}^{p} r_x(i)r_a(i)] - 1 \tag{D.8}$$

where $r_x(i)$ and $r_a(i)$ denote the auto-correlation sequences of the input speech data and the polynomial coefficients of A(z), respectively. In [Roucos et al., 1982] the Euclidian distance on LARs was proposed as an alternative distance measure, and it was reported that in informal listening tests no differences could be heard between VQ using such a LAR measure and a quantizer using an Itakura distance measure.

D.6 Lattice Quantizers

Lattice quantizers are considered as a special class of vector quantizers that possess a structured set of output vectors [Gersho, 1982]. The output points of a lattice quantizer lie in a bounded region of a lattice. The basic theory of lattices, a branch of geometric number theory, is presented in [Cassels, 1971]. A lattice in k dimensions is defined by a non-singular k by k matrix M, the so-called generator matrix, and the lattice consists of all integer combinations of the rows of M. The lattice points form a regularly spaced array of points in a k-dimensional space and the origin is always a lattice point. The Voronoi cells for all lattice points are congruent and, equivalently, the lattice quantizer is a uniform quantizer.

In [Sloane, 1981] tables are given for a set of basic or root lattices. The normalized moment of inertia of a lattice determines the performance of a lattice quantizer when the mean-square error distortion measure is used. In [Conway and Sloane, 1982] the values of these moments of inertia are tabulated for different lattice structures.

An interesting lattice is called E_8 [Sloane, 1981] and may be obtained by applying construction method A to the extended Hamming code of length eight. Construction method A goes as follows: if C is a binary code of length n, then the set of centers

$$c + 2x \ (c \in C,\ x \in Z^n) \tag{D.9}$$

forms a sphere packing in R^n. Most of the properties of this packing can be obtained directly from the code C. The extended Hamming code is given by

$$\begin{matrix}
0&0&0&0&0&0&0&0\\
0&0&0&1&0&1&1&1\\
0&0&1&0&1&1&0&1\\
0&0&1&1&1&0&1&0\\
0&1&0&0&1&1&1&0\\
0&1&0&1&1&0&0&1\\
0&1&1&0&0&0&1&1\\
0&1&1&1&0&1&0&0\\
1&0&0&0&1&0&1&1\\
1&0&0&1&1&1&0&0\\
1&0&1&0&0&1&1&0\\
1&0&1&1&0&0&0&1\\
1&1&0&0&0&1&0&1\\
1&1&0&1&0&0&1&0\\
1&1&1&0&1&0&0&0\\
1&1&1&1&1&1&1&1
\end{matrix} \tag{D.10}$$

and can be spanned by the following base:

$$\begin{matrix}
0&0&0&0&0&0&0&0\\
\\
1&1&1&0&1&0&0&0\\
0&1&1&1&0&1&0&0\\
0&0&1&1&1&0&1&0\\
\\
1&1&1&1&1&1&1&1
\end{matrix} \tag{D.11}$$

The component 2x ensures that all vectors with even coordinates belong to the lattice. This is the space spanned by

$$\begin{matrix}
2&0&0&0&0&0&0&0\\
0&2&0&0&0&0&0&0\\
0&0&2&0&0&0&0&0\\
0&0&0&2&0&0&0&0\\
0&0&0&0&2&0&0&0\\
0&0&0&0&0&2&0&0\\
0&0&0&0&0&0&2&0\\
0&0&0&0&0&0&0&2
\end{matrix} \tag{D.12}$$

By combining the sets of Eqs. (D..) and (D.) and removing the dependent vectors, we obtain the following generator matrix M:

$$M = \tfrac{1}{2}\begin{bmatrix}
2&0&0&0&0&0&0&0\\
0&2&0&0&0&0&0&0\\
0&0&2&0&0&0&0&0\\
0&0&0&2&0&0&0&0\\
1&1&1&0&1&0&0&0\\
0&1&1&1&0&1&0&0\\
0&0&1&1&1&0&1&0\\
1&1&1&1&1&1&1&1
\end{bmatrix} \tag{D.13}$$

The normalization factor $\frac{1}{2}$ will normalize the value of the sphere radius to $\rho=\frac{1}{2}$. The lattice points of E_8 are located on concentric shells with radius m around the origin. In [Sloane, 1981] the number of points for a given value of m are tabulated. If we drop the factor $\frac{1}{2}$ the value of ρ will become equal to 1. The radius of the first shell will be equal to $4\rho^2 = 4$. The value of N_1 equals 240, which means that we have to find 240 vectors generated by the matrix M. By subtracting the vectors of Eq. (D.12) from those of Eq. (D.11) we obtain the following combinations:

All 14 vectors of Eq. (D.10) with norm 4

$$\left.\begin{matrix}
\pm1&\pm1&\pm1&0&\pm1&0&0&0\\
0&\pm1&\pm1&\pm1&0&\pm1&0&0\\
0&0&\pm1&\pm1&\pm1&0&\pm1&0
\end{matrix}\right\}\ 14 * 16 = 224$$

all perturbations of

$$\left.\begin{matrix}
\pm2&0&0&0&0&0&0&0
\end{matrix}\right\}\ 2 * 8 = 16$$

Total: 240

The algorithms for finding the closest point of a lattice to an arbitrary point $x \in R^k$ can be simple. For example, to find the closest point of the integer lattice Z^n (i.e. the lattice consisting of all points with integer coordinates in the space R^n) to an arbitrary point x, we round each coordinate of the vector x to the nearest integer to obtain the vector f(x), which points to the closest point of the lattice Z^n. In [Conway and Sloane, 1982] algorithms for the other root lattices can be found. When the subset of the lattice consists of a shell with radius m, more efficient algorithms [Adoul et al., 1984] can be found.

References

Abut, H., R.M. Gray, and G. Rebolledo, "Vector quantization of speech and speech like waveforms," *IEEE Trans. Acoust., Speech, Signal Processing* Vol. **ASSP-30** pp. 423-435 (June 1982).

Adoul, J.P., C. Lamblin, and A. Leguyader, "Baseband speech coding at 2400 bps using spherical vector quantization," *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing*, pp. 10.4.1-10.4.4 (March 1984).

Buzo, A., A.H. Gray, R.M. Gray, and J.D. Markel, "Speech coding based upon vector quantization," *IEEE Trans. Acoust., Speech, Signal Processing* Vol. **ASSP-28** pp. 562-574 (Oct. 1980).

Cassels, J.W.S., *An introduction to the Geometry of numbers*, Springer-Verlag, New York (1971).

Conway, J.H. and N.J.A. Sloane, "Voronoi regions of lattices, second moments of polytopes, and quantization," *IEEE Trans. Inform. Theory* Vol. **IT-28**(2) pp. 211-226 (March 1982).

Conway, J.H. and N.J.A. Sloane, "Fast quantizing and decoding algorithms for lattice quantizers and codes," *IEEE Trans. Inform. Theory* Vol. **IT-28** pp. 227-232 (March 1982).

Gersho, A., "On the structure of vector quantizers," *IEEE Trans. Inform. Theory* Vol. **IT-28** pp. 157-166 (March 1982).

Gray, R.M., A. Buzo, A.H. Gray, and Y. Matsuyama, "Distortion measures for speech processing," *IEEE Trans. Acoust., Speech, Signal Processing* Vol. **ASSP-28** pp. 367-376 (August 1980).

Gray, R.M. and Y. Linde, "Vector quantizers and predictive quantizers for Gauss-Markov sources," *IEEE Trans. Commun.* Vol. **COM-30**(2) pp. 381-389 (Feb. 1982).

Gray, R.M., "Vector Quantization," *IEEE ASSP Magazine* Vol. **1**(2) pp. 4-29 (April 1984).

Juang, B.H., D.Y. Wong, and A.H. Gray, "Distortion performance of vector quantization for LPC voice coding," *IEEE Trans. Acoust., Speech, Signal Processing* Vol. **ASSP-30** pp. 294-304 (1982).

Linde, Y., A. Buzo, and R.M. Gray, "An algorithm for vector quantizer design," *IEEE Trans. Communications* Vol. **COM-28** pp. 84-95 (Jan. 1980).

Rabiner, L.R., M.M. Sondhi, and S.E. Levinson, "Note on the properties of a vector quantizer for LPC coefficients.," *Bell Syst. Tech. J.* Vol. **68**(8)(Oct. 1983).

Roucos, S., R.M. Schwartz, and J. Makhoul, "Vector quantization for very-low-rate coding of speech," *Proc. GLOBCOM*, pp. 1074-1078 (1982).

Sloane, N.J.A., "Tables of sphere packings and spherical codes," *IEEE Trans. Inform. Theory* Vol. **IT-27**(3) pp. 327-338 (May 1981).

Wong, D.Y., B.H. Juang, and A.H. Gray, "An 800 bits/s vector quantization LPC vocoder," *IEEE Trans. Acoust., Speech, Signal Processing* Vol. **ASSP-30** pp. 770-780 (Oct. 1982).

E. APPENDIX E: CORDIC ALGORITHM

The CORDIC (COordinate Rotation DIgital Computer) technique [Volder, 1959] [Walther, 1971] is a digital shift-and -add bit recursive computing technique to execute a specific set of functions including plane rotations on two dimensional vectors, trigonometric functions, logarithms, multiplications and divisions.

Let the quantity m characterize one of the following coordinate systems: circular (m=1), linear (m=0), and hyperbolic (m=-1). Within each coordinate system the norm $|u|$ and angular component α of a vector $u=(x,y)$ are given by

$$|u| = (x^2 + my^2)^{\frac{1}{2}} \tag{E.1}$$

$$\alpha = m^{\frac{1}{2}} \tan^{-1}(m^{\frac{1}{2}}y/x) \tag{E.2}$$

This vector u can be rotated over an angle α by a multiplicative norm-preserving operator $R_m(\alpha)$, that is, if

$$v = R_m(\alpha)\ u = \begin{bmatrix} \cos(m^{\frac{1}{2}}\alpha) & -m^{\frac{1}{2}}\sin(m^{\frac{1}{2}}\alpha) \\ m^{-\frac{1}{2}}\sin(m^{\frac{1}{2}}\alpha) & \cos(m^{\frac{1}{2}}\alpha) \end{bmatrix} \tag{E.3}$$

$$= \cos(m^{\frac{1}{2}}\alpha) \begin{bmatrix} 1 & -m^{\frac{1}{2}}\tan(m^{\frac{1}{2}}\alpha) \\ m^{-\frac{1}{2}}\tan(m^{\frac{1}{2}}\alpha) & 1 \end{bmatrix}$$

then

$$|v| = |u| \tag{E.4}$$

and

$$\phi_v = \phi_u + \alpha \tag{E.5}$$

The rotations in the different coordinate systems are shown in Fig. E.1.

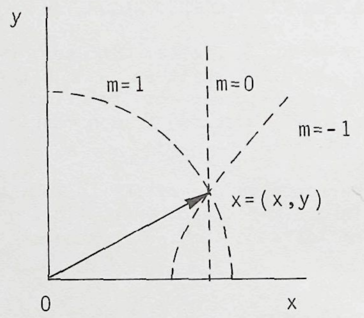


Figure E.1. Rotation in different coordinate systems.

Different functions can now be obtained in two different modes. In the first mode, the coordinate components of a vector and an angle of rotation are given, and the coordinate components of this vector, after rotation through the given angle, are computed. This mode is called *rotation*. In the second mode, called *vectoring*, the coordinate components of a vector are given and the magnitude and angular argument of this vector are computed. For example, the circular rotation (m=1) is given by

$$\begin{bmatrix} x_n \\ y_n \end{bmatrix} = \begin{bmatrix} \cos\alpha & -\sin\alpha \\ \sin\alpha & \cos\alpha \end{bmatrix} \begin{bmatrix} x_0 \\ y_0 \end{bmatrix} \tag{E.6}$$

First we introduce an auxiliary variable z that accumulates the total rotation α

$$z_n = z_0 + \alpha \tag{E.7}$$

In the rotation mode we set z_0 equal to $-\alpha$, and rotate the vector until $z_n \to 0$. The values of x_n, y_n and z_n will become

$$x_n = x_0 \cos\alpha - y_0 \sin\alpha$$

$$y_n = x_0 \sin\alpha + y_0 \cos\alpha$$

$$z_n = 0$$

In the vectoring mode, the vector $u_0 = (x_0\ y_0)$ is rotated through an angle α until $u_n = (x_n\ 0)$, and z_n will be equal to the net rotation α provided it was initially equal to zero.

$$x_n = x_0 \cos\alpha - y_0 \sin\alpha = 0 \to \alpha = \tan^{-1}(y_0/x_0)$$

$$y_n = x_0 \sin\alpha + y_0 \cos\alpha = (x_0^2+y_0^2)^{\frac{1}{2}}$$

$$z_n = z_0 + \tan^{-1}(y_0/x_0)$$

By choosing either vectoring mode or rotation mode and using different coordinate systems (m=0,-1,1), we obtain the functions listed in Fig. E.2.

```
    +---+                                   +---+
x -| x |- x cos z - y sin z            x -| x |- (x^2 + y^2)^½
    |   |                                   |   |
y -| y |- y cos z + x sin z            y -| y |- 0
    |   |                                   |   |
z -| z |- 0                            z -| z |- z + tan^-1(y/x)
    +---+                                   +---+

CIRCULAR (m=1):z → 0                   CIRCULAR(m=1):y → 0

    +---+                                   +---+
x -| x |- x                            x -| x |- x
    |   |                                   |   |
y -| y |- y + xz                       y -| y |- 0
    |   |                                   |   |
z -| z |- 0                            z -| z |- z + x/y
    +---+                                   +---+

LINEAR (m=0):z → 0                     LINEAR(m=0):y → 0

    +---+                                   +---+
x -| x |- x cosh z - y sinh z          x -| x |- (x^2 - y^2)^½
    |   |                                   |   |
y -| y |- y cosh z + x sinh z          y -| y |- 0
    |   |                                   |   |
z -| z |- 0                            z -| z |- z + tanh^-1(y/x)
    +---+                                   +---+

HYPERBOLIC (m=-1):z → 0                HYPERBOLIC (m=-1):y → 0
```

Figure E.2. The CORDIC functions.

The basic idea of the CORDIC technique is to factor the rotation operator $R_m(\alpha)$ into a fixed length sequence of "elementary" rotations, that is

$$R_m(\alpha) = R_m(\sigma_0\alpha_{m,0})\, R_m(\sigma_1\alpha_{m,1}) \ldots\ldots R_m(\sigma_p\alpha_{m,p}) \tag{E.8}$$

where the $\alpha_{m,i}$ form an ordered set of positive angles for each of the coordinate systems, with

$$\tfrac{1}{2}\alpha_{m,i} \le \alpha_{m,i+1} \le \alpha_{m,i} \tag{E.9}$$

and the $\{\sigma_i\}$ form a set of sign parameters $(\sigma_i = \pm 1)$ such that the value of α is approximated by

$$\alpha = \sum_{i=0}^{p} \sigma_{m,i}\alpha_{m,i} \qquad (E.10)$$

and

$$|\alpha - \sum_{i=0}^{p} \sigma_{m,i}\alpha_{m,i}| \le \alpha_{m,p} \qquad (E.11)$$

By choosing the basic angles $\alpha_{i,m}$ equal to integral powers of two, the micro rotations $R_m(\sigma_i\alpha_{m,i})$ can be performed as shift and add operations, instead of multiplications. Thus

$$m^{-\frac{1}{2}}\tan m^{\frac{1}{2}}\alpha_{m,i} = 2^{-s_{m,i}} , \; s_{m,i} \ge 0 \qquad (E.12)$$

The rotation $R_m(\alpha)$ can then be written as

$$\begin{bmatrix} x_n \\ y_n \end{bmatrix} = K_m \prod_{i=0}^{p} \begin{bmatrix} 1 & m\sigma_{i,m}2^{-s_{i,m}} \\ \sigma_{i,m}2^{-s_{i,m}} & 1 \end{bmatrix} \begin{bmatrix} x_0 \\ y_0 \end{bmatrix} \qquad (E.13)$$

The factor K_m

$$K_m = \prod_{i=0}^{p} \cos(m^{\frac{1}{2}}\alpha_{m,i}) = \prod_{i=0}^{p} [1 + m\tan^2(m^{\frac{1}{2}}\alpha_{m,i})]^{-\frac{1}{2}} \qquad (E.14)$$

depends only on the values of m and $\{\alpha_{m,i}\}$ and is independent of the value of α and can thus be considered as a constant scale factor. Instead of performing a multiplication by this factor, the inverse scaling procedure can be incorporated in the micro rotations [Haviland and Tuszynski, 1980]. For example, the scaling factor K_m can be approximated by

$$K_m = \prod_{i=0}^{p} (1 - m\epsilon_{i,m}2^{-s_{i,m}}) \qquad (E.15)$$

where the values of $\epsilon_{i,m}$ are either 0 or 1. By performing after each micro rotation a scaling cycle

$$\begin{bmatrix} x_{i+1} \\ y_{i+1} \end{bmatrix} = \begin{bmatrix} 1-m\epsilon_{i,m}2^{-s_{i,m}} & 0 \\ 0 & 1-m\epsilon_{i,m}2^{-s_{i,m}} \end{bmatrix} \begin{bmatrix} x_{i+1} \\ y_{i+1} \end{bmatrix} \qquad (E.16)$$

the final vector $[x_n \; y_n]$ will have a norm equal to $[x_0 \; y_0]$. Note that this scaling is only required for the circular and hyperbolic rotations

(m=1,-1). Furthermore, to obey to Eq. (E.11) on a given interval and with equal accuracy, the CORDIC function evaluation time will depend on the system parameter m. To be able to use CORDIC modules in a pipeline fashion the following features are required.

1. The execution time is constant and independent of the parameter m .
2. The hyperbolic and circular rotations have a common function domain
3. The norm scaling factor K_m can be written as a single radix 2 shift $K_m = 2^{-S(m)}$

In [Deprettere et al., 1984] it was shown that these conditions can be met by choosing

$$m^{-\frac{1}{2}}\tan(m^{\frac{1}{2}}\alpha_{m,i}) = 2^{-s_{m,i}} - \mu_{m,i}2^{-r_{m,i}} \qquad (E.17)$$

where $\mu_{m,i}$ is either 0 or 1 and $s_{m,i}$ and $r_{m,i}$ are positive integers. The shift S(m) for the scale factor is -2,0,1 for m= -1,0 and 1 ,respectively.

References

Deprettere, E.F., P. Dewilde, and R. Udo, "Pipelined cordic architectures for fast VLSI filtering and array processing," *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing*, pp. 41A6.1-41A6.4 (March 1984).

Haviland, G.E. and A.A. Tuszynski, "A CORDIC arithmetic chip," *IEEE. Trans. Computers* Vol. **C-29**(2) pp. 68-78 (Feb. 1980).

Volder, J.E., "The CORDIC trigonometric computing technique," *IRE Trans. Electronic Computers* Vol. **EC-8** pp. 330-334 (Sep. 1959).

Walther, J.S., "A unified algorithm for elementary functions," *Proc. of the 1971 Spring Joint Computer Conference*, pp. 379-385 (1971).

F. APPENDIX F: DESCRIPTION OF DEMONSTRATION RECORD

All speech files, except the wide-band examples have been filtered to a bandwidth of 0-3400 Hz. Unless otherwise indicated the predictor parameters were determined according to the parameter settings listed in Table 2.1. The coder analysis parameters for the RPE and MPE coders were listed in Tables 3.1 and 4.1 respectively. Two utterances (see Table F.1) spoken by a male and a female speaker are used in the following examples.

A lathe is a big tool.	(27 = female, 37 = male)
An icy wind raked the beach.	(29 = female, 39 = male)

TABLE F.1. Utterances used for demonstration record.

Different Coder Structures

The following examples have been generated without quantization of the coder parameters.

a) PCM encoded signals (12-bit quantizer, 8 kHz sampling frequency). These utterances were used as coder input in all simulations.

b) RPE coder (pulse spacing 4).

c) RPE coder + pitch predictor (pulse spacing 4).

d) RPE coder (pulse spacing 4, Toeplitz modification RPEM1, see Chapter 3).

e) RPE coder (fixed weighting filter, procedure RPF2, see Chapter 3).

f) MPE coder (8 pulses per 10 ms).

g) MPE coder + pitch predictor (8 pulses per 10 ms).

h) MPE coder signals (Toeplitz modification).

i) Multi-path Search Coder

1. bb version (see Chapter 5).

2. as bb version but without pitch predictor.

j) Speech degraded by additive white noise (SNR=12 dB). Examples played in pairs (RPE-MPE coder).

RPE Coding Combined With Vector Quantization

In the following examples the regular-pulse excitation signal is quantized using scalar and vector quantizers. The quantizer characteristics have been generated with the aid of an iterative procedure. The utterances "27" and "37" were not contained in the training sequence. In all examples the excitation sequence is quantized with 1 bit per pulse. (i.e 1/4 bit per sample).

k) RPE + pitch predictor (iterative code books).

1. scalar L=1.
2. vector L=10 (normalization to the absolute maximum value).

l) RPE + pitch predictor (NSSIZE= 4 ms) (lattice quantizer LQ1, Chapter 7).

RPE and MPE performance at 10 kb/s and 16 kb/s

In the following examples all coder parameters have been quantized.

m) RPE coder 9.6 kb/s (bit allocation as listed in Table 7.1).

n) RPE coder + pitch predictor 10 kb/s (bit allocation in Table 7.1 + pitch parameters).

o) MPE coder 9.6 kb/s (bit allocation as listed in Table 7.1).

p) MPE coder + pitch predictor 10 kb/s (bit allocation in Table 7.1 + pitch parameters).

q) RPE coder 16 kb/s (Table 7.2 , procedure RPE2).

r) MPE coder 16 kb/s (Table 7.2, procedure MPE2).

Wide-band Speech

The wide-band speech examples have been sampled with a 16 kHz sampling frequency (0-7200 Hz bandwidth, 12 bit quantizer). All parameters have been quantized according to the procedures in Table 7.7.

s)

1. PCM
2. RPE 25 kb/s
3. MPE 25 kb/s

HET CODEREN VAN TELEFOONKWALITEIT SPRAAK IN HET TIJDDOMEIN MET BITSNELHEIDEN LAGER DAN 16 KB/S

Peter Kroon

Technische Hogeschool Delft,
Mekelweg 4, 2628 CD Delft

SAMENVATTING

Het gebruik van digitale technieken in communicatie systemen heeft talrijke voordelen. Met name de mogelijkheid om zowel spraak- als controle-signalen met een digitale code te kunnen weergeven, leidt tot systemen met een betere kwaliteit en een hogere betrouwbaarheid dan de traditionele "analoge" systemen. De laatste 15 jaar is er veel onderzoek gedaan naar methodes voor het efficient coderen van spraak signalen, maar veel van de ontwikkelde methodes waren praktisch niet realiseerbaar. De opkomst van de VLSI techniek, maakt het nu mogelijk dat zelfs vrij ingewikkelde methodes geimplementeerd kunnen worden tegen relatief lage kosten.

In dit proefschrift valt de nadruk op tijd-domein technieken welke geschikt zijn voor de codering van telefoonkwaliteit spraaksignalen met bitsnelheden lager dan 16 kb/s. De basis van de onderzochte methode bestaat uit een Vertraagd Beslissingssysteem in combinatie met adapterende voorspellers. De voorspellers verwijderen zoveel mogelijk de in het spraak signaal aanwezige correlatie. De resterende voorspellings fout is vervangen door een zuiniger codeerbaar excitatie signaal, waarbij de juiste keuze hiervan gebaseerd is op een vetraagde beslissing. De keuze van de juiste procedure voor het vinden van het juiste excitatie signaal bepaalt in niet geringe mate het gedrag van de coder.

Een recentelijke in de literatuur voorgestelde methode, de multipuls excitatie (MPE) coder [Atal en Remde, 1982], is diepgaand bestudeerd en diverse modificaties zijn onderzocht.

Tijdens het promotieonderzoek is een alternatieve methode ontwikkeld welke reguliere-puls excitatie (RPE) codering wordt genoemd. De kwaliteit van deze coder is vergelijkbaar met die van de RPE coder, maar de laatsgenoemde bezit een lagere complexiteit en is veel beter geschikt voor VLSI implementatie.

De RPE en MPE methoden kunnen beschouwd worden als een geparametriseerde benadering van het excitatie signaal. Een alternatief is om het meest geschikte signaal te zoeken uit een verzameling kandidaat excitaties. Deze aanpak is wel de meest complexe, maar kan bij een lage bit snelheid beter resultaten geven dan de RPE and MPE methodes. Deze manier van coderen is voornamelijk onderzocht om te dienen als referentie voor de RPE en MPE coders.

Om de coder parameters met een beperkt aantal bits te kunnen weergeven is het noodzakelijk om deze te quantiseren. De quantisatie procedure heeft invloed op de uiteindelijke kwaliteit van het spraak signaal en diverse alternatieven zijn onderzocht. Tevens zijn efficiente procedures onderzocht voor de codering van het MPE excitatie signaal.

Een evaluatie van de diverse onderzocht coderings systemen onthult dat zowel de RPE als de MPE coder bijna telefoon kwaliteit spraak produceren bij 10 kb/s. De kwaliteit van de RPE coder is zelfs bij lage bitsnelheden rond 6 kb/s vrij goed. Het is dan wel noodzakelijk om vector quantisatie technieken te gebruiken. Een ander interessant aspect van de RPE en de MPE coders is dat zij ook zeer geschikt zijn voor de codering van breed-bandige spraak (7 Khz) bij bitsnelheden onder de 32 kb/s.

De RPE methode heeft de meest aantrekkelijke eigenschappen voor VLSI implementatie, en diverse algoritmische structuren worden beschreven. Uiteindelijk is er een geschikte realisatie gevonden, welke gebaseerd is op CORDIC bouwstenen.

Tenslotte beschrijven we de programmatuur die ontwikkeld is voor het uitvoeren van het onderzoek. Hierbij is getracht een software omgeving te ontwerpen die zo algemeen mogelijk is en ook gebruikt kan worden voor diverse andere soorten signaalonderzoek.

OVER DE AUTEUR

Peter Kroon werd geboren te Vlaardingen op 7 september 1957. Na het behalen van het Atheneum-B diploma, heeft hij Elektrotechniek gestudeerd aan de TH-Delft. Na zijn afstuderen in 1981 op het onderwerp signaal verwerkings programmatuur, heeft hij zijn vervangende dienstplicht verricht bij de vakgroep netwerktheorie TH-Delft. De daarbij verrichte werkzaamheden vormden de aanleiding tot dit promotie werk. Naast belangstelling voor de verwerking van audio signalen, gaat zijn interesse ook uit naar de ontwikkeling van software voor complexe systemen. In zijn vrije tijd probeert hij piano te spelen en houdt zich aktief bezig met skieen en parachute springen.

STELLINGEN

1. De RPE methode codeert spraak met een kwaliteit die vergelijkbaar is met die van de MPE methode, maar bezit een geringere complexiteit.

2. Het is niet juist dat het gebruik van een tijd-variërend fout weegfilter essentieel is voor de in dit proefschrift onderzochte coderingssystemen.

3. Het negeren van het verschijnsel pitch bij het ontwerpen van spraakcoderingssystemen leidt tot een inefficient gebruik van de beschikbare hoeveelheid bits.

4. Verbale beschrijvingen van de kwaliteit van een spraakcoderings- systeem zijn ontoereikend indien deze beschrijvingen niet vergezeld gaan van audio materiaal.

5. Ieder voor zich en UNIX voor ons allen.

6. Omdat het verkeersgedrag van de meeste automobilisten sterke overeenkomst vertoont met hun (vroegere) fietsgedrag, is het aan te bevelen om voor een veiliger verkeersbeeld de fietsers meer discipline bij te brengen.

7. Dat de "juiste tijd" ook voor bezitters van een elektronisch uurwerk maar een betrekkelijk begrip is, is bij lezingen e.d. duidelijk hoorbaar op de hele uren.

8. Tekstverwerkings faciliteiten zijn een noodzakelijke maar niet voldoende voorwaarde voor het schrjven van wetenschappelijke publikaties.

9. Om in het treinverkeer de stoptijden op de stations te minimaliseren, is het aan te bevelen om het in en uitstappen alleen toe te staan bij respectievelijk de voor en achterkant van het treinstel.

10. Het feit dat de computer in de wetenschappelijke wereld steeds meer als een normaal hulpmiddel wordt gezien, wordt bevestigd doordat in huidige publicaties de gebruikte computer en bedrijfssysteem niet meer met naam en toenaam worden genoemd.

11. Het verdient aanbeveling om in de toelatingseisen voor een universitaire studie het bezit van een diploma machineschrijven op te nemen.

12. Dat sportparachutisme een gevaarlijke bezigheid is wordt alleen maar beweerd door mensen die liever met beide benen op de grond blijven staan.

13. Het toelaten van geen ABN sprekende omroeppresentatoren dient beperkt te worden tot regionale zenders.

14. De in de universitaire wereld veel gebruikte kreet "wat niet geschreven is, is niet", moet verstaan worden als "wat niet in het Engels geschreven is, is niks".

15. Het gezegde "hardlopers zijn doodlopers" kan steeds meer letterlijk worden genomen, naarmate het aantal trimmers dat s'avonds gebruik maakt van onverlichte wegen e.d. toeneemt.

Peter Kroon

Delft, 21 Mei 1985