# LOW BIT-RATE SPEECH ENCODING FOR DIGITAL MOBILE RADIO

JOSEPH KANU

Doctor of Philosophy

THE UNIVERSITY OF ASTON IN BIRMINGHAM

July 1991

THE UNIVERSITY OF ASTON IN BIRMINGHAM

# LOW BIT-RATE SPEECH ENCODING FOR DIGITAL MOBILE RADIO

BY

JOSEPH KANU

A THESIS SUBMITTED FOR THE DEGREE OF DOCTOR OF PHILOSOPHY

## SUMMARY

The need for low bit-rate speech coding is the result of growing demand on the available radio bandwidth for mobile communications both for military purposes and for the public sector. To meet this growing demand it is required that the available bandwidth be utilised in the most economic way to accommodate more services.

Two low bit-rate speech coders have been built and tested in this project. The two coders combine predictive coding with delta modulation, a property which enables them to achieve simultaneously the low bit-rate and good speech quality requirements. To enhance their efficiency, the predictor coefficients and the quantizer step size are updated periodically in each coder. This enables the coders to keep up with changes in the characteristics of the speech signal with time and with changes in the dynamic range of the speech waveform.

However, the two coders differ in the method of updating their predictor coefficients. One updates the coefficients once every one hundred sampling periods and extracts the coefficients from input speech samples. This is known in this project as the Forward Adaptive Coder. Since the coefficients are extracted from input speech samples, these must be transmitted to the receiver to reconstruct the transmitted speech sample, thus adding to the transmission bit rate. The other updates its coefficients every sampling period, based on information of output data. This coder is known as the Backward Adaptive Coder.

Results of subjective tests showed both coders to be reasonably robust to quantization noise. Both were graded quite good, with the Forward Adaptive performing slightly better, but with a slightly higher transmission bit rate for the same speech quality, than its Backward counterpart. The coders yielded acceptable speech quality of 9.6kbps for the Forward Adaptive and 8kbps for the Backward Adaptive.

## KEY TERMS

**Digital Communications, Low Bit-rate Speech Coding, Mobile Radio Communications, Low Bit-rate Speech Encoding For Digital Mobile Radio, Low Bit-rate Digital Speech Coding techniques.**

## ACKNOWLEDGEMENTS

I wish to express my sincere thanks and appreciation to my supervisor, Dr. R. L. Brewster, for his guidance and assistance throughout this project. I also wish to thank the staff (technical and academic) of the Department of Electrical and Electronic Engineering and Applied Physics for their kind support and assistance, in various ways, during the course of this project.

My special thanks also go to my wife and children for their encouragement and support, and to all my colleagues in the department.

## CONTENTS

## LIST OF TABLES

## LIST OF FIGURES

# CHAPTER ONE

# INTRODUCTION

The aim of this project is to investigate methods for digitally coding speech at low bit rates and to propose a suitable method for telephony over mobile channels, which maintains the necessary speech qualities such as naturalness and speaker recognisability.

The need for low bit-rate speech coding is the result of growing demand on the available radio bandwidth for mobile communications both for military purposes and for the public sector.

In Mobile Communications it is necessary to utilise the available bandwidth in the most economic way so as to accommodate more services to meet the growing demand from both the military and public sectors [1]. Current research trend in low bit rate digital speech coding takes advantage of successes in the field of digital electronics, and exploits the advantages of digital communications such as noise-free transmission, ease of regeneration, a network that is cheaper to manage and maintain, and increased channel capacity. However, low bit rate speech coding raises its own problems of system complexity and loss of speech quality. This is due to the fact that low bit rate speech coding makes use of the redundant nature of the speech waveform and, by removing all or some of the redundancies, the remaining speech parameters can be coded with fewer bits ( binary digits ). However, this also results in a more complex system and possibly the loss of certain speech qualities such as accent, speaker recognisability, and naturalness as, for example, is frequently experienced in vocoders. In general, the more redundancy that is removed from the signal before encoding, the lower the transmission bit rate but also, the more complex the coding system will be and the more impairment there will be to the speech quality [1]-[4].

Office automation requires that the available telephone network be used to handle data as well as speech. This can only be achieved in an economic way by using the same transmission and switching methods for both speech and

data. However, inherent differences do exist between speech and data [2]. Basically speech is continuous in nature while data is digital in nature. This implies that the speech signal has to undergo a digitisation process to convert it into the same discrete form as data. To accommodate digital speech on a bandwidth equivalent to that required to handle analogue speech signals, it is necessary that the speech signal be coded at a low bit-rate.

A waveform that is continuous in nature, such as the speech waveform, can be digitised as follows [2]-[5]:

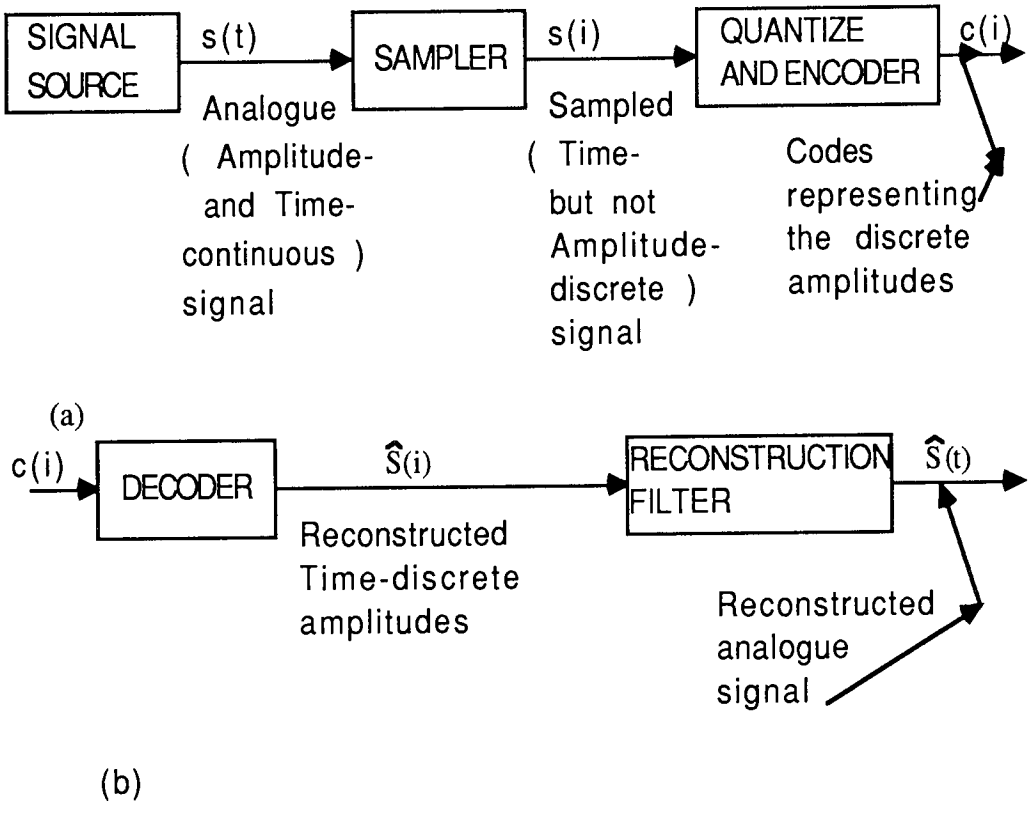



Fig.1.1. Digitisation and Reconstruction processes of an analogue waveform;

(a) Digitisation process

(b) Reconstruction process

The signal source generates the analogue signal. In telephony, the speech signal is generated by the microphone in the mouthpiece of the telephone handset. The generated signal is continuous in both amplitude and time. The digitisation process starts with the sampler, where the signal is made discrete in time. The block labelled SAMPLER in fig.1a consists of an electronic switch that passes amplitude samples of the analogue signal at equal intervals of time.

The process is known as sampling and, provided that the signal is sampled often enough, it can be reconstructed at the receiver by reassembling the samples and filtering. Thus it is not necessary to transmit the analogue signal in its entirety. The sampling theory states that the minimum rate at which an analogue signal must be sampled to enable its reconstruction without distortion is twice the bandwidth of the analogue signal. This minimum sampling rate is known as the Nyquist sampling rate.

Often it is necessary to sample a signal at a rate higher than the Nyquist rate to allow for the reconstruction filter imperfections and to avoid distortion in the reconstructed signal. This distortion, known as aliasing distortion, is caused when frequency components from higher bands of the signal spectrum are present in the reconstructed low band signal. Basically, sampling replicates the spectrum of the sampled signal on the frequency axis. The signal is recovered by separating the low band spectrum from the high bands and the spectra have to be sufficiently spaced apart to enable distortionless reconstruction. To avoid aliasing it is usual to bandlimit the analogue signal before sampling. This ensures that upperband frequencies are not present in the sampled signal. Bandlimiting is achieved by low-pass filtering the signal. Hence, a low-pass filter will normally be introduced between the signal generator and the sampler. Speech is normally bandlimited at 0 to 3kHz and sampled at the rate of 8kHz, which is about 2kHz above the Nyquist rate. This creates a guardband of 2kHz in the frequency spectrum, allowing for a gentler slope, and hence a less expensive reconstruction filter. The reconstruction filter is another low-pass filter that gets rid of all spurious frequency components in the output signal. Thus, provided the signal is properly bandlimited and sampled at a rate greater than the Nyquist rate, the signal can be reconstructed at the receiver with very little discrepancy, simply by low-pass filtering the stream of samples.

The next stage in the digitisation process is the quantization process. The output of the sampler consists of a stream of samples of the original analogue signal taken at regular intervals of time. These samples are continuous in amplitude and are thus still analogue in nature, exhibiting most of the

characteristics of analogue signals. These include susceptibility to noise and distortion, cross-talk and pick-up.

The quantizer, labelled QUANTIZER in fig.1a, assigns a finite amplitude level to each sample value. These amplitude levels are predetermined and are approximations of the real sample amplitudes. Each sample value is measured against a range of amplitude levels and the level nearest to the sample value is assigned to the sample value. These amplitude levels are coded by assigning to each a code word of binary digits ( bits ) and the codes transmitted. In digital telephony, a codeword normally consists of eight bits. At the receiver, the codes are decoded into their respective amplitude levels. A method by which analogue signals are transmitted by digital means, as described above, is known as Pulse Code Modulation ( PCM ). Variations of the technique are being researched into, as in this project, to achieve economies, in bit rate and in system complexity, without trading off signal quality. Pulse Code Modulation, while adequate for toll quality telephony, is wasteful on bandwidth.

Because of the quantization process, the reconstructed signal can never be a true replica of the original signal. The discrepancy is known as quantization error (noise) and is due to the fact that quantization is an approximation process. This error is inherent in the digitisation of analogue signals. In addition to the quantization error, another drawback is the increased bandwidth required to transmit the digital signal.

However, the advantages of digital communications are overwhelming and more than make up for the drawbacks discussed above. These include immunity to noise, ease of regeneration and economy of transmission in that repeater stations can be situated farther apart than for analogue transmission. Also, the repeaters, and also the receiver, are simpler and cheaper than their analogue counterparts. This is due to the simpler nature of a digital signal as compared to an analogue signal. A digital signal consists of codes made up of the binary symbols one and zero. Thus, all that is required of a repeater or receiver upon receiving a code word is a decision of which of the two code units, one or zero, was transmitted. In this way, it is possible to reconstruct a signal transmitted by

digital means without additional noise effects due to the transmission path.

On the other hand, analogue signal repeaters and receivers are concerned with faithfully reproducing the transmitted waveform. This requires more complex and expensive receiving equipments. Also, since analogue signals are noise-prone, faithful reproduction implies that the signal is reproduced with its noisy component. This makes noise-free reception of analogue signals difficult, if not impossible. In addition, analogue receivers consist of amplifying devices which are made of electronic components. These components themselves are natural generaters of noise, which further complicates the noise issue in analogue communication devices.

Apart from the drawbacks of digital coding of analogue signals in general, low bit rate digital coding of analogue waveforms is a complex process giving rise to more complex systems. In general, the lower the bit-rate, the more complex the coding algorithm will be and hence the more complex the coding system will be. Also, the signal quality tends to degrade further as the bit-rate is reduced. This is due to the fact that in low bit-rate coding less binary digits are normally assigned per sample value as it is not possible to reduce the sampling frequency below the Nyquist rate.

However, advances in digital technology continue to make possible more large-scale integration, thus enabling the construction of more complex equipment with fewer and relatively cheaper components. Thus, cost is becoming less of an impediment in low bit-rate signal coding. Also, as will be demonstrated in this project, it is possible to achieve low bit rate signal coding in a relatively simple and cheap way by taking advantage of certain coding algorithms.

This project makes use of the simple nature of Delta Modulation and the efficiency of Predictive Coding. Delta Modulation (DM) is a special case of Differential Pulse Code Modulation (DPCM), a variant of Pulse Code Modulation in which the difference between a sample value and its reconstructed predecessor is quantised and encoded. Since the difference value will normally be smaller than the actual sample value, the difference signal can

be encoded with less binary digits than will be required to encode the sample value itself. In Delta Modulation the difference signal is encoded with one binary digit, and takes the value 0 or 1 according as whether the difference value is negative or positive. A sample value is compared with its reconstructed predecessor and a binary one transmitted if the sample value is greater than the reconstructed predecessor, or a binary zero transmitted if it is less than the reconstructed predecessor. The result is a significant reduction in bit-rate compared to Pulse Code Modulation ( 8kbits/s for speech sampled at 8kHz compared to 64kbps in the case of PCM for speech sampled at the same rate ) [4]-[6].

This very coarse quantization ( one bit per sample value compared to eight bits per sample value for PCM ), however, poses a serious impediment in the use of Delta Modulation. Further, the resulting signal quality is generally poor compared to Pulse Code Modulation. Also, since Delta Modulation relies on the faithful prediction of a sample value from its reconstructed predecessor, a sampling rate far greater than the Nyquist rate is required to enhance the necessary sample-to-sample correlation. For communications quality speech, the sampling rate for a Delta Modulator is normally six times the Nyquist rate. This compares with Pulse Code Modulation which requires a binary digit rate of eight times the Nyquist rate.

Notwithstanding these draw-backs, Delta Modulation remains a strong contender for low bit-rate signal coding. This is due largely to the simplicity of its algorithm and the equipment needed for its implementation. Delta Modulation is the simplest of the Pulse Code Modulation variants. A simple integrator circuit is used to reconstruct a transmitted sample value, and quantization and encoding are accomplished by a simple comparator circuit. Methods are being sought to enable the effective implementation of Delta Modulation. In this project predictive coding is used with Delta Modulation, thus taking advantage of the simple nature of Delta Modulation and the efficiency of Predictive Coding as a low bit rate coding technique.

Predictive Coding makes use of the sample- to -sample correlation in speech

signals sampled at a reasonably high frequency ( above the Nyquist rate ). This high correlation between samples makes it possible for a sample value to be "predicted" from past sample values over a fairly long sampling period. In predictive coding of speech signals, a prediction is made of a speech sample at the transmitter. In this project, the prediction is done over four sampling periods. The predicted sample value is subtracted from the actual sample value. The difference value is quantised, encoded, and the codes transmitted. At the receiver, a similar prediction is made using past received sample values. This predicted value is added to the corresponding received difference value. The original signal is thus reconstructed.

Here again the transmission of a difference value rather than an actual sample value makes it possible for a sample value to be encoded with less binary digits without a significant loss in signal quality. The number of bits needed to adequately encode a difference value depends on the efficiency of the predictor. The more efficient the predictor, the smaller the difference value will be and hence the less binary digits required to adequately encode it. In this project, the difference signal is delta modulated . In this way the signal quality no longer depends on making the sampling frequency larger, as is the case in Delta Modulation alone, but on the efficiency of the predictor at both the transmitter and the receiver, while taking advantage of the simplicity of the Delta Modulation technique.

The efficiency of the predictor is improved by periodically updating the predictor parameters. This can be done either instantaneously, on a sample-by-sample basis, or periodically, once every several sampling periods. Both methods are experimented with in this project.This prediction technique in which the predictor parameters are updated regularly is known as adaptive prediction and takes into consideration the nonstationary nature of the speech waveform. This will be discussed further in the next chapter.

Quantization noise ( error ) was mentioned earlier as an inherent drawback of the digitisation process. In Delta Modulation, quantization noise takes two forms: ( i ) Slope Overload, when the quantization step size is too small to

enable the reconstructed signal to catch up with the input signal, and

(ii) Granular Noise, when the quantization step size is too large, causing the reconstructed signal to deviate further from the input signal. ( refer to Fig.1.2 below)

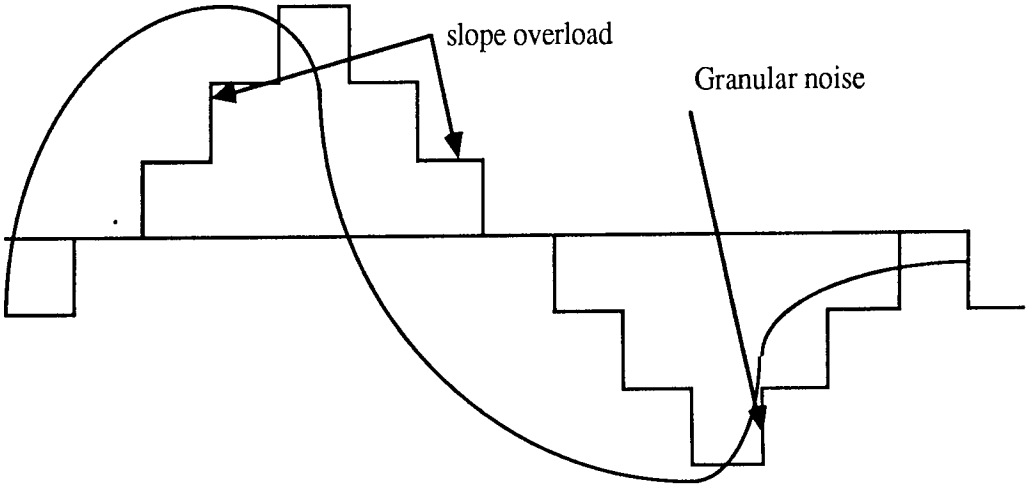



Fig.1.2 Slope overload and granular noise of delta modulation system [4].

The quantization step size is the amount by which the output signal increases or decreases to keep in step with changes in the input signal level. When the step size is fixed it may be either too large or too small, giving rise to either of the two discrepancies discussed above. A Delta Modulator that uses a fixed step size is known as a Linear Delta Modulator ( LDM ).

To help overcome the above discrepancies, the step size may be varied according to variations in the input signal level [6]. This, in general terms, is known as adaptive quantization and the Delta Modulator that uses variable stepsize is known as an Adaptive Delta Modulator ( ADM ). Judgment on how to adapt the step size is based on succeeding output conditions. In Delta Modulation, judgment is based on succeeding output binary digits. A succession of the same binary digits ( ones or zeros ) implies slope overload has occurred. The step size is then increased so as to enable the output to catch up with the input. On the other hand, a block of opposite binary digits implies granular noise and again the step size is adjusted accordingly. In speech communication, slope overload is more annoying than granular noise. Hence,

in general, stepsize increases are made faster than step size decreases.

**Chapter Two** discusses speech coding techniques. These techniques can be grouped under two broad categories of waveform coders and vocoders.

**WAVEFORM CODERS ( Section 2.1 )**

The coding technique of this category is concerned with faithfully reproducing the speech waveform. Pulse Code Modulation (PCM) and its variants fall under this category of speech coders. They involve straightforward analogue-to-digital and digital-to-analgue conversion and are noted for their hardware and software simplicity.

However, conventional Pulse Code Modulation is wasteful of transmission bit rate and, hence, transmission bandwidth. Consequently, several techniques are being used to obtain a better bit-rate economy. These techniques include the two variants of Pulse Code Modulation, Differential Pulse Code Modulation (DPCM) ( **section 2.1.2** ) and Delta Modulation (DM) ( **section 2.1.3** ). These two techniques are further improved by updating their parameters on a regular basis, resulting into their adaptive versions ( ADPCM and ADM, ) respectively.

Another technique used to improve on Pulse Code Modulation is companding, by which the speech signal is first compressed, then passed through a uniform PCM system. The PCM signal is then expanded again at the output. This ensures that the more common low amplitude values are quantised more finely than the less frequent high amplitude values.,

In general, techniques used in low bit rate speech coding are based on a redundancy removal principle. The speech signal has a lot of redundancy some of which can be removed using appropriate techniques without much effect on the resulting speech quality. Most exploited is the sample-to-sample correlation, which can be high enough to enable a speech sample value to be fairly accurately predicted from several sample values preceding it. Differential Pulse Code Modulation (DPCM) is one of the techniques based on the sample-to-sample correlation property of speech. In this technique, the predicted sample value is subtracted from the actual sample value and the

difference is quantised and encoded for transmission. In this way, the difference signal could be encoded with less bits than required for the actual sample value, thus giving a lower transmission bit rate. Delta Modulation is a special case of DPCM in which the difference signal is encoded with one bit. Another technique based on the correlation property is Predictive Coding (PC) (**section 2.1.5**), which is actually an extension of DPCM. The two coding techniques differ mainly in that Predictive Coding uses more past sample values than DPCM in the prediction of a sample value, (typically 10 compared to 4 for DPCM). Also, Predictive Coding exploits another redundancy in addition to the sample-to-sample correlation. This redundancy has to do with the quasi-periodicity of the speech signal. It involves pitch-to-pitch correlation and it is known as long-term redundancy while sample-to-sample correlation is called short-term redundancy. These are the two most widely exploited speech redundancies in speech coding.

Obviously, predictive coding yields a smaller bit rate than DPCM for the same speech quality but this is achieved at the expense of system complexity. Its performance can be improved, as in Differential Pulse Code Modulation, by periodically updating the predictor parameters. The resulting technique is called Adaptive Predictive Coding (APCM).

Another solution to the high bit rate problem of conventional Pulse Code Modulation is Sub-band coding (section 2.1.6). This involves dividing the frequency spectrum of the speech waveform into frequency bands by passing the signal through a bank of band-pass filters. Any of the coding techniques discussed above, ADPCM, DM, or APCM, can be used to separately encode the output from each band-pass filter. The individual bit-streams are then multiplexed and transmitted. At the receiver, they are demultiplexed, decoded, and the signal reconstructed by combining the decoded values. This technique makes use of the non-uniform nature of the speech spectrum in which the more frequent low amplitudes are encoded more finely than the less frequent higher amplitudes. A draw back in sub-band coding is system complexity in the number of filters involved, which can also add to the distortion through

electronic noise.

## VOCODERS ( section 2.2 )

These category of coders, unlike waveform coders, are concerned only with the information-bearing parameters of the speech signal and not with reproducing the speech waveform. What is more, they are speech specific as they are based on techniques that are a simulation of the speech producing organs of the human mouth, mainly the vocal tract. On the other hand, except for Predictive Coding, the waveform coding techniques are based on the general principle for digitising signals; analogue-to-digital conversion and digital-to-analogue conversion, by which any bandlimited signal can be converted to digital form and then reconverted to analogue form.

In general, vocoders are more complex than waveform coders and their use is limited to applications where most of the speech qualities common in commercial telephony and for communication purposes, such as speaker recognisability, speaker accent, etc., are not necessary. Such applications include precoded messages for answering machines used in telephone networks, airline booking desks, and in toys. Lacking some naturalness, the resulting speech is often synthetic sounding.

There are three basic types of coders in this category. They are channel vocoders (**section 2.2.1**), format vocoders (**section 2.2.2**), and linear predictive coders (**section 2.2.3**). In general, vocoder techniques are based on a further exploitation of the redundancies in the speech waveform. They differ in the method employed to exploit the redundancies.

Linear Predictive Coding is the most popularly used of the three vocoding techniques. This is reflected in the several variations of the coder that is a result of the bid over the years to improve on the performance of the technique. Its main advantage over the others in its category lies in its time-domain nature. This means that the algorithm is based on a time-domain simulation of the the speech production mechanism, thus making the technique easier and more straightforward than the other two, which are frequency-domain. The main

variations of Linear Predictive Coding are Multipulse Excited Linear Predictive Coding (MPLPC) discussed in section 2.2.4, and Code Excited Linear Predictive Coding (CELP) discussed in section 2.2.5.

Methods of assessing the performance of low bit-rate coders are discussed in section 2.3. The received speech quality of low bit-rate coders is less straightforward to assess than for the waveform coders discussed in section 2.1. This is due mainly to the diverse nature of the distortions in these coders. Their performance is by nature input-dependent. Up to date, the most effective method of assessing the performance of a low bit-rate coder is the subjective technique of listening.

Section 2.4 discusses current trends in low bit-rate speech coding. These are based on algorithms that combine aspects of waveform coding with those of vocoding. The result is a coder that reaps the benefits of both coding techniques - better speech quality with minimum system complexity, and low bit-rate. This project is based on this hybrid concept (**section 2.5**).

**Chapter Three** discusses the method of the project. As mentioned above, the project is based on the current hybrid trend in which certain aspects of waveform coding are combined with those of vocoding. Delta Modulation and linear predictive coding are used thus exploiting the effective coding technique of LPC and the simplicity and robustness to transmission errors of DM.

Section 3.1 gives a background description of the combined algorithm. Section 3.2 describes the hardware and software layout of the system. The system, like any communication system, consists of three basic functional parts, the input, the transmitter, and the output, implemented by three separate hardware devices and the respective software instructions that direct their functions. The devices and the programs used in implementing the system blocks are described in section 3.3. The input is a data acquisition device which accepts the speech signal from a telephone handset connected to it, converts it to digital samples and buffers the digital data. It is connected to an Opus PC III personal computer through the computer's serial port. A program written in a high-level language reads the digital data and stores it in a file in the computer's

secondary storage. Another program reads and processes the data thus stored, and stores the processed data in a different file. This second program makes up the main algorithm of the project and is basically a data compression algorithm.

The output device consists basically of a digital-to-analogue system card designed for the purpose of this project and connected to the computer through an 8255 parallel peripheral interface (ppi) card. A third program reads the processed file sample-by-sample and outputs them to the digital-to-analogue system through the ppi interface card. A loudspeaker that is part of the output device system generates the speech sound. The functional parts ( hardware and software ) of a microcomputer system, the main functional device in the project, are discussed in section 3.4.

**Chapter Four** discusses the two coding techniques adopted in this project. In both techniques data compression is achieved by making a prediction of a speech sample from four past reconstructed samples (i.e. a predictor of order 4), subtracting the predicted sample from the actual sample, and outputting a binary one or zero depending on whether the difference value is positive or negative, respectively. At the receiver, the transmitted code (1 or 0) is received and the difference value reconstructed from one of two step sizes depending on which code was received. At the same time, a prediction of the transmitted sample is made using the same method as at the transmitter and this predicted value is then added to the reconstructed step-size.

To ensure accurate prediction of a sample, the predictor coefficients are updated periodically. The two techniques used differ in their method of updating the coefficients.

In the first technique, the predictor parameters are updated once every one hundred sampling periods ( **section 4.2** ). This involves dividing the entire speech signal into blocks of 100 samples each. Starting with the first block in the sequence, each block is transmitted separately by first extracting from it the predictor coefficients. These coefficients are then used to predict each sample in the block, starting with the first. The next block is then processed in the same way as the one preceding it. This is continued until the whole speech signal has

been coded thus. This method of updating the predictor coefficients is known as syllabic and the coding technique based on this method of updating predictor coefficients is known as the Forward Adaptive Coder.

This technique has two disadvantages; it involves a delay of at least the length of period involved in updating ( 100 sampling periods in this case ) and increased bit rate as the predictor coefficients for each block of samples are transmitted as side information. Thus the more often the coefficients are updated, the greater the added bit rate due to side information. On the other other hand, the predictor performance can be enhanced by using shorter updating periods. This dilemma is overcome by a compromise between increased bit-rate on the one hand and predictor efficiency and delay on the other hand. On the plus side, its algorithms are uncomplicated computation-wise. This is due to the fact that the predictor coefficients are extracted using the real samples and not their quantised, reconstructed versions. This also ensures that the coefficients thus computed are not corrupted by noise before they are used in the prediction.

The method of extracting the coefficients is also discussed in this subsection. The autocorrelation method used in this project is described. This method is simple and straightforward, based on the linear prediction representation of the speech producing organs of the human mouth. The autocorrelation matrix is solved using Durbin's recursive technique. This technique computes the predictor coefficients via intermediate parameters known as the reflection coefficients. These latter set of coefficients are the ones best suited for the purpose of transmission. The same recursive technique is used at the receiver to compute the predictor coefficients from the received reflection coefficients.

The other technique ( **section 4.3** ) computes a new set of predictor coefficients every sampling period, based on signal information at the output. Hence the main disadvantage of this technique is that the coefficients are extracted from quantised, reconstructed versions of the actual samples, thus increasing the chance of the signal being corrupted by noise. Another

disadvantage of the technique is in the complexity of its algorithm. To minimise the amount of computations involved, which is an indication of the system complexity, an approximation mean-square error technique involving the signs of the error signal and the output data, rather than their actual values, is used in computing the predictor coefficients. This sign(error)-sign(data) technique is best suited to the binary coding system used in this project. The corresponding coding technique is known as Backward Adaptive Coding. Its main advantage over its forward counterpart is reduced transmission bit-rate as it does not involve transmission of the predictor coefficients. They can be computed at the receiver simply by duplicating the method of the transmitter. This duplication of the transmitter functions at the receiver, plus the fact that the predictor coefficients are updated every sampling period, accounts for the coding system's complexity.

The quantizer step size is also adapted in accordance with time-variations of the speech signal. Two methods of doing this, Forward Quantizer Step-Size Adaptation ( **section 4.2.2** ) and the backward technique ( **section 4.3.2** ) are discussed. However, only the backward method is adopted in this project. This facilitates evaluation of the two coders against each other. Also, and more importantly, the backward technique is better suited to the binary nature of the quantizer. The step-size is increased or decreased based on the output code. This also facilitates the computation of the step-size at the receiver, thus further minimising the amount of side information transmitted.

The sequence of computer instructions that implement the two coding techniques are described in section 4.4, together with flow chart and block diagram representations of the two coding processes.

**Chapter Five** describes the experimental work carried out as part of this project and compares the performances of the two coders based on the two coding techniques used in the project. In general, both coders were rated as satisfactory. However, the one based on the forward adaptive algorithm performed slightly better than that based on the backward adaptive algorithm, though at a slightly higher bit-rate. Performance rating was based on subjective

listening tests carried out during the experiment and on the coder's ability to reproduce the input speech waveform as indicated by the output waveforms.

**Chapter Six** discusses the achievements and shortcomings in this project. Among the shortcomings encountered during the course of the project were limited computer memory, the lack of a mathematics coprocessor in the computer, and the inherent inability of the input device to accept and input more than one word of speech at a time. This latter drawback made it impossible to process sentences rather than just single words. Although an attempt was made to investigate the performance of the coding algorithms when used to code more than one word by merging word files to construct a "sentence", this could not serve as a true performance test of the coders in this respect for two reasons; first, the "sentences" thus constructed were not true sentences in the true sense of the word and, secondly, the limited computer memory posed a limit on the number and size of words that could be combined.

The lack of a mathematics coprocessor was more apparent in computation-intensive algorithms like the backward adaptive algorithm which was significantly slow although it was based on an approximation method. More accurate methods such as the recursive least squares ( RLS ) method could not be experimented with as these are known for the complications of their computations. However, the approximation method was quite adequate and not without other advantages.

The chapter concludes with recommendations for further development. Among these recommendations are the need to increase the computer's computational power by increasing its memory capacity from its present value of 640k bytes to at least one megabyte, and by installing a mathematics coprocessor to take over floating point operations from the central processing unit ( CPU ). Also, to enable the processing of sentences rather than single words, it is recommended that a proper input device which communicates with the computer as part of its input devices be used in place of the present data acquisition device. This also will eliminate the need for a separate input program to link the computer and the device. The speech signals input by this device for

processing were already impaired by undertones from the telephone handset attached to the device and into which the words were spoken. The use of a proper input device will eliminate this.

# CHAPTER TWO

# LITERATURE REVIEW

## 2.0 INTRODUCTION

The subject of voice digitisation can be broadly classed into two categories[4]:

(i) Waveform coding, which involves encoding the speech signal by means of a straightforward reconstruction of the acoustic time waveform. This category of voice digitisation involves simpler coding algorithms, and hence coders, and yields good to high quality speech. Four basic techniques are used in this category, pulse code modulation (PCM) and its two derivatives, differential pulse code modulation (DPCM) and delta modulation (DM), and sub-band coding. These techniques are based on the principle of analogue-to-digital and digital-to-analogue conversion, which accounts for their simplicity. They are, however, handicapped by a higher bandwidth requirement inherent in the digitisation process, and a corresponding high bit-rate, making them unsuitable for bandwidth conservation situations such as in mobile communications, where there is need to economise the available radio bandwidth. Techniques in this category of voice digitisation are more useful for commercial digital telephony.

(ii) The other category of voice digitisation is vocoding, by which only the information bearing aspects of the speech signal are processed and transmitted. This category is characterised by more complex coding algorithms and hence coders. As only the information bearing aspects of the speech waveform are coded for transmission by this category of speech coders, the resulting speech output is synthetic in quality and contains none of the speech qualities, such as accent, speaker recognisability and naturalness, inherent in normal conversational speech. They are therefore unsuitable for applications, such as commercial telephony, where greater signal quality is required. However, they can produce very low bit-rate speech and are suitable for applications where

bandwidth is at a premium and speech quality is not of great importance.

However, current practice in digital speech coding is to combine some aspects of waveform coding techniques with those of vocoders to produce coders that combine the good speech quality of waveform coders with the low bit-rate of vocoders. Such coders can produce good quality speech for communications purposes at comparatively low bit rates.

In turn, each of the two broad categories of voice digitisers can be classed as either a time-domain or frequency-domain coder, according to whether the digitisation process is done in the time- or frequency-domain. Again, each of these latter categories has its merits and demerits and are more suitable for some applications than for others.

In general, voice digitisation is based on the exploitation of redundancies inherent in human speech and on the perceptive limitation of the human ear. Due to this limitation in human perception, speech spectral content above 4kHz is not necessary for intelligibility. Thus, for telephony, the speech waveform can be bandlimited to 4kHz without much loss in its quality. Thus the sampling principle, which requires the signal to be bandlimited before sampling, can be applied to the speech signal. Sampling precedes digitisation.

In this chapter, the voice coding techniques mentioned above are discussed in detail, including the merits and demerits of each category. This is followed by a discussion of current trends and achievements in the field of low bit-rate speech coding. Since the performance of a speech coder is accepted or rejected on the basis of the subjective views of its possible future users, speech coder assessments based on subjective evaluations are discussed in this chapter as well as objective techniques of evaluation.

## 2.1 WAVEFORM CODERS [4]-[14]

These coders encode speech by means of a straightforward reconstruction of the acoustic time waveform. The techniques used are the closely related discrete-time, discrete-amplitude representations known as pulse code modulation (PCM), differential pulse code modulation (DPCM), and delta

modulation (DM). These were mentioned in chapter one and will now be considered in detail.

### 2.1.1 Pulse Code Modulation (PCM)

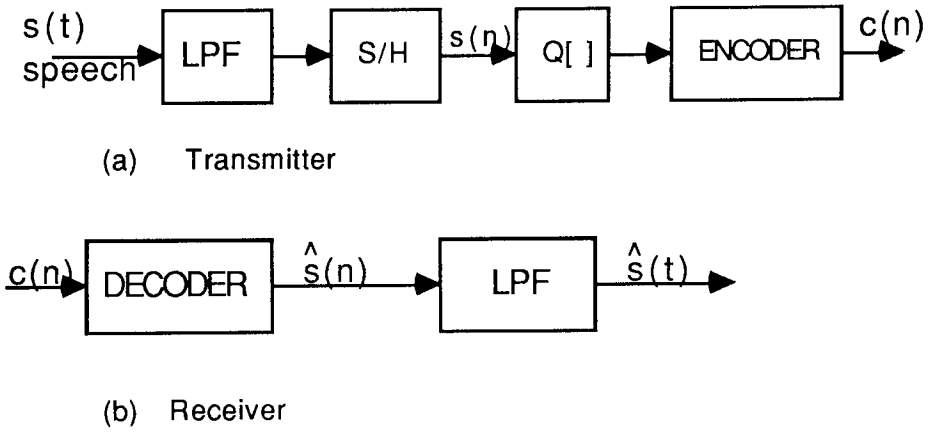



Fig.2.1 Block Diagram of Pulse Code Modulation

Fig.2.1 illustrates the principle of pulse code modulation.

(a) The analogue waveform, s(t), is bandlimited by passing it through a low-pass filter with cut-off frequency w, which is the highest frequency we wish to retain in the analogue signal. In the case of speech, this frequency is 3400Hz.

(b) The bandlimited waveform is sampled at a rate at least 2wHz. This rate, known as the Nyquist rate, is the minimum rate required for perfect reconstruction of the analogue signal.

(c) The amplitude of each signal sample is quantised into one of $2^B$ levels. This yields an information rate of B bits per sample and an overall information rate of 2wB bits per second (bits/s).

(d) The discrete amplitude levels are represented by distinct binary words of length B. This process of representing discrete-time, discrete-amplitude signal levels by distinct binary words is known as encoding. In practice, quantization and encoding are together referred to as analogue-to-digital conversion (A/D

conversion) and are performed in one single block.

(e) The binary code words are transmitted and, assuming a perfect channel, the binary words are mapped back into amplitude levels at the receiver, through the reverse process of digital-to-analogue conversion (D/A conversion).

(f) The amplitude-time signal is low-pass filtered with a filter whose cut-off frequency is w, to reconstruct the analogue signal.

Digital coding of waveforms as analysed above suffers two main drawbacks; increased signal bandwidth and an inherent discrepancy between the input analogue signal and its reconstructed version. The latter drawback is due to the fact that quantization is an approximation process, whereby the sampled amplitude values are approximated into definite amplitude levels. This is illustrated in Fig.2.2 below for an eight-level (i.e., B = 3) uniform quantizer [5]. The amount by which the reconstructed signal deviates from the input signal is known as quantization noise (error).

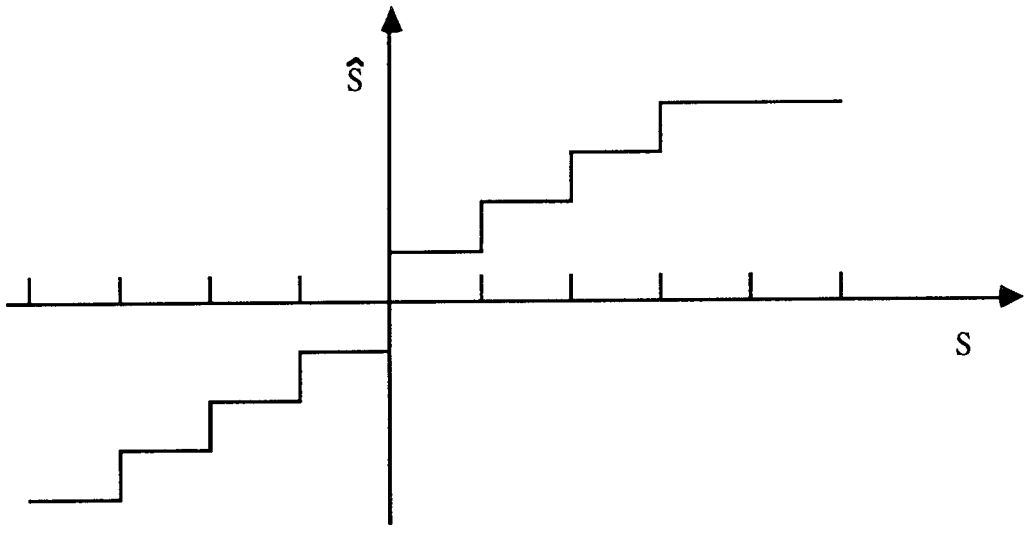



Fig.2.2 The uniform quantizer characteristics

The distance between two adjacent quantization levels is known as the quantization step for that interval. For uniform quantization, the quantization step is the same for all quantization intervals as illustrated in Fig.2.2 above. All sample values that fall in a particular quantization interval are represented by a

single discrete value located at the centre of the quantization interval. Quantization noise is minimised by setting a large number of small quantization intervals (i.e. small stepsize). This however, results in a corresponding increase in the number of bits to uniquely identify the quantization intervals. Thus, in digitising a signal, a trade-off has to be made between signal quality and increased transmission bit-rate and, hence, bandwidth.

Denoting the quantizer stepsize as $\delta$, and if the number of quantizer levels is large, the quantization error may be assumed to have the following uniform distribution:

$$p(E) = \frac{1}{\delta}, \quad \frac{\delta}{2} \leq E \leq \frac{\delta}{2} \qquad 2.1$$

and the mean square value of the quantizer error from this expression is

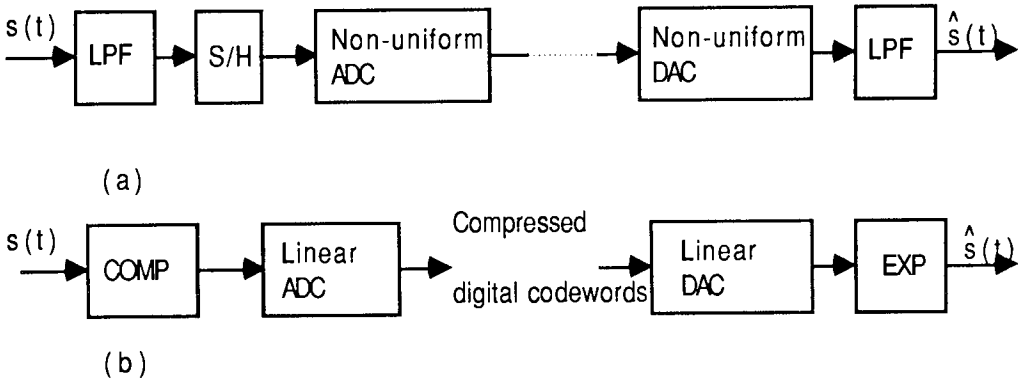
$$\int_{-\delta/2}^{\delta/2} E^2 p(E)dE = \frac{\delta^2}{12} \qquad 2.2$$

It was also mentioned in chapter one that pulse code modulation itself was a waste of bit-rate. For telephony, the sampling frequency is normally 8kHz, and each sample is encoded with eight binary digits, resulting in an overall transmission bit-rate of 64kbits/s.

In addition, uniform pulse code modulation encodes all sample values of a given analogue signal in the same way. This is not needed in the case of speech, which exhibits a nonuniform amplitude distribution, lower amplitude sample values being more common than higher amplitude sample values. Hence, the coded space in a uniform pulse code modulation system is very inefficiently utilised.

A more efficient coding procedure is achieved if the quantization intervals are not made uniform but are allowed to increase or decrease with the sample

value. The idea is to quantise small sample values more finely than large ones. This can be achieved either by passing the sample values through a nonuniform quantizer, which assigns small quantization intervals to small samples and large quantization intervals to large sample values, or by a technique called companding. In this method, the signal is compressed at the input and the compressed signal quantised using a uniform quantizer. At the receiver, the decoded sample values are then passed through a reverse process to expand them. (See Fig.2.3 below)




(a) Nonuniform Quantizer Method

(b) Compression Followed by Uniform Encoding/Decoding, Then Expansion Technique

Fig.2.3 Non-Linear Pulse Code Modulation

Two types of companding laws are currently in use in telephone speech quality pulse code modulation. These are [4]:

(i) The μ−law, used in North America and Japan. The compression process of this law is expressed as follows:

$$F_\mu(x) = \mathrm{sgn}(x)\frac{\ln(1+\mu|x|)}{\ln(1+\mu)} \qquad 2.3$$

where x is the input signal amplitude (-1< x< 1), sgn(x) is the polarity of x, and μ is a parameter used to determine the amount of compression.

Because of expression (2.3 ) above, companded pulse code modulation is sometimes referred to as Log-PCM (Logarithmic pulse code modulation). A logarithmic compression curve is ideal in the sense that quantization intervals, and hence quatization noise, are proportional to the sample amplitude. This is illustrated in Fig.2.4 below.

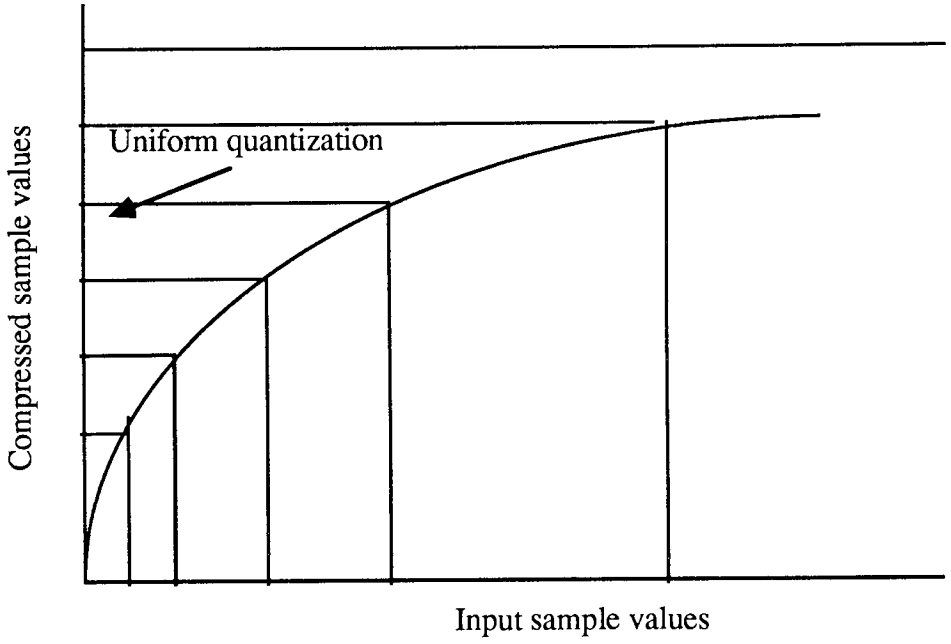



Fig.2.4 Typical compression characteristic.

The inverse or expansion characteristic of a μ–law compandor (compressor-expander) is given by

$$F_{\mu}^{-1}(y) = \mathrm{sgn}(y)\left(\frac{1}{\mu}\right)\left[(1+\mu)^{|y|} - 1\right] \qquad 2.4$$

where y is the compressed value,$y = F_{\mu}(x)$, $(-1 < y < 1)$, sgn(y) is the polarity of y, and μ is the companding parameter. It has been found that $\mu = 255$ gives a quite good performance and this is the value commonly used [5]

The other companding law is known as the A-law, recommended by the International Consultative Committee for Telephony and Telegraphy (CCITT). The normalised compression characteristic of this law is defined as

$$F_A(x) = \operatorname{sgn}(x)\left(\frac{A|x|}{1+\ln(A)}\right) \quad 0 \le |x| \le \frac{1}{A}$$
$$= \operatorname{sgn}(x)\left(\frac{1+\ln|Ax|}{1+\ln(A)}\right) \quad \frac{1}{A} \le |x| \le 1 \qquad 2.5$$

and the inverse or expansion characteristic is defined by

$$F_A^{-1}(y) = \operatorname{sgn}(y)\frac{|y|[1+\ln(A)]}{A} \quad 0 \le |y| \le \frac{1}{1+\ln(A)}$$
$$= \operatorname{sgn}(y)\frac{\left(e^{|y|[1+\ln(A)]-1}\right)}{A} \quad \frac{1}{1+\ln(A)} \le |y| \le 1 \qquad 2.6$$

where $y = F_A(x)$

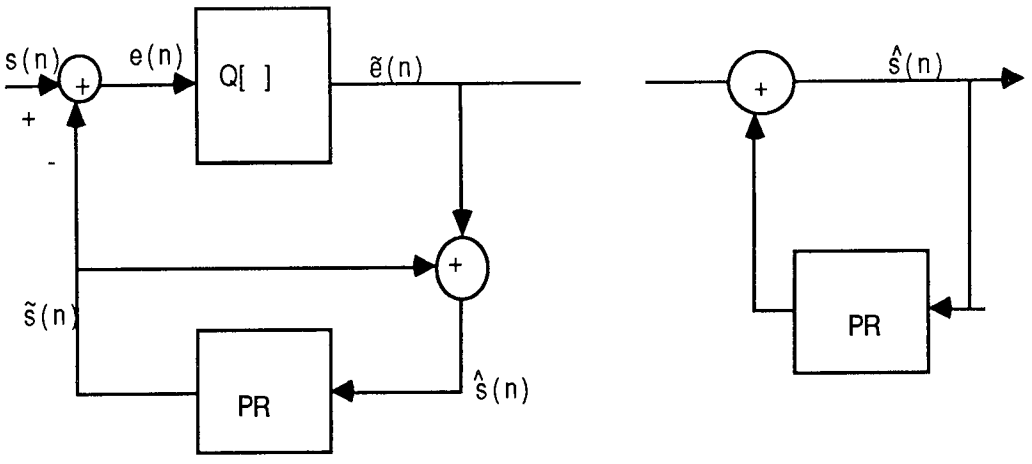
A conventional pulse code modulation system encodes each sample of the input waveform independently from all other samples. A PCM system is thus inherently capable of encoding an arbitrarily random waveform whose maximum frequency component does not exceed one-half the sampling rate. However, as was mentioned in chapter one, analysis of the speech waveform indicates there is considerable redundancy from one sample to the next. This sample-to-sample correlation is generally 0.85 or higher between adjacent 8kHz samples. This redundancy in conventional PCM codes suggests significant savings in transmission bandwidths are possible through more efficient coding techniques.

The other two waveform coding techniques, differential pulse code modulation (DPCM), and its special case, delta modulation (DM), are discussed next in this chapter. They both exploit the sample-to-sample correlation of the speech waveform with the intent of reducing the bit-rate.

### 2.1.2 Differential Pulse Code Modulation (DPCM)

An implication of the sample-to-sample correlation in speech is that sample

values can be "predicted" from quantised versions of their predecessors. Thus, in practice, differential pulse code modulation is implemented by quantising and encoding the difference between a sample value and a reconstructed version of its predecessor. This "feedback" implementation ensures that quantization errors do not accumulate indefinitely. If the feedback signal drifts from the input signal as a result of an accumulation of quantization errors, the next encoding of the difference signal automatically compensates for the drift. In a system without feedback the output produced by a decoder at the other end of the connection might accumulate quantization errors without bound.




Where $\tilde{s}(n) = \sum_{k=1}^{p} a_k \hat{s}(n-k)$, is the prediction of the sample value, s(n),

$\tilde{e}(n)$ is the quantized version of the prediction error, e(n),
and $\hat{s}(n)$ is the reconstructed version of the sample value, s(n).
$a_k$, $k = 1, 2, \ldots, p$, are the predictor coefficients.

Fig.2.5 Differential Pulse Code Modulation

The box labelled PR is known as a "predictor". Its function is discussed in detail in this chapter under "predictive coding". In a simple form of DPCM, PR can be represented by an integrator which, as will be shown later, can be regarded as a first order predictor.

However, the high sample-to-sample correlation implies that a sample value can be predicted more accurately by a prediction method based on several past sample values. A more efficient implementation of DPCM should therefore incorporate prediction based on several reconstructed sample values. This

aspect of predictive coding will be discussed later. Next to be discussed in this section is delta modulation (DM), a special case of differential pulse code modulation.

### 2.1.3 Delta Modulation (DM)

The exploitation of signal correlations in DPCM suggests the further possibility of oversampling a signal to increase the adjacent-sample correlation and, thus, to permit the use of a simple quantization strategy. Delta Modulation, the 1-bit version of DPCM, is precisely such a scheme.

An approximation to the input waveform is constructed in the feedback path by stepping up one quantization level when the difference is positive ("one") and stepping down one quantization step when the difference is negative ("zero"). In this way, the input is encoded as a block of "ups" and "downs" in a manner resembling a staircase. The feedback signal continues to step in one direction until it crosses the input, at which time the feedback step reverses direction until the input is crossed again. Thus, when tracking the input signal, the DM output "bounces" back and forth across the input waveform allowing the input to be accurately reconstructed by a smoothing filter.

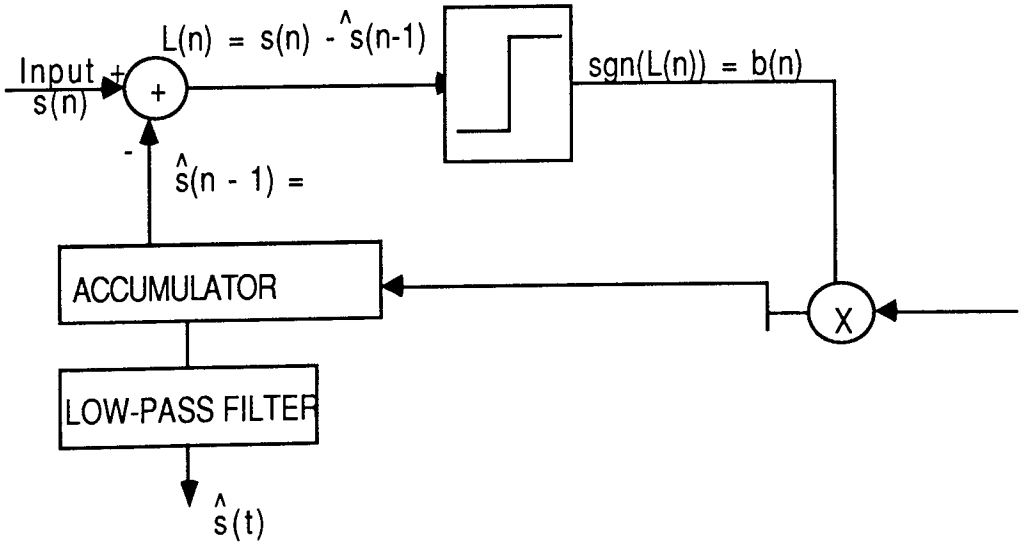



Fig.2.6 Block Diagram of Linear Delta Modulation (LDM)

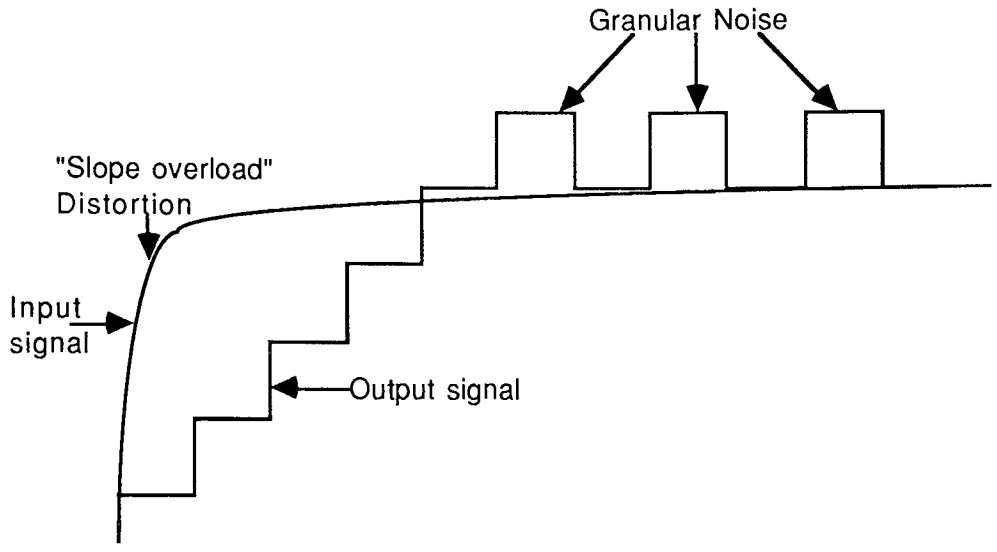



Fig.2.7 Linear Delta Modulation (LDM), showing the two types of quantization error; Granular noise and Overload noise

Figures 2.6 and 2.7 above illustrate the principle of linear delta modulation (LDM). As can be seen from these figures, the stepsize is fixed and reconstruction of the input signal is done by stepping up or down by one stepsize, depending on whether the last difference was positive or negative. Fig.2.7 also illustrates the two types of distortion noise inherent in delta modulation; slope overload and granular noise. These two types of distortion occur respectively when the stepsize, $\delta$, is too small to follow a steep segment of the input waveform and in a situation where the staircase function hunts around a relatively flat segment of the input function with a stepsize that is too large relative to the local slope characteristics of the input. Hence, for given statistics of the input signal slope, relatively small values of $\delta$ accentuate slope overload while relatively large values of $\delta$ increase granularity. It should, therefore, be possible to tailor the time-invariant stepsize, $\delta$, to provide a minimum error power.

### 2.1.4 Adaptive Delta Modulation (ADM)

In Adaptive Delta Modulation (ADM), the stepsize is varied to increase during a steep segment and to decrease when the delta modulator is quantising a slowly varying segment of the input waveform. In this way, the stepsize increases by a predetermined factor when slope overload is noticed and decreases by another predetermined factor when granular noise is noticed. In general, stepsize increases are higher than stepsize decreases.

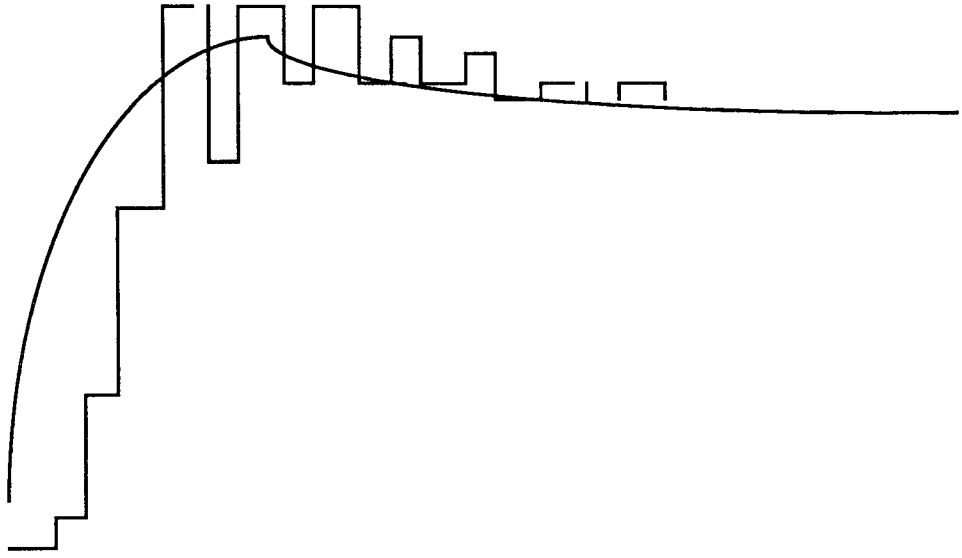


Fig.2.8 Adaptive Delta Modulation

A design problem in ADM is to specify suitable rules for stepsize variation. Studies in this area have produced several step adaptation methods. In general, adaptation is based on the observation of sequences of quantizer outputs . In a typical and conceptually simple realisation described by Jayant [ 7], successive bits b(n) and b(n - 1) are compared to detect probable slope overload, (b(n) equal to b(n - 1)), or probable granularity, (b(n) not equal to b(n - 1)). The adaptation rule is as follows:

$$\delta(n) = \delta(n-1)P, \text{ if } b(n) = b(n-1)$$

$$= \delta(n-1)Q, \text{ if } b(n) \neq b(n-1) \qquad 2.7$$

where $Q = 1/P$ and $P > 1$.

The rate of step size increase or decrease is given by a single factor P. We note from the expression that P = 1 represents linear (non-adaptive) delta

modulation. Typically $1 < P < 2$.

The step size can be adapted either instantaneously (instantaneous adaptation), when the step size is adapted from sample to sample, or syllabically (Syllabic Adaptation), when the step size adaptation is made more smoothly in time, with a time constant that is of the order of 5 to 10ms.

Syllabic Adaptation is characterised by slow adaptation. The effect of this is decreased granular noise in the output speech, at the cost of a significant increase of slope-overload distortion. The result is a "clean sounding" signal at relatively low bit rates but with a loss of "crispiness". On the other hand, the performance of an instantaneous adaptor at low bit rates can be seriously impaired by excessive granular noise. In addition, syllabic adaptation is resistive to bit errors, and therefore preferable in applications with high probability of channel errors.

A drawback of delta modulation is the high sampling rate requirement (far in excess of the Nyquist rate) to ensure the necessary sample-to-sample correlation to compensate for the very coarse quantization (one bit per sample). However, it has the following attractions:

(i) Simplicity; the A/D conversion function is provided by a simple comparator. A positive difference voltage produces a 1, and a negative difference voltage produces a 0.

(ii) Delta modulation allows the use of relatively simple filters for bandlimiting the input and smoothing the output. As mentioned in chapter one, the spectrum produced by a sampling process consists of replicas of the sampled spectrum centered at multiples of the sampling frequency. The relatively high sampling rate of a delta modulator produces a wider separation of these spectra, and hence fold-over (aliasing) distortion is prevented, with less stringent roll-off requirements for the filter.

### 2.1.5 Adaptive Predictive Coding (APC) [8],[9],[13]

Pulse Code Modulation and Delta Modulation are relatively simple to implement but they require considerably more transmission bandwidth than the

analogue signals they encode. Adaptive Predictive Coding is a digitisation technique that encodes voice signals with significantly lower bit rates. It extends the concept of differential pulse code modulation to several past samples in predicting the present input sample, thus making use of the predictability of speech samples as the result of the high sample-to-sample correlation of the speech signal. APC thus performs better than conventional DPCM, which uses only first-order prediction.

Adaptive Predictive Coding also takes advantage of other speech redundancies such as cycle-to-cycle correlations and pitch-interval to pitch-interval correlation. The result is a significant increase in complexity and the amount of signal delay required in the feedback loop (and the decoder).

Referring to Fig.2.5, the predictor, PR, can be represented in z-transform notation as

$$P(z) = \sum_{j=1}^{p} a_j z^{-j} \qquad 2.8a$$

where $a_j$, j = 1, 2, 3,.....,p, are the predictor coefficients, p is the order of the predictor filter and $z^{-1}$ represents a delay of one sample. We note that p = 1 represents a conventional (first-order predictor) DPCM. From equation 2.8a above, the output of the predictor, PR, in Fig.2.5 can be determined as

$$\tilde{S}(n) = \sum_{j=1}^{p} a_j \hat{S}(n-j) \qquad 2.8b$$

Briefly, the system functions as follows , ( refer to Fig. 2.5 for illustration and explanation of the values involved ):

(i) The speech signal is sampled to produce a block of samples S(1), S(2), ....,S(n).

(ii) A section of the speech samples is chosen and the predictor parameters determined for this section.

(iii) The prediction parameters are coded and transmitted.

(iv) The predictor, PR, forms an estimate of a sample value based on past decoded sample values ( as in equation 2.8b ) and the estimate subtracted from the actual sample value, S(n).

(v) The difference signal, e(n), is then quantised, encoded and the codes transmitted.

(vi) At the receiver, e(n) is recovered by decoding the transmitted codes. An estimate, S(n) predetermined in the same way as by the predictor in the transmitter, is then added to the decoded value of e(n) to recover the actual input sample. A slight deviation of the actual input sample value may, of course, still exists due to quantization error as discussed earlier in this chapter.

(vii) Sample values are formed simultaneously at the transmitter and the receiver. These values are used in the prediction process as illustrated in Fig.2.5 and equation (2.8b), to form an estimate of the next input sample in the block and steps (v) and (vi) above repeated until all the samples in the chosen section are transmitted.

(viii) Another section in the block of samples is chosen and the process repeated as described by step (ii) to (vii).

Thus the predictor is periodically updated at time intervals given by the length of the stored section. This is necessary to cope with the non-stationary nature of the speech waveform. Only a predictor that varies with time can predict the speech signal values at all times. The periodic transmission of predictor coefficients does not consume excessive channel capacity as the coefficients tolerate coarse quantization and slow updating.

Periodically updating the predictor coefficients as described above is known as syllabic predictor adaptation. The predictor coefficients are normally updated once every 5-25ms. Syllabic predictive adaptation has the advantage that the predictor coefficients are computed from input samples free from quantization noise. The coefficients can also be error-protected before transmission.

However, this method involves a time delay equal to the period of predictor update, which may be significant when used in a system with echo problems.

Predictor coefficients can also be updated on a sample-by-sample basis. This is known as instantaneous predictor adaptation. The main advantage here is that, since the predictor coefficients are computed from output sample values, there is no need to transmit them as they can easily be derived at the receiver. This method therefore minimises the amount of information transmitted as side information, thus minimising the transmission bit-rate. Its main draw back is system complexity, as the receiver has to be capable of deriving the predictor coefficients based on received information. Also, because the predictor coefficients are computed every sampling period, algorithms based on this method of updating predictor coefficients tend to execute more slowly than those based on the syllabic method.

Another speech redundancy often exploited in conjunction with sample-to-sample correlation discussed above is pitch-interval to pitch-interval correlation, that is, the quasi-periodicity of the speech signal. Based on this redundancy, the present value of a speech signal can be predicted by equating to the value of the speech signal one or more periods earlier. The predictor has to provide gain adjustment so as to account for amplitude variations from one period to another. This type of predictor is represented in z-transform notation as

$$P_d(z) = \beta z^{-M} \qquad 2.9$$

where M is a relatively long delay in the range of 2 to 20ms, corresponding in most cases to a pitch period (or a multiple of pitch periods), and $\beta$ is a scaling factor. The accuracy of prediction depends on the correlation between adjacent pitch periods, the correlation in turn depending on the speech sound and on the speaker.

The prediction gain can be increased by using additional samples on both sides of the pitch delay for prediction of the present sample. This can be represented in z-transform notation as

$$P_d(z) = \beta_1 z^{-M+1} + \beta_2 z^{-M} + \beta_3 z^{-M-1} \quad 2.10$$

The two types of predictive filters, PR, and $P_d$, are often combined in series [13]-[15] to form a composite predictive filter with transfer function,[1 - P(z)],which is the product of the transfer functions of the two filters

Predictive coding can be made even more effective by making use of the theory of auditory masking. This theory states that noise in the frequency regions where speech energy is concentrated (such as formant regions) would be partially or totally masked by the speech signal. The implication here is that a large part of the perceived noise in a coder comes from those frequency regions where the signal level is low. Hence the coder can be made more effective by including a filter whose function is to reduce the noise in the regions of low speech signal level and increase the noise in the formant regions where the noise would be effectively masked [14].

### 2.1.6 Sub-band Coding [4],[5],[10]

This type of waveform coder uses a frequency domain analysis of the input signal as opposed to a time domain analysis as in pulse code modulation and its derivatives discussed earlier.

Basically, the coder functions by first dividing the spectrum of the input signal into separate bands using a bank of band-pass filters as shown in Fig.2.9. The output of each of the relatively narrow sub-bands is individually encoded with separate adaptive PCM (APCM) encoders. The individual bit streams are then multiplexed for transmission to the decoder, where they are demultiplexed, decoded, and combined to reconstruct the inputs.

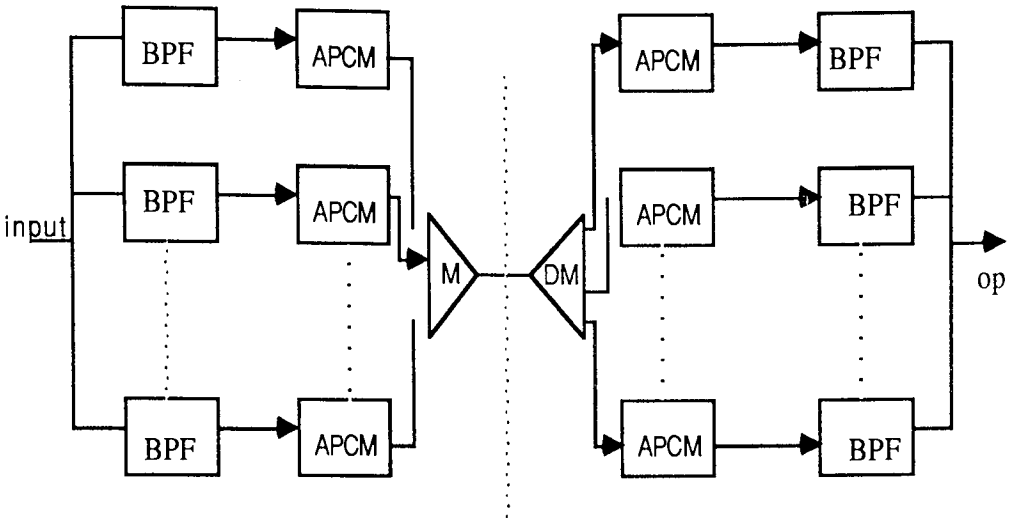



Fig.2.9 Sub-band Coder

A serious drawback of sub-band coders is their system complexity. This can be seen in the number and complexity of the filtering devices needed for its implementation. Also, since the different sub-bands are coded separately, the system treats them as if they were from different inputs, thus requiring a more sophisticated operational algorithm to direct and co-ordinate the coding and decoding operations.

## 2.2 VOCODERS [4],[5],[14]-[22]

For the most part, the encoding/decoding algorithms described previously have been concerned primarily with reproducing the input waveform as accurately as possible, thus assuming little or no knowledge of the nature of the signal they process. They are basically applicable to any signal occurring in a voice channel. Exceptions are sub-band coding and adaptive predictive coding, which are designed for relatively low bit-rates. The coders are therefore closely tailored to the statistics of speech signals and may not provide comparable quality for other signals. To a certain extent, differential pulse code modulation and delta modulation also exhibit certain signal-specific properties.

Vocoders (voice coders) are speech specific. That is, they encode speech signals and speech signals only. Furthermore, they often produce unnatural or

synthetic sounding speech, their basic goal being to encode only the perceptually important aspects of speech with fewer bits than the more general waveform encoders. This makes them more applicable in limited bandwidth applications where the other techniques cannot be used. These application areas include recorded message (e.g. "wrong number"), encrypted voice transmission over analogue telephone circuits, computer output, and educational games.

However, as indicated above, the dividing line between vocoders and some of the waveform coders, such as sub-band coders and adaptive predictive coding used for low bit-rate speech coding, is not very definite. In fact, some of the lowest bit rate voice encoders and decoders use a certain amount of analysis and synthesis to digitally represent speech. In general, the lower the bit-rate required, the closer the coding algorithm approaches that for a vocoder. As the aim of this project is two-fold: the coding of speech at low bit-rate as well as maintaining the speech naturalness, a discussion of vocoder techniques in conjunction with waveform coding is appropriate , and we now proceed to discuss various vocoding techniques.

Vocoders fall into three basic categories: CHANNEL VOCODERS, FORMANT VOCODERS, and LINEAR PREDICTIVE CODING. Like their waveform counterparts, vocoders can further be classed as time-domain and frequency-domain. Each of the above vocoder types is based on the exploitation of one or more speech redundancies and their difference can be seen in the type of speech redundancy each type is based on. We therefore proceed by first further examining the redundancies in speech and then discuss the various vocoder techniques.

Speech is highly redundant and a digital speech coding technique is more efficient, in general, the more successfully it exploits these redundant properties of speech. Two of these speech redundancies, non-uniform amplitude distribution and sample-to-sample correlation, were discussed in section 2.1.1. Log-PCM (companded pulse code modulation) exploits the non-uniform amplitude distribution property to assign small quantization step sizes to the more frequent low amplitudes, while the less frequent high amplitudes undergo

more coarse quntization. The net result is a saving in bit-rate and a more efficient coding system than conventional pulse code modulation. The exploitation of this property of speech is also the basis of adaptive quantization, in which the step size is varied according to variations in the slope of the input signal, as was discussed in adaptive delta modulation. The sample-to- sample correlation property of speech is made use of in predictive coding.

The other speech redundancies are :

(i) Cycle-to-Cycle Correlation; where the speech waveform correlation extends over numerous samples corresponding to several cycles of an oscillation. This speech property is due to the fact that at any particular instant in time, certain sounds may be composed of only a few frequencies within the speech bandwidth.

(ii) Pitch-Interval to Pitch-Interval Correlation: This redundancy is due to the voiced/unvoiced nature of human speech sounds. "Voiced" sounds are caused by vibrations in the vocal cords. The period of vibration of the vocal cords is referred to as the pitch interval or, simply, the rate of excitation is known as the pitch. In general, voiced sounds arise in the generation of vowels and the later portions of some consonants. "Unvoiced" sounds, on the other hand, correspond to certain consonants such as f, j, s, x.

(iii) Inactivity Factor, which is mainly the result of one person listening while the other talks during a telephone conversation. Thus, a conventional (circuit-switched) full-duplex, connection is significantly under utilised. Full utilisation is obtained through digital speech interpolation (DSI), which involves sensing a speech activity, seizing a channel, digitally encoding and transmitting the utterances, and releasing the channel at the completion of each speech segment.

(iv) Non-uniform Long-term Spectral Densities; This speech redundancy is a frequency domain interpretation of the long-term predictability of the speech signal. This is characterised in the frequency domain by a non-uniform spectral density, which is an indication of redundancy in the speech waveform.

Thus, a frequency domain approach to more efficient coding involves flattening the spectrum before encoding the signal. This flattening process can be achieved by passing the signal through a high-pass filter to emphasise the high-frequencies before sampling. The original waveform is recovered by passing the decoded signal through a filter with a complementary low-pass characteristic. Since a high-pass filter exhibits time domain characteristics similar to a differentiator, and a low-pass filter has time domain characteristics similar to an integrator, the spectrum flattening process is essentially equivalent to encoding the slope of the signal at the source and integrating at the destination to recover the signal. This is the basic procedure of sample-to-sample redundancy removal in the time domain.

(v) Short-Term Spectral Densities; Over shorter periods of time the spectral densities of speech vary considerably, exhibiting sound specific structures with energy peaks (resonances) at some frequencies and energy valleys at others. The frequencies at which the resonances occur are called formant frequencies or, simply, formants. Voiced sounds typically contain three to four identifiable formants. Again, this is a frequency domain version of the time-domain pitch-interval to pitch-interval correlation redundancy discussed earlier. The presence of formants indicate the presence of voiced sounds.

Each vocoder technique to be discussed is based on the exploitation of at least one of the above redundancies. They are either time-domain or frequency-domain, depending on whether the coding technique is based on time-domain or frequency- domain versions of the above redundancies. Each category of vocoders has its merits and demerits. Frequency-domain vocoders provide improved coding efficiency by encoding only the most important components of the spectrum on a dynamic basis. Hence, they provide lower bit rates than their time-domain counterparts. However, they typically produce more unnaturally sounding speech. We now proceed to discuss the different vocoder techniques in more detail

### 2.2.1 Channel Vocoders

This vocoder determines the short-term spectrum of a speech signal as a

function of time and uses a bank of bandpass filters to separate the speech energy into sub-bands. The sub-bands are then full-wave rectified and filtered to determine their relative power levels. The power levels are encoded individually and the codes multiplexed for transmission. In addition to measuring the signal spectrum, modern channel vocoders also determine the nature of speech excitation (voice/unvoiced) and the pitch frequency of voiced sounds. The excitation measurements are used to synthesise the speech signal in the decoder by passing an approximately selected source signal through a frequency domain model of the vocal tract transfer function. The voiced excitation is simulated by a pulse generator using a repetition rate equal to the measured pitch period. Unvoiced excitation is simulated by a noise generator. This type of vocoder, also known as the pitch excited vocoder, is illustrated in the block diagram below.

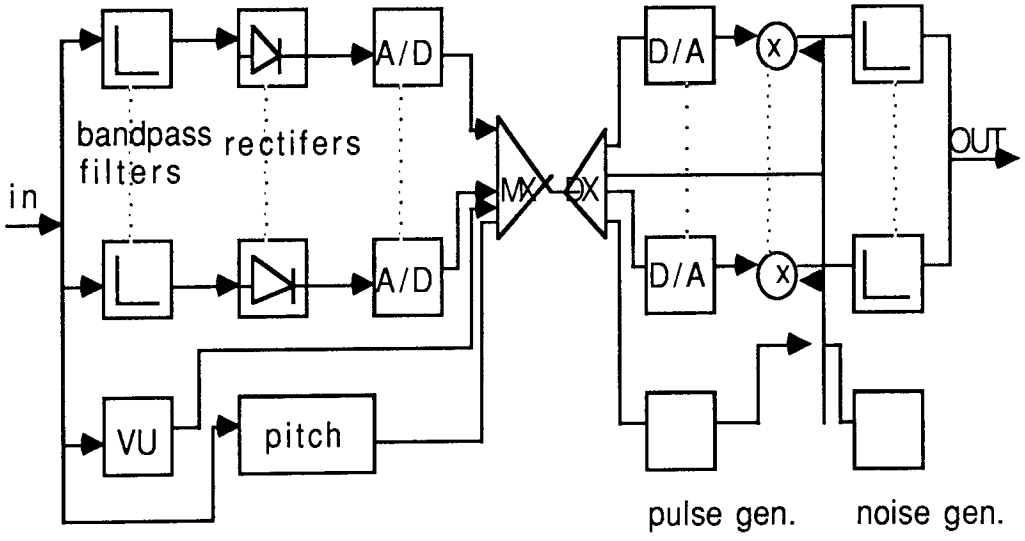



Fig.2.10 Block Diagram of Channel Vocoder

As can be seen from the diagram, the basic structure of a channel vocoder is similar to that of a sub-band coder. However, there are marked operational differences between the two speech coders. In particular, a sub-band coder uses wider band-pass filters, which necessitate sampling the sub-band waveforms more often. Also a sub-band coder encodes waveforms, as has already been discussed. Thus, it includes information that is ignored by a channel vocoder.

### 2.2.2 Formant Vocoder

This vocoder technique, as its name implies, is based on the formant nature of speech. That is, speech energy tends to be concentrated at peaks called formants, instead of being distributed across the entire voice band. A formant vocoder determines the location and amplitude of these spectral peaks and transmits this information instead of the entire spectrum. It also enables changes in formants to be accurately tracked.Thus, by encoding only the most significant short-term components, a formant vocoder achieves lower bit rates than most others of its kind. Typical bit-rates achieved with this vocoding technique are as low as 1kbits/s, compared with 2kbits/s for a channel vocoder.

### 2.2.3 Linear Predictive Coding (LPC) [23]-[25]

Linear Predictive Coding is the only time domain form of the three vocoding techniques discussed in this chapter. It is also the most popular of the three due, no doubt, to its direct application in the time domain. The technique is based on the principle that human speech consists of "voiced" sounds and "unvoiced" sounds. Voiced sounds are produced when air from the lungs is forced through the vocal cords causing them to vibrate. The frequency of vibration of the vocal cords depends on the air pressure in the trachea. This frequency is known as the fundamental frequency and the perceived fundamental frequency is the pitch. The pulses of air emitted from the vocal cords excite the vocal tract giving rise to resonant frequencies in the radiated signal. The shape and size of the vocal tract is varied by the constant movement of the mobile articulators (the lower jaw, tongue, lips, and uvula), thus changing the resonant frequencies. This variation in resonant frequencies gives rise to distinct speech sounds. The vocal cords become loose during unvoiced sounds, allowing air from the lungs to pass into the trachea unaffected.

The coder analyses a speech waveform at the transmitter and the voiced/unvoiced features extracted, plus the pitch period for the voiced sounds, and a gain factor. These are encoded and transmitted. The voiced/unvoiced features are known as the excitation. At the receiver, a synthesizer forms a

mathematical model of the vocal tract and passes the excitation through this model to recreate the speech. The process of speech reproduction by linear predictive coding can be illustrated diagrammatically as in Fig.2.11 below. Voiced sounds are represented by a pulse generator with pulse width determined by the pitch period of the voiced sounds, while unvoiced sounds are represented by a noise generator.

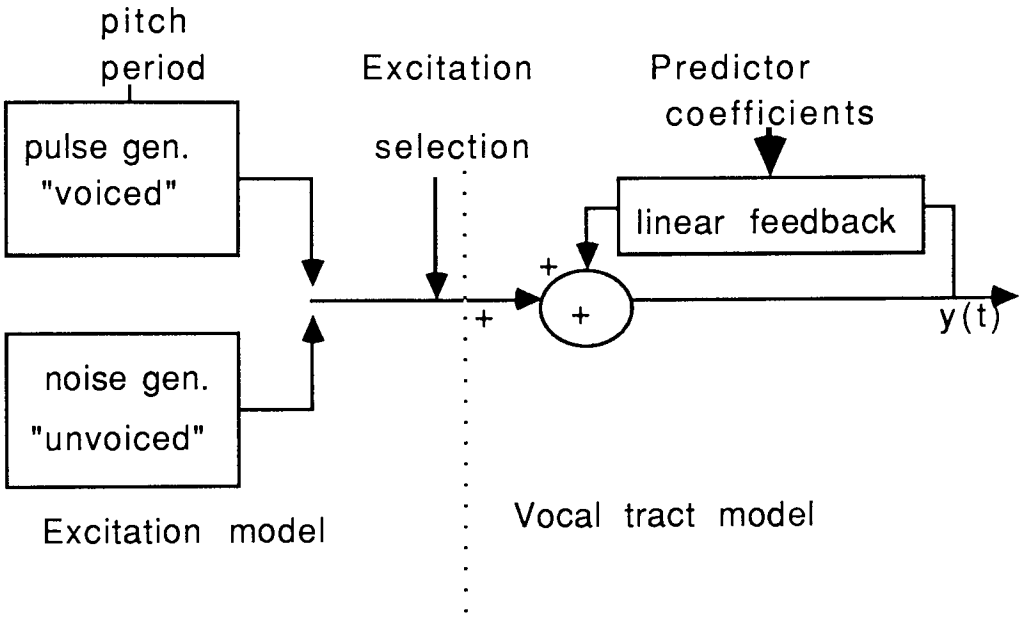



Fig.2.11 Speech generation model of linear predictive coding

The vocal tract model shown in the diagram above is represented mathematically as

$$y(n) = \sum_{k=1}^{p} a_k y(n-k) + G\,x(n) \qquad 2.11$$

where y(n) is the nth output sample, $a_k$ is the kth predictor coefficient, G is gain factor, x(n) is the input at sample time n, and p is the order of the model. The model can be made adaptive by periodically determining a new set of predictor coefficients, corresponding to successive speech segments, at the encoder. In this manner LPC is similar to DPCM or APC coding. However, there is a basic difference in the method of determining the prediction coefficients and the fact that an LPC does not measure and encode difference waveforms or error signals. Instead, the error signals are minimised in a mean

squared sense when the predictor coefficients are determined.

Linear predictive coders provide good estimates of the peaks of the speech spectrum, and are capable of effectively tracking changes in the spectrum envelope. The overall result is that they provide more natural sounding speech than their frequency domain counterparts. Research in LPC is concentrated on encoding speech in the range 1.2 to 2.4kbits/s.

As can be seen, all three vocoding techniques discussed above are closely based on the speech production model also discussed above. Their difference lies in the way the vocal tract section is modelled;

(i) the channel vocoder models it by the short-term amplitude spectrum of the speech signal evaluated at specific frequencies,

(ii) the formant vocoder by the major spectral peaks, and

(iii) the linear predictive coder by the linear predictive coding coefficients as defined above.

In all three, the excitation is modelled as in the linear predictive coding technique described above. This model is poor for the following main reasons:

(i) Speech does not fall neatly into the two categories of voiced and unvoiced;

(ii) In voiced speech the pitch is not constant.

The overall result of this simple model representation is that vocoders in general perform very poorly with high levels of background noise, multiple speakers and non-speech signals.

Of all the three vocoders discussed above, linear predictive coding is the most widely used these days, especially for military applications where the LPC speech quality is acceptable as a means of providing secure low bit rate communications. However, LPC suffers a serious drawback when the input speech is noisy. Errors occur in the voiced/unvoiced decisions, the pitch selected, and the LPC parameters. The resulting synthesised speech is, therefore, severely distorted. To solve this problem, variations of LPC which

are not based on the voiced/unvoiced nature of speech have been devised. These differ from conventional LPC basically in their method of generating the excitation signal. Two of these, multipulse excited LPC (MPLPC) and code excited LPC (CELP) will be discussed here. These are the two LPC coding techniques most widely used these days.

### 2.2.4 Multipulse Excited LPC (MPLPC) [25],[26],[28],[33],[35], [38]

A series of non-uniformly spaced pulses with different amplitudes is used to excite the filter. Here no distinction is made between voiced and unvoiced speech. The same type of excitation waveform is used for all speech segments. This is illustrated by the block diagram of the decoder in Fig.2.12.

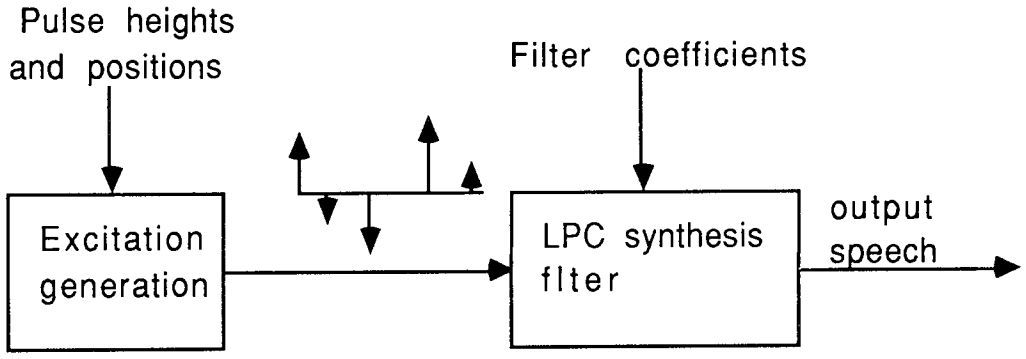



Fig.2.12 Multipulse decoder

For good quality speech, several pulses per pitch period are required and, as all the pulse positions and amplitudes must be transmitted, a trade-off of speech quality versus bit-rate has to be made. Also, it is crucial for the performance of the coder that the appropriate pulse positions and amplitudes be derived at the encoder. The encoder functions as illustrated in Fig.2.13 below.

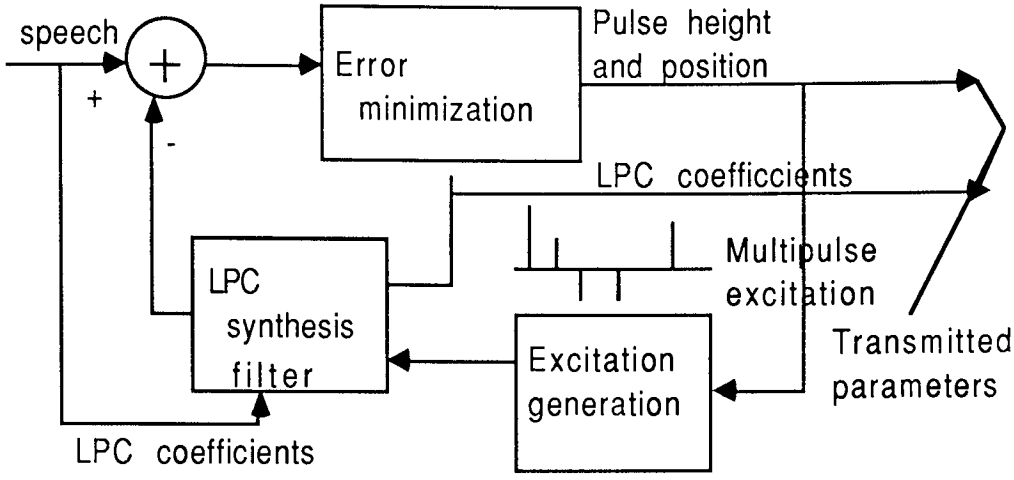



Fig.2.13 Basic Multipulse Encoder

A comparison is made between the input and the synthesised speech and the excitation derived to minimise the error between the two signals. This feedback characteristic of the coder accounts for its better performance, as compared with conventional LPC, with noisy speech inputs. Errors in the feedback LPC loop are compensated for by the excitation selected. The coded output speech is thus a relatively accurate representation of the noisy input.

Again, the coder is made more effective by adaptive techniques. The input speech signal is partitioned into small blocks (32-128 samples), and a search for the pulse position and amplitudes which minimise the error between the input and the synthesised speech is made. The technique can also be modified to include pitch related correlations of voiced speech, as explained earlier for conventional LPC. The coder can operate over a very wide range of bit-rates. Its main drawback lies in the amount of computation involved, which is excessive. A variant of this coder uses uniformly spaced excitation pulses, usually every three to five sample positions. It is thus known as regular pulse excited LPC (RAE).

### 2.2.5 Code Excited LPC (CELP) [39],[40]

This coder, also known as the stochastic or vector excited coder, is another coder based on analysis-by-synthesis techniques. It differs from the multipulse excited LPC coder in the excitation function, the pulses being replaced by an "innovation block". This innovation block consists of M samples of white

Gaussian noise. This is illustrated in Fig.2.14.

The encoder stores many of these sequences in a codebook. A copy of this codebook is stored at the decoder. At the encoder, an "optimum" innovation block is selected by filtering each block in the codebook in turn. The block which results in the minimum mean squared error is chosen. An index number is transmitted to identify the selected innovation block at the decoder. Here again, as in multipulse excited LPC, no distinction is made between voiced and unvoiced speech. The same method of analysis is used to determine the excitation waveform for all speech segments. The innovation block length, M, is typically 32 samples and the LPC predictor analysis is performed typically every 16ms (128 samples), which means four sequences selected for each set of filters. The coder is usually designed to operate between 4 to 6kbit/s, and can be modified to include pitch related speech correlations, as in MPLPC.

Again, as in MPLPC, the main drawback in this coder lies in the excessive amount of computation needed at the encoder in selecting the optimum innovation block. Another drawback is the coder's sensitivity to the accuracy of the quantization of the filter. Its subjective performance degrades rapidly as the quantization error increases.

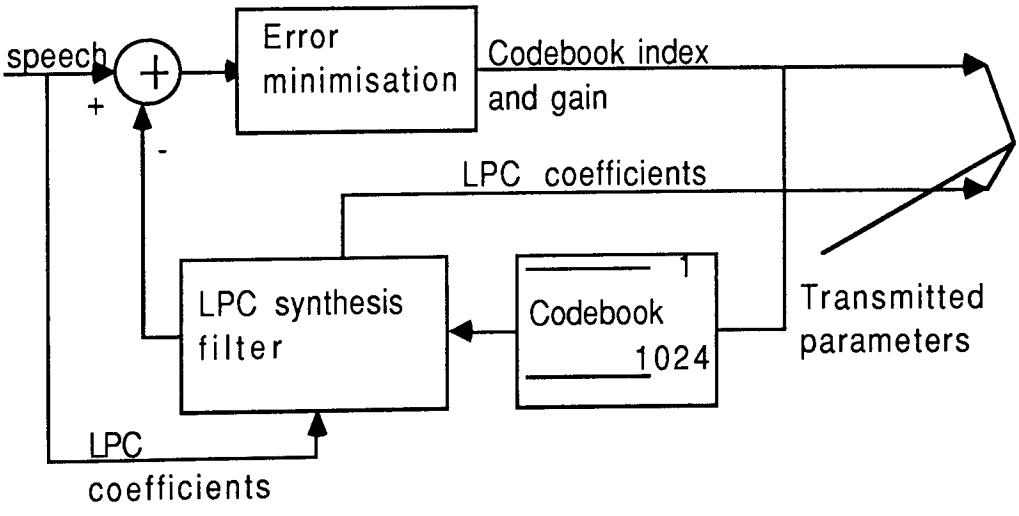



Fig.2.14 Code Excited LPC Encoder

Before leaving this section on speech coding algorithms it is important to discuss another algorithm here that is mostly used for quantising the LPC tract

filter. This technique is known as vector quantization or codebook coding [37].

As its name implies, it operates by storing a codebook containing a set of quantised vectors at both the encoder and the decoder. The quantization process involves mapping an input vector onto a codeword such that the difference between the input vector and the quantised vector is minimised according to some chosen criterion. The encoder then transmits the index number of the selected codeword to the decoder where it is used to select the appropriate quantised vector from the decoder's copy of the codebook. This coding technique achieves bit-rate reduction because fewer bits are needed to transmit the index number than to transmit the vector itself. The technique is illustrated in Fig.2.15.

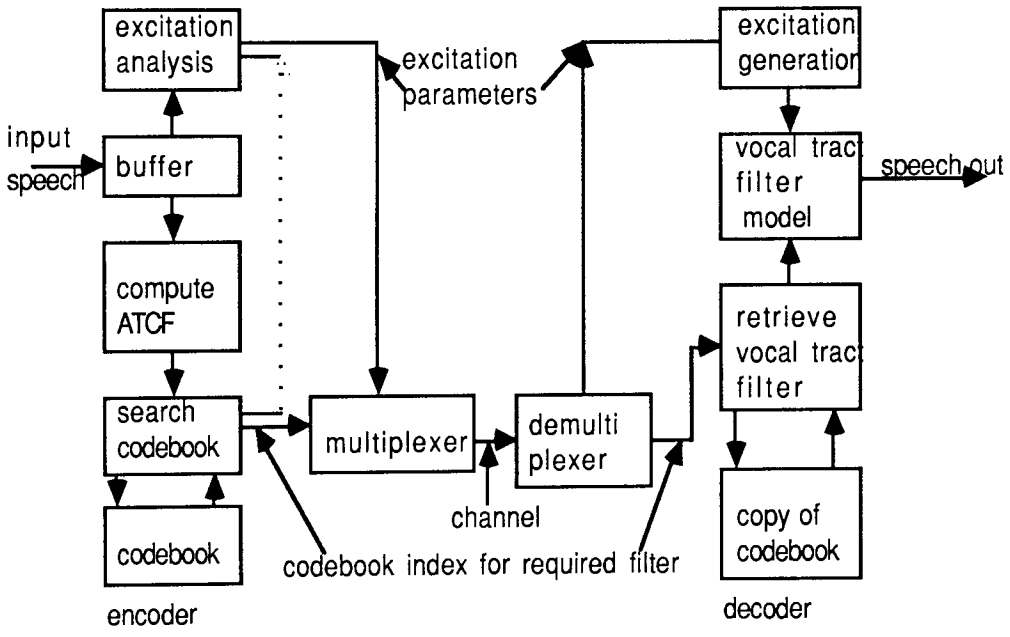



Fig.2.15 Vector Quantization - application to vocal tract model in speech coders.

When used for the quantization of LPC filter parameters, each filter, which is specified by a set of coefficients, is regarded as a vector. The codebook then comprises a set of stored vectors or filters which have been chosen from the possible vocal tract filters. For scaler quantization, LPC analysis and coefficient quantization form two separate stages. However, with vector quantization, the analysis stage (which aims to identify the filter which minimises the residual signal energy) may be by-passed, as shown in Fig.2.15, using the residual energy as the distortion measure for selecting the optimum filter codebook.

In spite of its usefulness in the quantization of LPC filters, resulting in bit-rate reduction, the restriction of the filters which can be used to only those stored in the vector quantizer codebook (usually only 512 or 1024) can result in a noticeable reduction in speech quality. Also, full search of the codebook can be computationally very expensive, as a distortion computation must be carried out for every filter in the codebook. This latter problem can be overcome by tree structure-arranging the codebook [47]-[49]. In a binary tree structure codebook, for example, the codebook is arranged into ten levels with two filters in the first level, four at the second level, and 1024 at the tenth level.

The vector to be quantised is first compared with the two filters at the first level and the branch corresponding to the best matching filter selected. The vector is next compared with the two filters at the second level and the process repeated as for the first level. This is repeated at each of the remaining levels until the final level where the filter selected is the filter used. The result of this tree structure is a dramatic reduction in the number of distortion computations and comparisons needed (only 20, compared with 1024 for full searched codebook.) However, the binary tree structure, as described above, requires more storage for the encoder codebook and the quantization performance is inferior.

An alternative design with improved quantization and reduced storage is known as the graded tree. This requires 30 distortion calculations and comparisons to search the codebook. Its performance is very close to a full search.

## 2.3 ASSESSMENT OF LOW BIT RATE CODECS [4],[5]

Assessment of waveform codecs such as the A-law PCM, is quite straightforward. Two methods are used in the assessment of codec performance:

(i) An objective method, which involves comparing the output speech power with the embedded noise power. The ratio of the speech power to the noise power, known as the signal-to-noise power ratio (SNR), is measured, normally

in decibels (dB), to assess the performance of the codec. In theory, a codec performance is judged better the higher the signal-to-noise power ratio. It can also be used as a design criteria, where a codec is designed to achieve an optimum signal-to-noise power ratio. This is often the criteria used in designing adaptive differential pulse code modulators. As has already been discussed in earlier sections of this chapter, this involves varying the step-size in accordance with variations in the input signal gradient. The maximum step-size is determined to satisfy a given optimum signal-to-noise power ratio

(ii) The subjective method, in which possible users listen to the codec performance and give their subjective views. This generally involves a grading system by which a listener states his perception of impairments in the codec by assigning grades to the individual impairments. The mean score for a group of listeners is taken as the codec performance.

Unfortunately, assessment of low bit-rate speech codecs is not so straightforward for the following reasons:

(i) The diverse nature of the distortions produced by low bit-rate codecs; varying from spurious tones to weird squawks and gargling sounds, which may be correlated with the speech signal or they may be random. Also, the type of distortion produced depends on the low bit-rate codec producing the distortion. Moreover, different peoples' assessment of these diverse degradations can vary significantly. While some people may find background noise or other non-speech sounds generated in the codecs more disturbing than distortion of the speech, others are prepared to accept noise if the speech is clear.

(ii) Performance of low bit-rate codecs can be strongly dependent on their inputs. In general, some low bit-rate speech codecs perform better on certain voices than others. Also, there can be interactions between the effects of the different impairments introduced by low bit-rate speech codecs and other networks.

(iii) The lack of objective test methods for low bit-rate speech codecs. This, as

was mentioned earlier, is the most straightforward method for assessing the performance of a codec. As yet, no such tests have been devised which will reliably give the same results over a range of different coding methods as subjective assessments by potential users.

(iv) Many of the degradations introduced by low bit-rate speech codecs have not been tested before and their effects on the other systems in the network are unknown. The only reliable method of evaluation to date are conversation tests, where the effects of delay, echo, etc. and their interactions can be taken into account.

A speech codec should be tested to evaluate the effect of all factors which are expected to affect its performance. The following parameters should be considered in testing low bit-rate speech codecs:

(i) Range of input levels; a low bit-rate speech codec may perform quite well in a narrow range of input levels centred on an ideal level, but deteriorate rapidly when input levels depart from this range.

(ii) Different speakers; low bit-rate speech codec algorithms often affect male and female voices differently. Hence it is essential to test a codec for voice dependency.

(iii) Transmission errors; being robust to errors is an essential requirement of any speech codec. A low bit-rate speech codec should be tested for expected error distributions, depending on its application, for example, random errors are typical in the Public Switched Telephone Network (PSTN), and bursty errors in radio applications.

(iv) Transcoding; it is important to establish the effects of tandeming systems that utilise encoding at different bit rates. Low bit-rate speech codecs should be tested considering the most probable combinations.

(v) Environmental noise (sending); it is important that the effect of the environment be taken into account in testing a low bit-rate speech codec for the following reasons: (a) Noise added to the input signal can have a more adverse

effect on certain low bit-rate speech codecs than on conventional waveform coding systems. (b) Low bit-rate speech codecs may well be used in noisy environments such as moving vehicles or aircraft.

(vi) Multiple speakers; it is important to know the robustness of the codec to multiple-voice input signals, and to find out whether any adverse effects occur such as "break-up" of transmission, spurious signals, etc. For example, if the codec is to be used in a conference mode, where more than one speaker can speak at the same time, then it must be ensured that the encoding algorithm can deal with multiple speakers where the difference in speech levels could be zero.

## 2.4 LOW BIT-RATE SPEECH CODING: Current trends [4],[32],[51],[52]

Foremost among the factors considered in choosing a particular codec or digitisation algorithm are[4]:

(i) speech quality

(ii) transmission rate

(iii) tolerance to transmission errors

(iv) coding format and data patterns

(v) signal processing requirements

(vi) timing requirements

(vii) implementation cost.

The relative importance of these factors depends on the application: all applications have certain minimum quality requirements. Beyond these requirements, the most important characteristics of a transmission codec are its cost, bit rate, and its performance in the presence of channel errors. On the other hand, an encoder for a digital switch is concerned mostly with implementation cost, while a codec for digital voice storage is mostly concerned with storage requirements (bit rate) but allows considerable flexibility in data format.

Speech quality assessment for a given codec was discussed in the preceding section. In general, characterisation of speech quality involves two

considerations, listener acceptability and intelligibility. Listener acceptability factors include naturalness, speaker recognition, and perception of noise or distortion. Two methods of evaluating acceptability, the subjective method and the objective measurement method, were discussed in the preceding section.

The subjective method involves a group of listeners rating various speech sections in terms of overall subjective quality. The acceptability of a codec is determined as the percentage of listeners rating the quality of the speech segments as adequate. Generally, a more consistent procedure in which listeners are required to indicate a preference between outputs of two different codecs or between the output of a codec and unprocessed speech. In general, only higher-rate waveform codecs such as PCM, DPCM, or DM, provide high acceptability scores. Objective measurement methods involve determining the signal-to-noise ratio of the codec output. As was mentioned in the preceding section, this method is meaningless with respect to low bit-rate codecs. Such codecs are based on digitisation algorithms that preserve only the perceptually significant factors of speech. They make no attempt to preserve source waveforms. Thus an SQR (signal-to-quantizer noise ratio) measurement has limited relevance when comparing codecs with different noise characteristics.

Objective determination of speech quality has another problem, even with waveform codecs. This involves selecting an appropriate input. Sine waves are often used because of their convenience. However, a sine wave differs from a typical speech waveform in several respects:

(a) Speech typically contains several strong frequency components (formants) at any one time, but some encoding algorithms encode different frequencies with differing quality.

(b) A speech waveform has a higher peak-to-peak average ratio than a sine wave. Thus, overloading or clipping is more likely to occur on a speech waveform.

(c) Speech activity is intermittent. Hence, the transient response (adaptation

speed) must be analysed in addition to steady state performance.

Intelligibility tests require listeners to recognise specially designed utterances consisting of isolated syllables and words or whole phrases and sentences. Unlike acceptability tests, they do not ask for preferential evaluations on the part of the listeners. This makes the tests subjective to the nature of the test material (word or sound familiarity, accents, etc.) and listener capabilities. They are used mostly for low bit-rate vocoders where intelligibility may be the only achievable and necessary criterion.

Speech quality can be broadly classified into three categories, namely, toll, communications, and synthetic quality. Toll quality speech is equivalent to the speech quality of a log-PCM codec, with transmission bit-rate of about 64kbit/s. It is the accepted quality for commercial telephone speech. Communications quality refers to speech quality with noticeable degradations but good intelligibility and at least some naturalness. Synthetic quality refers to speech quality with intelligibility only. This quality is characteristic of vocoder outputs.

Research in low bit-rate speech coding has been directed, in recent years, towards achieving two goals:

(i) The design of speech coders that will produce good quality speech at low bit-rates, typically less than 8kbits/s, and

(ii) the implementation of such coders at a realistic complexity, using current digital signal processing (DSP) technology.

For reasons mentioned in an earlier section in this chapter, the most favoured and the most widely used vocoding technique has been linear predictive coding (LPC). Considerable effort has been directed towards improving the performance of algorithms based on this technique. These include multipulse excited linear predictive coding (MPLP),code excited linear predictive coding (CELP), and regular excited linear predictive coding (RELP). These coding techniques, known in general as analysis-by-synthesis coding

techniques because of their linear predictive speech production model basis, differ principally in the way that the excitation signal, used to drive the vocal tract filter, is derived. Each has its own advantages and drawbacks, and current studies have been directed towards overcoming those drawbacks in a particular algorithm so as to fully and effectively exploit its advantages.

Multipulse excited linear predictive coding was discussed earlier in this chapter. Its main advantage over conventional LPC lies in its flexibility for both voiced and unvoiced sounds. The two speech sounds are represented by the same pulses, with no distinction made between voiced and unvoiced sounds, as in conventional LPC. The result of this is that multipulse provides a compressed representation of the ideal excitation signal.

However, it suffers a drawback that the transmitted bit-rate depends on the number of excitation pulses transmitted. This means that to reduce the data rate requires that the number of the excitation pulses transmitted be reduced. This in turn calls for better modelling of the LPC spectral filter, and/or the use of more efficient coding methods. The net result of all these is, of course, a more complex system. This has been the main reason for the poor performance of the technique for data rates below 10kbits/s[8]. Direct application of vector quantization as a more efficient coding method has been tried.

As was mentioned earlier in this chapter, vector quantization achieves low bit-rate transmission by coding and transmitting only the index of the codebook value of the input speech segment. In this way, the LPC tract filter for a particular speech segment can be identified at the receiver by the decoded transmitted index value for that speech segment. Its main drawback has been the limited size of its codebook, which is stored at both the transmitter and the receiver. Also, the codebook search involves excessive computations. Research effort has been directed both towards minimising the computation and extending the codebook size.

The codebook size problem can be overcome by the use of a tree codebook search. This, as was mentioned earlier in this chapter, involves a trade-off of search speed for suboptimal performance. Several methods of designing the

tree code, aimed at optimising vector quantization performance and thus achieves the low bit-rate target at a minimal complexity, have been proposed.

Regular pulse excited LPC, (RELP) is a multipulse-excited LPC with the excitation pulses occurring at regular time intervals. Here again, vector quantization has been introduced as the solution to the problem posed by pulse search. In addition, new methods for excitation search are being looked into. Among these is the one in which the pulse amplitudes are computed one at a time, rather than simultaneously in a block as in the case of a conventional RELP coder. The immediate result of such sequential computation is a reduction in coder computational complexity.

As was mentioned earlier, code excited LPC is quite different from the other analysis-by-synthesis coding techniques in that it is a code excited, rather than a pulse excited, technique. It forms a codebook of the excitation signal in the same way as vector quantization forms a codebook of possible input signals. It is mainly for this reason that it has been the most widely used of the three. However, the complexity of its algorithm has so far made real-time implementation of the technique prohibitive. Considerable attention has been directed towards this complexity issue in recent years. A number of simplifications to the original algorithm have been proposed.

Also known as vector excited coding, CELP is often combined with vector quantization as in the case of the other two analysis-by-synthesis coding techniques, to achieve the low bit-rate requirement. Among the several methods proposed for the simplification of the coding algorithm is the dynamic bit allocation technique (DBA) intended to improve the performance of a CELP coder for a given bit-rate.

Based on the observation that the minimum bit-rate needed to adequately code both the long-term and the short-term predictors in a CELP system varies dynamically with time, the method reallocates bits saved from frames, where fewer bits suffice for the predictor, to the excitation vectors. In this way, the overall coder performance can be improved without increasing bit-rate, as the total number of bits in each frame is kept fixed.

In addition to the dynamic bit allocation technique, the suggested method also includes variable rate encoding of long-term and short-term predictors. This is intended to reduce the total bit-rate without degrading the speech quality. It is based on the fact that the long-term prediction is of little use for unvoiced speech frames and for some transitional regions, while the short-term predictor parameters which carry formant information of the speech signal usually change slowly with time. This variation can sometimes be so small that a simple repeat of the previous frame's parameters would be adequate for characterising the current spectrum. Thus, it is unnecessary to perform both predictors (short-term and long-term) for every frame, the quantization of which typically takes 40 per cent of the total bit-rate in a conventional CELP coder. In this method, speech parameters are updated only when perceivable changes between the current and the previous frame occur. Otherwise, the receiver will simply repeat the same parameters used previously for the current frame.

Obviously, this requires a separate algorithm stored at both the transmitter and the receiver to make the decisions of when to send update parameters to the receiver and what decision to make at that end whether update parameters are given or not. For this, the proposed system uses Magills' likelihood ratio called the delta coding algorithm. This is the ratio of two prediction residual energies obtained using the predictor from the previous frame and the optimal predictor designed for the current frame respectively to predict the speech waveform of the current frame. The smaller the likelihood ratio, the closer the two spectra. The LPC parameters of the current frame are not transmitted if the ratio is less than a given threshold.

Another current trend towards achieving low bit-rate speech with good quality while minimising coder complexity has been the use of hybrid coders. These, as the name implies, combine waveform coder techniques with those of vocoders. As was discussed earlier in this chapter, waveform coders are concerned with processing the waveform of the input signal at the transmitter and then reproducing it at the receiver. Thus, they can reproduce such subjective qualities of the speech signal as accent, speaker recognisability, all

with less complex algorithms and equipment. However, their main drawback lies in the increased transmission bandwidth as the result of digitisation and high bit-rate. On the other hand, vocoders are low bit-rate coders and are therefore transmission bandwidth economic. They are handicapped, however, in that they produce synthetic quality speech without the naturalness necessary for toll and communications purposes. By combining certain waveform coding techniques with those of vocoders, hybrid coders can produce speech which benefits in quality and bit-rate from both techniques.

A recent example of a hybrid coder is the one described by V. Savvides and C. S. Xydeas[32]. They recommended a coding system in which the preferential coding capabilities of sub-band coding are combined with the efficient waveform modelling properties of analysis-by-synthesis linear predictive coding.

Frequency domain coders have the ability for distributing encoding distortion across the spectrum of the input signal. In particular, sub-band coders have a structure that is well suited to the application of "preferential" coding to certain spectral regions of the input signal. The input signal spectrum is split into bands and each band encoded independently, thus allowing the system to vary its encoding accuracy across the short-term spectrum of the signal and thus provide the means of controlling the shape of the short-term spectrum of the encoding distortion present in the recovered speech signal. This is particularly important for speech signals, where the concentration of speech energy is primarily in the low frequency formant regions.

However, frequency domain coders, in general, perform very poorly as the bit-rate is reduced below 9.6kbit/s. This is due to the difficulties which conventional waveform coders encounter when coding the already decorrelated sub-band signals. Thus, although this is an efficient mechanism for distributing distortions across the speech spectrum in a subjectively meaningful way, the encoding noise generated by the waveform coders is so excessive that it renders the noise shaping mechanism ineffective (refer to Fig.2.9 and section 2.1.6 for description of sub-band coding). Furthermore, as the bit rate is reduced to

below 8kbit/s, frequency domain coders do not encode certain low energy spectral regions. This results in a severe "muffling" type of distortion in the recovered speech.

On the other hand, conventional linear predictive coding results in considerable deterioration in low bit -rate speech quality (refer to section 2.2.3 for description of linear predictive coding). On the advantage side, linear predictive coding efficiently models the waveform in terms of the excitations and a filter representing the vocal tract, thus overcoming the problem encountered by conventional waveform coders.

The recommended system uses a band of quadrature mirror filters to split the input signal into eight equal bandwidth signals, as in a conventional sub-band coder. The coding system encodes the sub-band signals by first making a decision as to which of the sub-band signals are important and which are not. The former sub-band signals are classed as active while the latter are classed as passive signals. Multipulse linear predictive coding is used to encode each sub-band signal independently. However, only the corresponding active channels transmit the multipulse excitation and the residual gain values as well as the LPC filter parameters, while the passive channels transmit only the filter parameters and the residual gain. At the receiver, the active sub-band signals are recovered from decoded values of the transmitted multipulse excitation sequences and the LPC filter parameters. The remaining passive sub-bands are reconstructed at the receiver with less accuracy by regenerating appropriate excitation sequences from those of the active sub-bands and by processing the regenerated sequences through the corresponding LPC filters. Vector quantization is used to encode the filter parameters with a separate codebook assigned to each channel. The system is said to achieve low bit-rate transmission by encoding and transmitting only the excitation sequences of the active sub-bands, while only the filter parameters of the passive sub-bands are transmitted. At the same time the complete information conveyed by the transmitted speech signal is recovered at the receiver by regenerating the passive sub-bands using the reconstructed active sub-bands and the filter parameters of

the passive channels.

Another example of the use of hybrid coding in low bit-rate speech coding is the one by B. S. Atal [13]. He combined adaptive predictive coding with delta modulation. The predictor parameters, comprising one delay and nine "other coefficients related to the signal spectrum", were readjusted every 5 milliseconds. The speech signal was sampled at a rate of 6.67 kHz, and the difference signal was quantised by a two-level quantizer (delta modulation) with variable step size. However, the results in this paper were based on "preliminary studies" and claim that the binary difference signal and the predictor parameters together can be transmitted at approximately 10 kilobits/seconds. In particular, no attempt was made to quantise the predictor parameters. This means that the effect of transmitted parameters on the transmission bit-rate and speech quality could not have been studied.

## 2.5 THE TREND IN THIS PROJECT

The hybrid trend described in the above examples will be followed in this project. System economy and good speech quality will be the final design parameters. In the light of this, it is hoped to make both the hardware and the software as simple as possible, while at the same time ensuring good quality speech.

# CHAPTER THREE

# METHOD

## 3.1 BACKGROUND

The main algorithm used in this project combines the efficiency of predictive coding as a low bit-rate speech coding technique with the simplicity of delta modulation. These two coding techniques were discussed in detail in chapter two.

The efficiency of predictive coding as a low bit-rate coding technique lies in the fact that it is based on the linear predictive model of speech production. Linear predictive coding itself has gained popularity over its vocoder counterparts over the years and has undergone more improvement, resulting in more variations in its implementation than the others. Two factors have contributed to this:

(i) Linear predictive coding can provide extremely accurate estimates of the speech parameters, and

(ii) computationwise, it is faster than the other vocoding techniques.

In predictive coding, the linear predictive model of the speech production machine, the vocal tract, forms an estimate of a speech signal based on several past reconstructed speech samples. This estimate value is subtracted from the actual speech sample value and the difference value quantised and encoded for transmission. Thus, only speech sample values which cannot be predicted beforehand are quantised and transmitted. Because of the short-term predictability of the speech signal, which forms the basis of the linear predictive model of speech reproduction, the difference can be very small over a short period. Thus, the difference value can be coded with fewer bits (binary digits) than would be possible with the actual sample value.

Delta modulation is the simplest waveform coding technique. The signal

value to be coded is assigned one of two binary values, 1 or 0, depending on whether it is greater than or less than a reference signal. Thus quantization and encoding are performed simply by comparing the signal value with the reference value and assigning and outputting a 1 or a 0, as explained in chapter two. In practice, a past reconstructed sample value provides the reference signal with which a present sample value is compared. This is illustrated below in Fig.3.1, where a simple comparator circuit performs the quantising and encoding and an integrator circuit performs the reconstruction process.

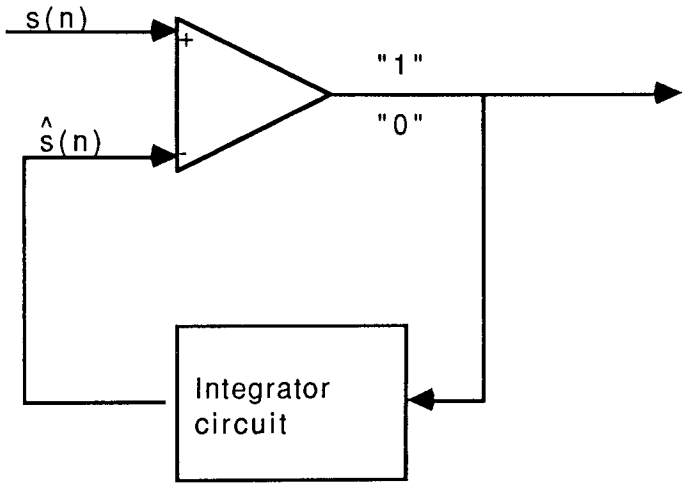



Fig.3.1 Illustrating the main functions of Delta Modulation

It can be seen from the above that delta modulation is a primitive form of predictive coding with the prediction based on the preceding sample value. Its main drawback is the high sampling rate required to enhance the necessary sample-to-sample correlation, resulting in a high transmission bit-rate as compared to the other waveform coding techniques. Thus, the dilemma in delta modulation when used to code speech is the trade-off between bit-rate economy and efficiency.

This dilemma can be overcome by replacing the integrator circuit in the above diagram with a predictor circuit. In this way, the efficiency of the coding system will no longer be determined by the sampling rate but by the prediction circuit; the more efficient the prediction is, the more efficient the overall coding system will be. The prediction, as was discussed in chapter two, can be expressed mathematically in z-transform notation as

$$P(z) = \sum_{k=1}^{p} a_k z^{-k} \qquad 3.1$$

where $z^{-1}$ represents a delay of one sample, $a_k$, k = 1, 2, .....,p are the filter coefficients. The prediction error, e(n), is defined as

$$e(n) = S(n) - \tilde{S}(n) = S(n) - \sum_{k=1}^{p} a_k \hat{S}(n-k) \qquad 3.2$$

where $\hat{S}(n-k)$ is the reconstructed version of the input sample at the (n - k)th sampling time.

The required predictor coefficients are the set of coefficients that minimise the mean-squared prediction error over a short segment of the speech waveform. These are determined by solving a set of linear equations using one of the following two techniques:

(i) The covariance method, and

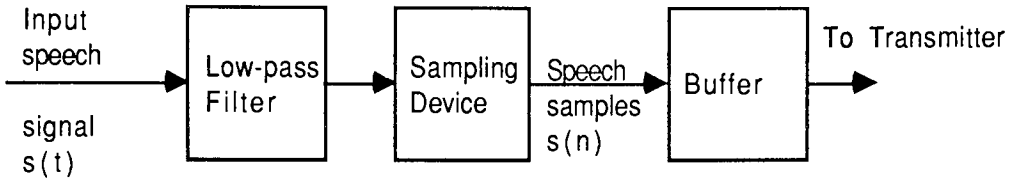
(ii) the autocorrelation method.

The autocorrelation method was used in this project. This is described in chapter four. The predictor coefficients are extracted from short segments of the speech waveform. This is necessary as speech is nonstationary. The result is adaptive predictive coding, in which the predictor parameters are updated periodically.

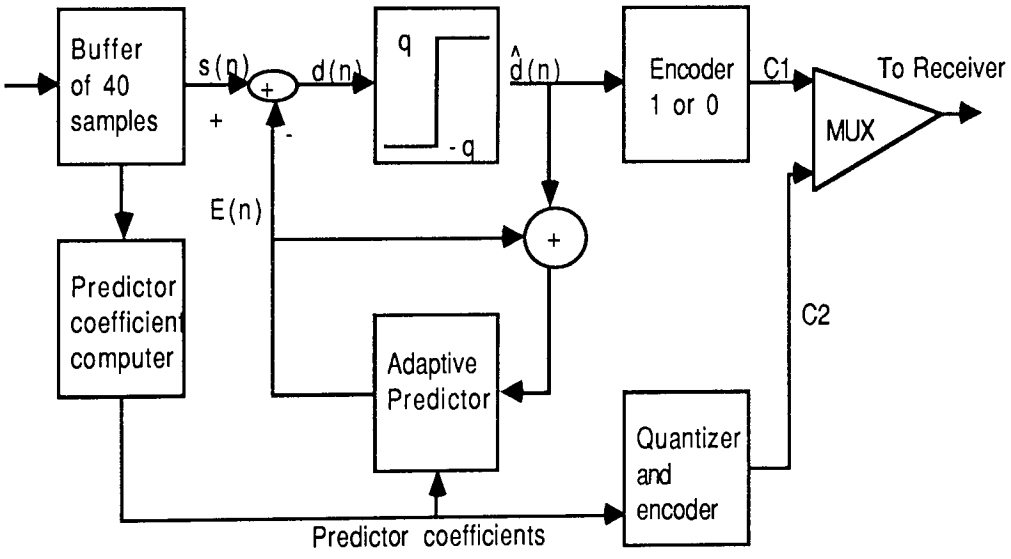
## 3.2 SYSTEM DESCRIPTION

The system used in this project, like all communications systems, consists of three basic sections; the input, the transmission, and the output section. The transmission section includes the transmitter, where the signal is encoded and

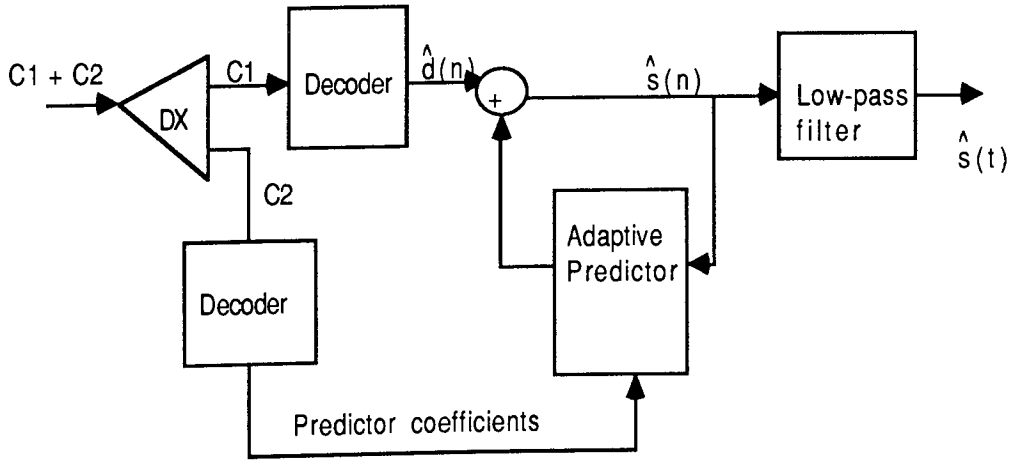
prepared for transmission, the channel, through which the codes are transmitted and which may alter their pattern, and the receiver, where the received codes are decoded and sent to the output section to recover the input signal. Thus, there are three basic processes corresponding to the three sections of the system. The entire system and the processes are illustrated below in Fig.3.2. The various parts of the system function as follows:




(a) Input




(b) Transmitter



(c) Receiver and output

Fig.3.2 Block Diagram of System functional parts

## INPUT

The basic functional blocks of this part of the transmitter are given in Fig.3.2(a). The input speech signal is low-pass filtered, sampled at about 8kHz, and the samples buffered to be transmitted sample-by-sample.

In this project the input device was a data acquisition system designed earlier as part of an undergraduate project in speech recognition. The system was interfaced with an IBM compatible personal computer through the computer's RS input port. The speech signal was generated by speaking into a telephone handset connected to the device. By its design, the data acquisition device could accept signals from the handset for only one word a time. Thus only one could be input at a time. The data acquisition device then coded the signal thus generated by pulse code modulation means and stored the codes in the system's memory.

A program called "capture", written in turbo pascal, read the codes from the temporary memory and stored them in a file on the computer's hard disk.

## TRANSMISSION

### Encoder

The first stage in the transmission system is the encoder. This, as its name suggests, encodes the buffered speech samples in a way suitable for transmission. In this project this was done in two stages. First an input speech sample was estimated at the predictor using past reconstructed speech samples, and this estimate was subtracted from the actual input sample. The difference was then passed through a delta modulator which output a binary 1 if the difference was positive or a binary 0 if it was negative. This code was then transmitted, from which the input speech sample was reconstructed at the receiver. At the same time, the same reconstruction process was done at the

output of the transmitter as can be seen from Fig.3.2 (b). This was achieved by adding the estimated value to the delta modulator output. The reconstructed value was then used, in conjunction with three other past values, to make an estimate of the next input sample value.

As illustrated in Fig.3.2(b), it is necessary to first read the speech signal from the file on the computer's disk into memory before proceeding to encode the signal. Two methods were used in this project. In one method, the speech word was read from the file in blocks of sequential samples and each block stored in a memory buffer. The predictor coefficients were then extracted from the block of samples using the method discussed in chapter four. The coefficients were next used at the predictor to form an estimate, E(n), of an input sample, S(n), in the block, starting with the first to the last sample. The next block of samples was then read into the buffer and processed as with the first, until the whole file was thus processed and transmitted. In this method it was necessary to transmit the predictor coefficients computed for each block of samples to the receiver to enable the transmitted samples to be reconstructed.

The second method was more straight-forward than the first. In this method, the word file was read as a whole and stored in memory.The word was then processed sample by sample as in the first method, with the exception that the predictor coefficients were computed at the output end of the transmitter, based on the code output from the delta modulator. Also the coefficients were computed on a sample-by-sample basis. This method did not require the transmission of the coefficients as they could easily be computed at the receiver based on the received code. This was an advantage since a minimum bit-rate of 8kbps could be achieved as no side information was involved in the transmission. These two methods are discussed in full, including their merits and demerits, in chapters four and five.

Just as it was necessary to adapt the predictor coefficients to enable the predictor to cope with the changing nature with time of the speech waveform, it was necessary to adapt the step size to enable the delta modulator to cope with variations in the signal's dynamic range. The adaptation was done on a

sample-by-sample basis, based on previous output codes (refer to chapter four for discussion of the algorithms used). Thus the step size could be computed at the receiver and it was not necessary to transmit it. This approach was to minimise the amount of side information to be transmitted and thus help attain the minimum bit-rate requirement of this project.

**Decoder**

The function of this stage in the transmission system is to reconstruct the transmitted sample value based on received information.

In this project, this information was in the form of the predictor coefficients and the quantizer step size, where necessary, and the delta modulated error , all in code form. Where the predictor coefficients were computed on a block basis, as in the first method described above, these were quantised, coded, and the codes transmitted with the error information of the first sample value in the block. These codes were received and the predictor coefficients decoded and stored to be discarded at the end of the block. The decoded predictor coefficients were used to form an estimate of a transmitted input sample in the same way as was done at the transmitter. The quantizer level was then reconstructed and, based on the received error code, was added or subtracted from the the estimate to reconstruct the transmitted sample value. In the second method the predictor coefficients were computed based on the received error information as was done at the transmitter. In both methods, the quantizer step size was derived at the receiver.

The reconstructed speech samples were buffered to form the complete word and the word stored in a file on the computer's hard disk.

**OUTPUT**

The output system does the opposite of the input system; converts the speech signal into sound.

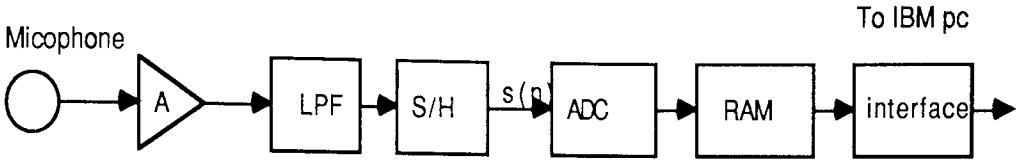
In this project, the output system consisted of a digital-to-analogue converter i.c. chip, a low-pass filter, a power amplifier , plus the necessary glue device, all built on the board. The output was input to a loud speaker on the same board. The system was interfaced to the computer through an 8255

interface board purchased for the project.

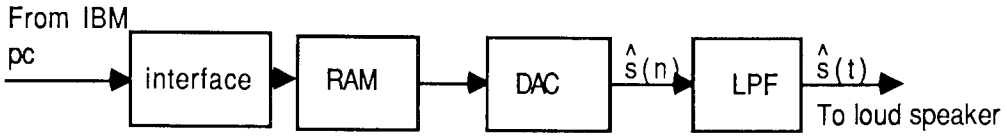
A program sequentially read speech samples from the reconstructed speech file and sent each sample code through the 8255 input-output board to the digital-to-analogue converter on the external board. Each sample code thus output was converted to its corresponding level and this, when low-pass filtered, converted to an analogue level. An analogue waveform was thus formed from which a speech sound was generated by the loud speaker. This was listened to and compared with its input version for grading. This is further discussed in detail in chapter five.

## 3.3 SYSTEM IMPLEMENTATION

Basically, the system is made up of two functional units; hardware and software. The hardware unit captures the signal from an analogue source, in this case microphone, converts the signal into digital data ready to be processed digitally and then converts the processed data back to analogue form to be output to an analogue destination, in this case a loud speaker. These two units were implemented by the data acquisition device and the external output circuit described above, respectively. The software unit consists of low-level port control routines and a high-level data analysis program, both stored in an IBM compatible personal computer. In this project, both the input-output port control and the analysis programs were written in turbo pascal, a high level language. This was mainly for the sake of continuity in operation and also because of the fact that high level languages are easier and safer to program in than low level languages. The two functional units are linked through appropriate interface units to be discussed later in this chapter. The functional blocks are illustrated in the figure below.




(a) The input functional unit




(b) The output functional unit

Fig.3.3 The basic functional blocks.

The analogue signal output from the microphone is first amplified, then bandlimited to about 3.4kHz, and the bandlimited signal sampled at 8kHz. Each sample value is then encoded using eight binary digits and the codes stored in a RAM ( random accessed memory ) unit to be processed as and when required by the personal computer. The data analysis software is a program written in Turbo pascal that reads a given number of samples at a time of the digital data stored in the RAM memory, processes it according to the coding technique described in chapter four, and then stores the processed block in another RAM memory. The whole process of input-process-output is illustrated in a flow chart in chapter four.

This block of forty processed digital samples is output, sample-by-sample, to the digital- to- analogue converter and the reconstructed sample values are low-pass filtered to recover the analogue speech signal. The analogue signal is then fed to a loud speaker through an amplifier to reproduce the speech sound. The following sections describes the different components represented in block form in Fig.3.3.

## 3.4 COMPONENT DESCRIPTION

The main functional device used in this project is the IBM compatible personal computer. A brief description of the PC is therefore in place. The following is a description of a microcomputer in general with specific mention to the IBM compatible personal computer where necessary. A microcomputer's functions are determined through the interaction of its hardware components with its software. We shall thus divide our description into HARDWARE and SOFTWARE.

### 3.4.1 Microcomputer Hardware [60],[61]

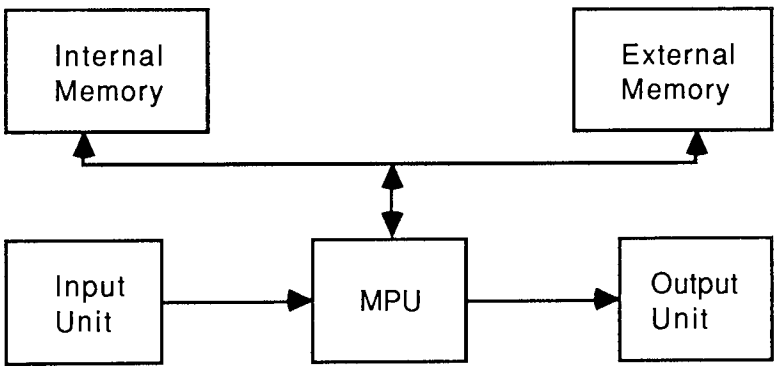



Fig.3.4 General Architecture of a Microcomputer

The above figure shows the general architecture of a microcomputer. The hardware of a microcomputer can be divided into four functional sections, the input unit, the microprocessor unit, the memory unit, and the output unit. Each of these units has a special function in terms of overall system operation.

The microprocessor unit (MPU) is the heart of a microcomputer. It is implemented on a single chip known as a microprocessor. This is a general-purpose device that comes under several different makes differing in capability.

In general, however, the main functions of a microcomputer are the same. These include performing arithmetic and logic operations, and directing and controlling the functions of other hardware blocks in the system. The

microprocessor's operations, in turn, are controlled by a set of instructions stored in the main memory. This set of instructions is known as the software of the microcomputer.

The IBM PC uses the Intel Corporation's 8088 microprocessor. The capability of a microprocessor is determined by the number of bits of data it can handle simultaneously. The Intel 8088 belongs to the company's 8086 family of 16-bit microprocessors.

The input and output units are the means by which the microcomputer communicates with the outside world. The input units, such as the keyboard on the microcomputer, allow the user to input information or commands to the microprocessor unit. Another input device used with personal computers is the mouse. The most widely used personal computer output devices are the display and the printer.

Information and the set of instructions that define the operations of the microprocessor are stored in the memory unit. Microcomputers use two types of memory, internal memory and external memory. External memory is used for long term storage of information that is not currently being used. It can hold files of data, program instructions, and files of information. Typical external memory devices are floppy and hard disks, although magnetic tapes and bubble memories can also be used.

Internal memory is a smaller segment of memory used for temporary storage of active information, such as the operating systems of the microcomputer, the program that is currently being executed and the data that are being processed. There are two types of internal memory, read-only memory (ROM) and random access or read/write memory (RAM). As their names imply, ROM can only be read while RAM can both be read and written into.

### 3.4.2 Software: The Computer Program

A computer on its own is near useless. It needs the second element, computer 'software', before it can do useful work. The type of software needed varies widely according to the kind of work the computer is going to be used

for.

Basically, there are two types of software; system software and application software. System software can, in turn, be classified as the basic input/output system, or BIOS, and the operating system, or OS. The BIOS consists of a set of instructions that controls all peripheral devices which may be connected to the personal computer. This set of instructions is stored in the ROM part of the internal memory during manufacture. ROM has the ability to retain its contents even when the system power supply is turned off and so BIOS is permanently held in the ROM. On the other hand, RAM is volatile, that is, it loses its contents when the system power supply is turned off.

The first set of instructions that the microprocessor executes when the microcomputer is turned on is the BIOS. These include checking for external peripheral devices, disk drives, and giving audible indication of correct operation.

The operating system consists of a collection of software essential to the operation of the computer. It performs two main roles:

(i) It enables the user to control the computer, to start particular programs running and to perform various housekeeping tasks such as viewing the contents of a disk, deleting unwanted files, copying information from place to place, etc.

(ii) The operation system looks after the fundamental operation of the computer and its components, storing and retrieving information from disks, organising data coming from the keyboard or being printed or displayed on the screen.

There are several operation systems, designed for different purposes. While some are multi-tasking and can load and execute several programs simultaneously, others are designed for only one program to be active at one time (single-tasking). Some operating systems split the computer processing power between several users at once, each user connected via their own keyboard and screen, while others can accommodate only one user at a time. These are known as multi-user and single-user operating systems respectively.

The type of operating system that a computer supports depends on the size

of the memory of the computer and the computer's processing power. Microcomputers have traditionally used single-user and single-tasking operating systems. An example is MS-DOS, the operating system used on the IBM PC and compatibles. This is designed by Microsoft, originally for the IBM PC but has now gained popularity far beyond the PC. It normally comes on a floppy or hard disk (thus the name 'DOS' for disk operating system) and is loaded on the computer to 'boot' it up.

There are two types of application software, programming languages and application packages. A computer programming language is a set of symbols and rules used to direct the operations of a computer. Dozens of computer programming languages are in common use, each primarily designed to solve certain kinds of problems. However, they all have a common purpose, to direct the operations of a computer.

In general, programming languages perform the following tasks:

(i) **Input and Output**. These instructions tell the computer to get data from the user or some input device and to present information on a display screen or printer.

(ii) **Calculation**. These instructions direct the computer to perform mathematical operations such as addition, subtraction, multiplication, and division.

(iii) **Comparison**. These instructions tell the computer to examine a number to see if it is less than, greater than, or equal to, another number.

(iv) **Data Movement, Storage, and Retrieval**. These instructions direct the computer's use of primary and secondary storage.

(v) **Transfer of Control**. These instructions tell the computer to deviate from the normal sequential processing.

Microcomputer programming languages can be classified as machine languages, assembly languages, and high-level languages. A machine language consists of binary codes that represent the microprocessor unit instructions,

memory addresses, and data needed to solve a particular problem. A machine language program can be used directly by a particular computer. However, programming in machine language is tedious and prone to mistakes.

The other two types of programming language; assembly language and high-level language, are more abstract as far as the computer is concerned and they need intermediate software instructions to convert their instructions into machine language instructions, the language which the computer understands and operates in.

An assembly language uses mnemonics to form a set of instructions to represent each binary coded machine language instruction. These instructions are then translated to their machine code equivalents by a program known as an assembler. High-level languages are very like a human language (English), which makes them easier to program in. Examples include pascal, C, FORTRAN, COBOL, etc. High-level languages are slower to execute as, by nature, they appear more complex to the computer. They thus need more complex interpreting programs known as compiler programs to interpret the high-level languages. The programs occupy more of the computer's memory.

Application packages are designed to help people do all kinds of work-a day jobs such as writing letters and planning budgets. They include word processing packages, data base packages, and spreadsheets.

### 3.4.3 Interfaces

Computers communicate with the outside world by means of input and output devices. These devices, such as keyboards, mice, display units, and printers, are known as peripheral devices. These peripheral devices communicate with the computer's CPU via interfaces.

Interfaces used to connect peripheral devices to microcomputers are either serial or parallel interfaces.

A **serial interface** or **serial port** transmits bytes of data between a computer and an external device serially, that is one bit at a time. In general, serial

interfaces are used to connect a computer to peripheral devices that either generate or accept data in serial form. Among other functions, therefore, a serial port serves as a serial-to-parallel and a parallel-to-serial data converter, since computer CPUs generally accept and output data in parallel form.

A standard serial interface used with many microcomputers is the **RS-232C**. The RS-232C defines the functions and voltage levels of 25 wires attached to connectors on a standardised plug socket, thus eliminating any compatibility problems due to different device and computer manufacturers.

A **parallel interface** or **parallel port** transmits an entire byte of data between a computer and an external device. The eight bits in each byte are sent simultaneously in parallel, with a single wire for each bit. Parallel interfaces can connect any device that either generates or receives data in parallel form.

Parallel input and output functions are performed in the IBM PC by a general-purpose LSI device known as the 8255. This chip contains three ports, PA, PB, and PC, each one byte long. In addition, there is a one-byte command register. All three ports and the command register are accessible via separate addresses. The three ports can be configured as input and output ports by feeding a code in the command register. Which of the three ports are configured as input and which as output depends on the code fed into the command register. A typical command code configures ports PA and PC as input ports, and port PB as an output port.

The command code, together with the set of instructions that cause either ports PA, PB, or PC to be selected and accessed, form part of the microprocessor input/output instruction set. This makes the functions of the 8255 chip totally software-controlled, and thus it is known as a programmable peripheral interface (ppi) device.

Computers in general are versatile machines. Their application is limited only by the user's imagination and the software available to him. For most practical applications, they serve as intermediate data processors, with the data originating in a form quite different from the digital in which computers

normally accept and process data. Moreover, the processed data will often be required in its original form. Thus, any device that can convert data from one form to digital form can serve as input for a computer, and any device that can convert digital data to any other form that may be required by an external device, can serve as output device to a computer. Of course, the particular device has to be connected to the computer via one of the interface ports discussed above, and which of the two interface ports used depends on how the device outputs or receives digital data, in serial or parallel form.

In this project the input data is speech. This originates in analogue form. The processed data must also be output as a speech signal. The input and output units described in section 3.2 convert the speech signal into digital form to be processed by the computer and then the digital output from the computer into analogue form to reconstruct the speech signal, respectively. They are linked to the computer via interface devices as illustrated in Fig.3.3. As the analogue-to-digital converter chosen outputs data in serial form, the input device in Fig.3.3 will be connected to the PC via the serial port. Also, a digital-to-analogue converter accepts parallel bits of data to be converted to analogue form. Hence, the output device illustrated in the figure will be connected to the PCs parallel port.

The computer used in this project was an Opus PC III personal computer. It had a hard disk of 20 mega bytes capacity and a single floppy drive for a 5.25 floppy disk. Its memory capacity was 640 Kbytes. The processor was an NEC version of Intel's 8088 microprocessor driven at a clock rate of 4.8kHz.

### 3.4.4 Input and Output Units

The input unit was a data acquisition device designed for an undergraduate project in speech recognition. Basically, a data acquisition device accepts analogue data as input and outputs a digital version of the input data. Hence the main functions in a data acquisition device are as illustrated in (a) of Fig.3.3. By its design, the system could accept only a single word at a time, spoken rapidly enough for the input program to "capture" it. It was interfaced to the

Opus PC III through the computer's RS serial input-output port, and a program, "capture", read the samples of speech processed by the data acquisition system into the computer's memory and the entire file stored on disk.

The output system was designed and built specifically for the project. It consisted of a digital-to-anlogue converter chip, plus its glue logic built on the same circuit board. The output from this arrangement was input to a loud-speaker via a low-pass filter and an amplifier, also in i.c form. The system was connected to the PC through an 8255 input-output board. A program written in turbo pascal read the processed file sample-by-sample and output the samples to the output system through the input-output port. The speech waveform was then reconstructed by first reconstructing the sample levels by the digital-to analogue converter and then low-pass filtering the sequence of levels. The waveform thus generated was amplified and then fed to the loud speaker which generated the corresponding speech sound.

In between the input and output processes, the input speech samples were coded using the algorithms discussed in chapter four. The coding process is illustrated in a flow chart in the same chapter.

Two techniques were used in the coding process. In one technique the speech samples were coded in blocks of 100 consecutive samples, the coefficients being updated for each block. This technique is known as the forward adaptive technique as the predictor coefficients are updated based on the input sample values. In the other technique, the whole speech file was coded sample-by-sample with the predictor coefficients updated every sample period based on information of the signal error at the output of the quantizer. This technique is known as the backward adaptive technique. The two techniques are discussed in detail in chapter four.

# CHAPTER FOUR

# ALGORITHMS

## 4.1 INTRODUCTION

The aim of this project is to encode speech at about 8kbps for communications purposes using adaptive predictive coding. At such low bit rates, the performance of the traditional waveform-following coders, such as delta modulation or differential pulse code modulation, is marginal, in terms of the quality of the coded speech. However, these coders are robust to background and other transmission impairments and are reasonable in complexity and cost. On the other hand, linear predictive coders (LPC), while achieving acceptable speech quality and intelligibility at approximately 4kbps, are very complex and somewhat susceptible to background impairments [8],[9].

Based on these facts, it is generally felt that a compromise between complexity, robustness and quality may be attainable by designing a waveform-following coder similar to delta modulation, but with slightly increased complexity, that operates at an intermediate transmission rate. The resulting coder is one based on a combination of differential pulse code modulation and linear predictive coding. This arrangement has been described in detail in chapter two.

The performance of such an encoding system depends on the two components that make up the coder, the predictor and the quantizer and, in particular, the parameters that make up these components, the predictor coefficients and the quantizer step size. These parameters determine the overall performance of the coder. In particular, the performance of the predictor is affected by quantizer parameter variations and vice versa.

To obtain adequate speech quality at bit rates as low as the target bit rate of this project, it is necessary to adapt both the predictor and the quantizer. This can be done using two schemes, forward adaptive and backward adaptive.

The forward adaptive scheme uses various algorithms to compute the desired parameters ( predictor coefficients and/or quantizer step size ) based on the actual input signal. This involves collecting several milliseconds of the input speech data into a frame or block and computing the predictor and quantizer parameters for the system from the block using the appropriate algorithms. The computed parameters are then quantised and transmitted to the receiver to generate the output signal. The effect of this transmission of side information is to reduce the rate available to transmit the quantised prediction error { $e_q(k)$ }.

In the backward adaptive technique the parameters are adapted on a sample-by-sample basis using past values of the reconstructed sequence {s(k)} which is available at both the transmitter and the receiver. This technique thus allocates all the total transmitted data rate to the quantised prediction error.

Both techniques have their advantages and disadvantages, depending on bit-rate requirements and transmission channel conditions. For example, backward adaptive schemes update the parameters on a sample-by-sample basis and thus do not involve any delay. Also, since it does not require the transmission of side information, a backward adaptive scheme may be preferred to a forward adaptive scheme in limited bit-rate circumstances. However, the backward adaptive system requires a more complex receiver than the forward adaptive system since the adaptive prediction algorithm must be duplicated at the receiver. Also, backward adaptive coders are characterised by complex algorithms that involve several hundreds of computations. This makes them a lot slower than their forward counterparts.

On the other hand, the coding delay inherent in forward adaptive schemes may contribute significantly to echo problems if several re-encodings are required after analogue switching. Also, because of side information, a forward adaptive scheme actually has higher transmission data rates than a backward adaptive system for a given system design requirement. However, forward adaptive parameters can be easily error-protected before transmission. Thus, under noisy channel conditions, forward adaptive schemes may have a significant advantage over backward adaptive schemes which must adapt the

parameters on the ( possibly erroneous ) transmitted prediction.

The two adaptive algorithms on which the two coders tested in this project are based are discussed in this chapter. In furtherance of the aims of this project, a low bit-rate coder of minimum complexity, a two-level differential pulse code modulator is used. In addition to its simplicity, a two-level dpcm has the added advantage of being robust to channel and transmission errors. This is due to the fact that in delta modulation the prediction error is quantised into one of two levels and the level encoded into one of two binary digits, a 1 or a 0. This binary code is then transmitted and all that needs to be done at the receiver is to reconstruct the transmitted level based on a simple decision of whether the received code is a binary 1 or 0. The reconstructed error is then added to or subtracted from the predictor output at the receiver. In this project, +1 is transmitted if the prediction error is positive and -1 transmitted if it is negative. The reconstruction error is therefore added to the predictor output at the receiver for +1 transmitted and subtracted from the predictor output for -1 transmitted. In this way, the coding process at the receiver becomes merely a comparator and decision-making process.

The next two sections discuss forward and backward algorithms, the algorithms used in the coding systems of this project. The performances of the two coders are compared and discussed in chapter five. As no coding system can be complete without a quantizer, or some mechanism by which digital coding can be achieved, each of the two algorithms discussed below is divided into two paths: the predictor coefficient adaptation algorithms and the quantizer step size adaptation algorithms. In both algorithms the technique of predictor coefficient extraction is based on the same principle of minimising the mean square prediction error.

## 4.2 FORWARD ADAPTIVE ALGORITHMS

### 4.2.1 Predictor Coefficient Adaptation Algorithm

#### Introduction

The speech waveform was first divided into blocks of one hundred samples each. The blocks were then sequentially processed as follows:

For each block of 100 samples, a short-term prediction error was formed. The error was then mathematically analysed to minimise its mean square value. Using the autocorrelation technique, intermediate values known as reflection coefficients were first computed from the minimum mean-square error. The predictor coefficients were recursively computed from the reflection coefficients using Durbin's recursive method. The above process was then repeated for the next block of 100 samples, and the next, and so on, until the predictor coefficients had thus been computed for every block of 100 samples in the sequence.

Each block was linearly predicted sample by sample using the predictor coefficients computed for that block and the prediction errors encoded and transmitted as discussed in chapter two.

At the same time, the reflection coefficients for each block were quantised and coded and the codes transmitted. At the receiver, the codes were decoded into the respective transmitted reflection coefficients, and the corresponding predictor coefficients computed from these, using the same recursive technique as at the transmitter. A predictor of order 4 was used in this project. Thus four reflection coefficients, and hence four predictor coefficients, were computed for each block of 100 samples.

The reasons for choosing the reflection coefficients rather than the predictor coefficients for transmission are discussed in detail later in this section. It will be shown that the reflection coefficients are more suitable for transmission as they guarantee filter stability, even after quantization. The choice of the authocorrelation method was based on the choice of transmission parameters. The autocorrelation method yields the reflection coefficients as intermediate

parameters so that the predictor coefficients are derived from them. A detailed analysis of the various techniques discussed above is now presented.

## EXTRACTION OF THE PREDICTOR COEFFICIENTS

[5],[16],[23],[57]

The linear predictive method of speech production is based on modelling the vocal tract, the speech production organ in the human mouth, by an all-zero digital filter given as[5]:

$$H(z) = \frac{G}{1 + \sum_{i=1}^{p} a_i z^{-i}} \qquad 4.1$$

where p is the order of the model.

The parameters, the $a_i$'s ( the predictor coefficients ) and G ( the gain ), are determined by minimising the squared error

$$\sum_{n} \left[ S(n) - \hat{S}(n) \right]^2 \qquad 4.2$$

over all the available samples, where

$$\hat{S}(n) = \sum_{i=1}^{p} a_i S(n - i) \qquad 4.3$$

Minimisation of the total squared error with respect to $a_i$ leads to the following set of linear equations:

$$\begin{aligned}
a_1 r(0) + a_2 r(1) + \ldots\ldots + a_p r(p - 1) &= -r(1) \\
a_1 r(1) + a_2 r(0) + \ldots\ldots + a_p r(p - 2) &= -r(2) \\
&\vdots \\
a_1 r(p - 1) + a_2 r(p - 2) + \ldots\ldots + a_p r(0) &= -r(p)
\end{aligned} \qquad 4.4$$

or, in matrix form

$$\mathbf{R}.\mathbf{a} = -\mathbf{r} \qquad 4.5$$

where

$$\mathbf{r}^T = [\, r(1)r(2) \ldots\ldots r(p) \,] \qquad 4.6$$

$$\mathbf{a}^T = [\, a_1 a_2 \ldots\ldots a_p \,] \qquad 4.7$$

$$R = \begin{bmatrix} r(0) & r(1) & \ldots & r(p-1) \\ r(1) & r(0) & \ldots & r(p-2) \\ \vdots & \vdots & \vdots & \vdots \\ r(p-1) & r(p-2) & \ldots & r(0) \end{bmatrix} \qquad 4.8$$

The above equations are obtained by defining

$$r(i) = r(-i) = \sum_{n=0}^{N-i-1} S(n)\, S(n+i) \qquad 4.9$$

to be the ith autocorrelation, where N is the number of samples in the block (frame). This means that the signal $S(n) = 0$ for $n < 0$ and $n >= N$. Then the ith autocorrelation is computed by shifting the signal by i samples. This formulation is called the autocorrelation method and produces a matrix R that is a Toeplitz matrix, i.e., one whose diagonals are composed of identical elements. Such a matrix is nonsingular and it can always be inverted yielding a solution

$$\mathbf{a} = -R^{-1}\mathbf{r} \qquad 4.10$$

We note that the matrix R as defined in equation 4.9 is symmetric. This property is due to selecting the data S(n), S(n - 1), and so on, in a sequential manner back to S(n-N+1) [16].

Durbin's recursive method is most suitable for solving a Toeplitz matrix. This method provides a stage-by-stage, or order recursive method in which a solution for the pth order predictor may be computed from the knowledge of the optimal (p - 1)th order predictor. The method is as follows:

Starting with the autocorrelation coefficients r(i), i = 0, 1, ....., p we proceed to compute recursively the filter coefficients, $a_i$, from the following relations:

$$E(0) = r(0) \qquad 4.11$$

$$k_i = \{r(i) - a_i^{(i-1)} r(i-1) - ....- a_{i-1}^{(i-1)} r(1)\}/ E(i-1) \qquad 4.12$$

for i = 1, ...., p

$$a_i^{(i)} = k_i \qquad 4.13$$

$$a_j^{(i)} = a_j^{(i-1)} + k_i a_{i-j}^{(i-1)} \qquad 4.14$$

$$j = 1, ..., i-1$$

$$E(i) = [1 - k_i^2] E(i-1) \qquad 4.15$$

The coefficients, $a_j^{(i)}$, j = 1, ...., i, are the filter coefficients of an ith order model. Hence the coefficients of the desired pth - order model are

$$a_j = a_j^{(p)}, j = 1, ...., p \qquad 4.16.$$

We note from the above that a natural consequence of Durbin's method is a set of alternative parameters, $k_i$, i = 1, ...., p, known as the reflection coefficients. These coefficients are very important parameters in that they are equivalent to the filter coefficients, $a_i$, and the latter coefficients can be derived from the reflection coefficients and vice versa. This is important when it is necessary to transmit the coefficients. It is preferable to transmit the reflection coefficients and then derive the predictor coefficients from them using equations 4.13 to 4.16. The choice of the reflection coefficients for transmission rather than the predictor coefficients is due to the fact that only the reflection coefficients have a property guaranteeing the stability of the filter. This property states that for a stable filter, i.e, an LPC filter with all the poles inside the unit circle, $-1 \leq k_i \leq 1$, i = 1, ..., p. This condition ensures the stability of the filter even after quantization. Also, the well-defined range of values that ki can

take, (-1 to +1), makes quantization easier.

## Quantization Of The Predictor Parameters [53],[54]

The predictor coefficients determined above were used to predict an input sample value at both the transmitter and the receiver for a particular block of sample values. It was thus necessary that the computed predictor coefficients be transmitted to the receiver. To do this it was necessary to first quantise and encode the individual coefficients, and then transmit the codes. This did not result in any significant addition in bit-rate, as the coefficients could tolerate coarse quantization. Also, the coefficients were only transmitted once for each block of samples. This was done at the beginning of the block time.

The most suitable coefficients for transmission are the reflection coefficients, the $k_i$'s since these guarantee filter stability after quantization. The $a_i$'s, though they offer a more straightforward solution, resulting in a reduced computation complexity, do not guarantee filter stability after quantization. Worse still, the natural ordering of the parameters may not be maintained, resulting in a different predictor at the receiver from the one at the transmitter. On the other hand, the reflection coefficients possess this natural ordering ability. The reflection coefficients can be quantised either linearly or after being suitably transformed. There are two transformation techniques, the inverse sine technique and the log-area technique. Linear quantization of the transmission parameters was found to be quite adequate for this project.

Thus, given a set of reflection coefficients { $k_i$ } at the receiver, the corresponding predictor coefficients can be computed from the following recursion

$$a_i^{(i)} = k_i \qquad 4.17$$

$$a_j^{(i)} = a_j^{(i-1)} - k_i a_{i-j}^{(i-1)}, \quad 1 < j < i - 1 \qquad 4.18$$

These two equations are solved for i = 1, 2, ...., p, with the final set obtained as

$$a_j = a_j^{(p)}, j = 1, ....., p \qquad 4.19$$

### 4.2.2 Quantizer Step Size Adaptation [5],[52]

The step size is computed at the transmitter using input sample values in much the same way as the predictor coefficients. Hence, like the predictor coefficients, the advantage of forward step size adaptation is that the quantizer step size adaptation is based on input uncorrupted samples and that the step size can be adapted on a window basis rather than on a sample-by-sample basis, which results in far less computation. However, like the coefficients, there is need for the step size to be transmitted as side information, thus further contributing to the total transmission bit-rate.

Two different techniques of adapting the quantizer step size where adopted in this part of the project. One technique adapted the step size based on the following mathematical expression:

$$\delta(k) = \alpha \sqrt{\frac{1}{N}\sum_{i=1}^{N} S^2[(k-1)N+i]} \qquad 4.20$$

where $\delta(k)$ denotes the step size for the kth block of samples, and $0 < \alpha < 1$, i = 1,2,....,N, N being the number of samples in a block. k = 1,2,....., denotes the kth block.

According to the above expression, the step size is a function of the block

$\delta(3)$ <..... and so on. However, in practice it is restricted to a range $\Delta_{min}$ < $\delta(k) < \Delta_{max}$. The ratio $\Delta_{max}/\Delta_{min}$ determines the dynamic range of the system. In this project this ratio was chosen to be 100.

The alternative technique is based on the difference between adjacent speech samples in a block. Starting with the first sample to the last sample in a block, the difference between a sample and the one preceding it is computed and the value stored without the sign. The step size for the particular block of samples is a linear function of the maximum of these difference positive values. The mathematical expression is as follows:

$$\delta(k) = \alpha \max\{|S(k-i) - S(k-i-1)|\} \qquad 4.21$$

over all i = 1,2,.....,k

where < $\alpha$ < 1 and S(k) is the speech sample at time interval k.

Algorithms based on the two mathematical expressions were written in Turbo Pascal and the algorithms tested experimentally. However, due mainly to operational problems, the experiment was unsuccessful. In particular, the computer could not cope with the number of floating point operations involved due mainly to the absence of a maths co-processor, but also due to the limited memory capacity.

For this reason, coupled with the need to minimise the amount of side information transmitted in view of the low bit rate target of this project and the need to maintain the same quantization technique for both coders, the forward technique of adapting the quantizer step size was not pursued further. The backward technique of adapting the predictor coefficients is discussed next, together with the corresponding technique of adapting the step size, the technique employed in both coders in this project.

## 4.3 BACKWARD ADAPTIVE ALGORITHMS [8],[9],[52],[58],[59]

### 4.3.1 Predictor Coefficient Adaptation Algorithm

The algorithm used in this part of the project was based on a generic form of algorithms known as stochastic approximation predictors (SAP) or adaptive gradient predictors. These adapt the coefficients on a sample-by-sample basis. In general, the kth coefficient is formed according to the following expression [59]:

$$a_k(i+1) = a_k(i) + P_i(x)e(i)x(i-k) \qquad 4.22$$

where $P_i(x)$ controls the rate of convergence of the predictor and depends on the mean square value of $\{x(i)\}$, $e(i) = x(i) - y(i)$ is the prediction error at the ith instant. The kth coefficient is computed, in general, by minimising the mean square prediction error, as in the case of the forward adaptive technique discussed earlier. The algorithms in the above generic form differ only in their method of achieving the mean square error minimisation.

Generally, it is common practice when computing a new coefficient estimate to multiply the immediately preceding estimate by a positive scaler less than but near to unity in magnitude. This is to limit their memory requirement as the algorithms are infinite impulse response (IIR) systems due to their recursive nature. Multiplying the past coefficient estimates by this damping factor causes the coefficients to decay toward zero over a long period of time [9].

The algorithm used in this project is a modified version of equation 4.22, with the error, e(i), and the data, x(i - k), replaced by their respective signs. Also, the quantised reconstructed data value was used instead of the input data value x(i-k). $P_i(x)$ was computed at each sampling period as follows[59]:

$$P_i(x) = \frac{A}{B + \frac{1}{M}\sum_{j=i-1-M}^{i-1} x_j^2} \qquad 4.23$$

where A and B are constants. The complete algorithm for computing the (k+1)th predictor coefficient from the kth predictor coefficient at the ith sampling interval then was as follows:

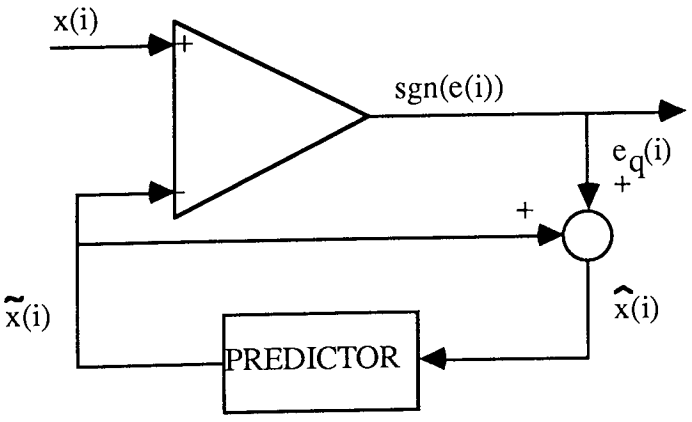
$$a(k+1) = a(k) + P_i(x)\,\mathrm{sgn}(e(i))\,\mathrm{sgn}(\hat{x}(i-k)) \qquad 4.24$$

where sgn(x) = 1 if x > 0, 0 if x = 0, -1 if x < 0 [22] 4.25

The use of the signs in equation 4.24, rather than the actual values, makes an otherwise complex expression a lot simpler. The choice of this sign-sign algorithm was also made to suit delta modulation, the quantization technique used in this project. This, as was discussed in chapter two, deals with signs rather than the actual values.

Fig. 4.1 Illustrates how the predictor coefficients are computed using the sign-sign technique. $e_q(i)$ is the reconstructed error signal.

$\hat{x}(i)$ is the reconstructed quantized version of the input sample value, x(i), and $\tilde{x}(i)$ is the predicted sample value based on past reconstructed sample values.




$\hat{x}(i)$ is the reconstructed quantized version of the input sample value, x(i), and $\tilde{x}(i)$ is the predicted sample value based on past reconstructed sample values.

Fig.4.1 Computation of predictor coefficients.

The values for the constants A, B, and M were determined by experiment as 10.0, 100.0, and 100.0, respectively.

### 4.3.2 Backward Quantizer Step Size Adaptation

Throughout the project, the delta modulator step-size was adapted backwardly, i.e., based on information available about the signal at the output. The following technique was used to compute the step size at the ith quantization interval:

$$q(k) = bq(k-1) + (1-b)q_{min} + f(k) \qquad 4.26$$

where $q_{min}$ is the minimum quantizer step size, $0 < b < 1$ is a parameter to be determined, and f(k) is determined as follows;

$f(k) = (1-b)(q_{max} - q_{min})$, if bit(k) = bit(k-1) = bit(k-2), or

$f(k) = 0$, otherwise 4.27

where $q_{max}$ is the maximum step size ( to be determined), bit(k) is the transmitted binary digit (0 or 1) at the kth sampling interval. f(k) is the overload factor, i.e., the factor by which the step-size is further increased at the occurrence of slope overload as determined by the occurrence of three successive bits of the same polarity.

The values of the above constants as determined by experiment were as follows:

$q_{max} = 25.0$, $q_{min}$ 0.0, and $b = 0.95$. These are the values that gave the best coder performance.

## 4.4 CODING ALGORITHMS

### 4.4.1 Forward Adaptive Coding Algorithm

Two techniques were followed in this section of the project. In one technique a given number of samples (one hundred) are read from a file and processed by first extracting the predictor coefficients as explained in section

4.2.1. Each sample in the block is coded using adaptive predictive coding techniques as explained in chapter three, and then transmitted. The next block of samples is then read and processed until all the samples in the file (or the required number of samples) have thus been dealt with. In the other technique the whole file (or the required number of samples) is read into memory and then partitioned into blocks of one hundred (or any given number) of samples each. The predictor coefficients are then extracted for each block, after which the file is coded on a sample by sample basis using the adaptive predictive coding technique. A new set of predictor coefficients is used for each block of samples which has been computed earlier for that block of samples. The advantage of this technique over the previous one is that transmission is continuous from the first sample to the last in the file. It is also faster and more robust to quantization noise.

The following algorithm illustrates the first technique. The second technique is described in the following flowchart and the Turbo pascal version is listed in the Appendix.

**Transmit**

1. Open a file

2. Initialise counter

**count := 0**

3. Read the next one hundred samples from the file and store them in

**s(1), s(2), s(3), ....., s(100)**

4. Compute the predictor coefficients

**a(1), a(2), a(3), a(4)**,refer to section 4.2.1

5. Increment counter

**count := count + 1**

6. Compute an estimate of the next sample and store it in

$\tilde{S}$(count)

I.e., save the prediction coefficients.

7. Compute difference as

Diff := S(count) - $\widetilde{S}$(count)

8. If Diff is positive then

**output binary 1**

**else**

**output binary 0**

9. If count = 1 then

Transmit prediction coefficients together with the output code ( 1 or 0 )

else

Transmit the output code only

I.e. the predictor coefficients for each block of one hundred samples are transmitted at the start of the block.

10. Compute the stepsize and store in

**q(count)**

11. Reconstruct sample as

$\widehat{S}$(count) := $\widetilde{S}$(count) + q(count)

12. If count = 100 then

**Go to step 3** and read the next block of samples.

**else**

**Go to step 5** to the next sample in the block.

13. Continue until end of file

14. Halt

**Receive**

1. If count = 1 then

Receive new set of prediction coefficient together with the code ( 0 or 1)

else receive only the code

2. Compute prediction value of the present sample as in the transmit above, and save prediction coefficients.

3. Determine new stepsize as in the transmit section

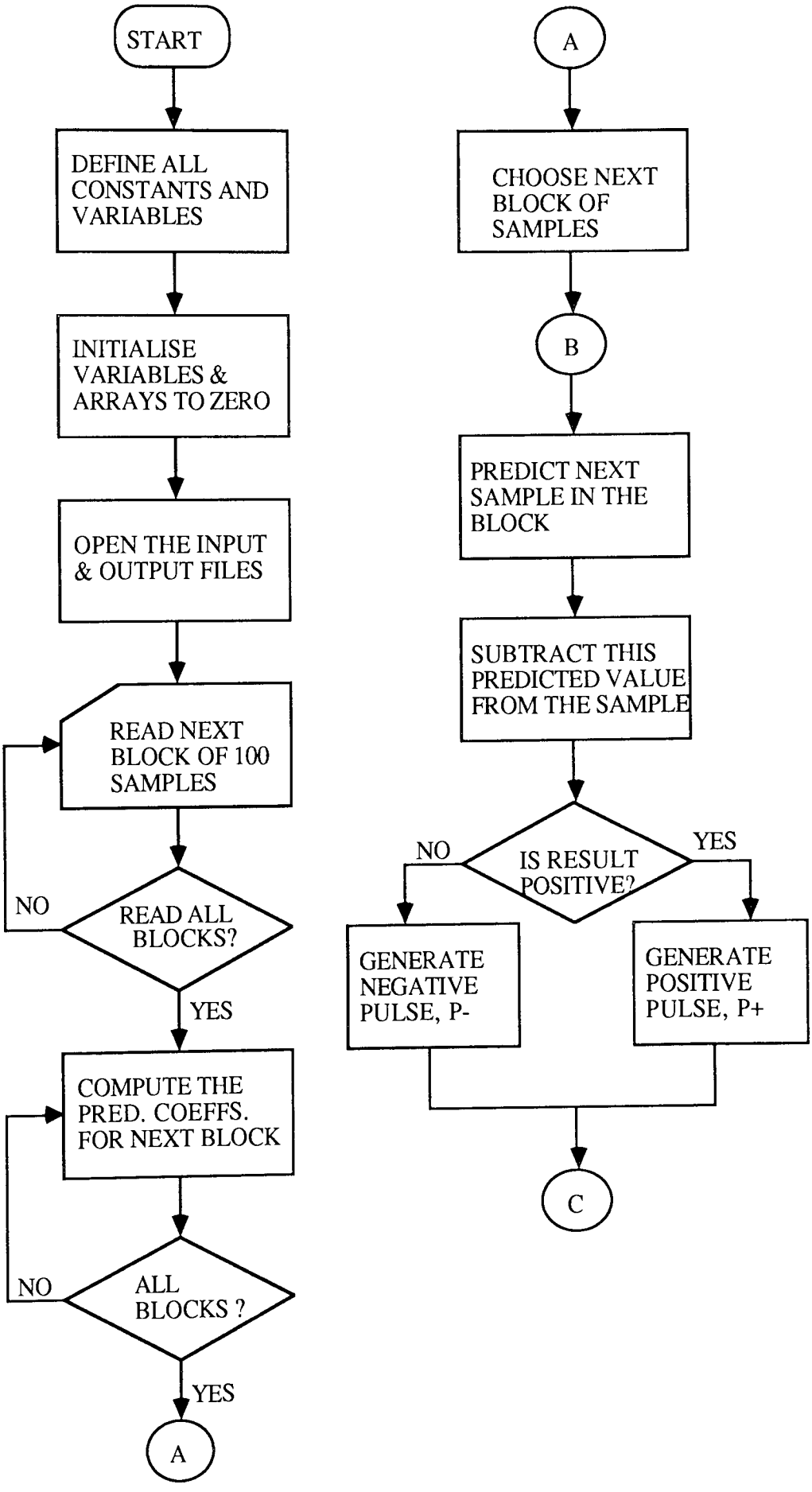
4. Reconstruct present sample as in the transmit section

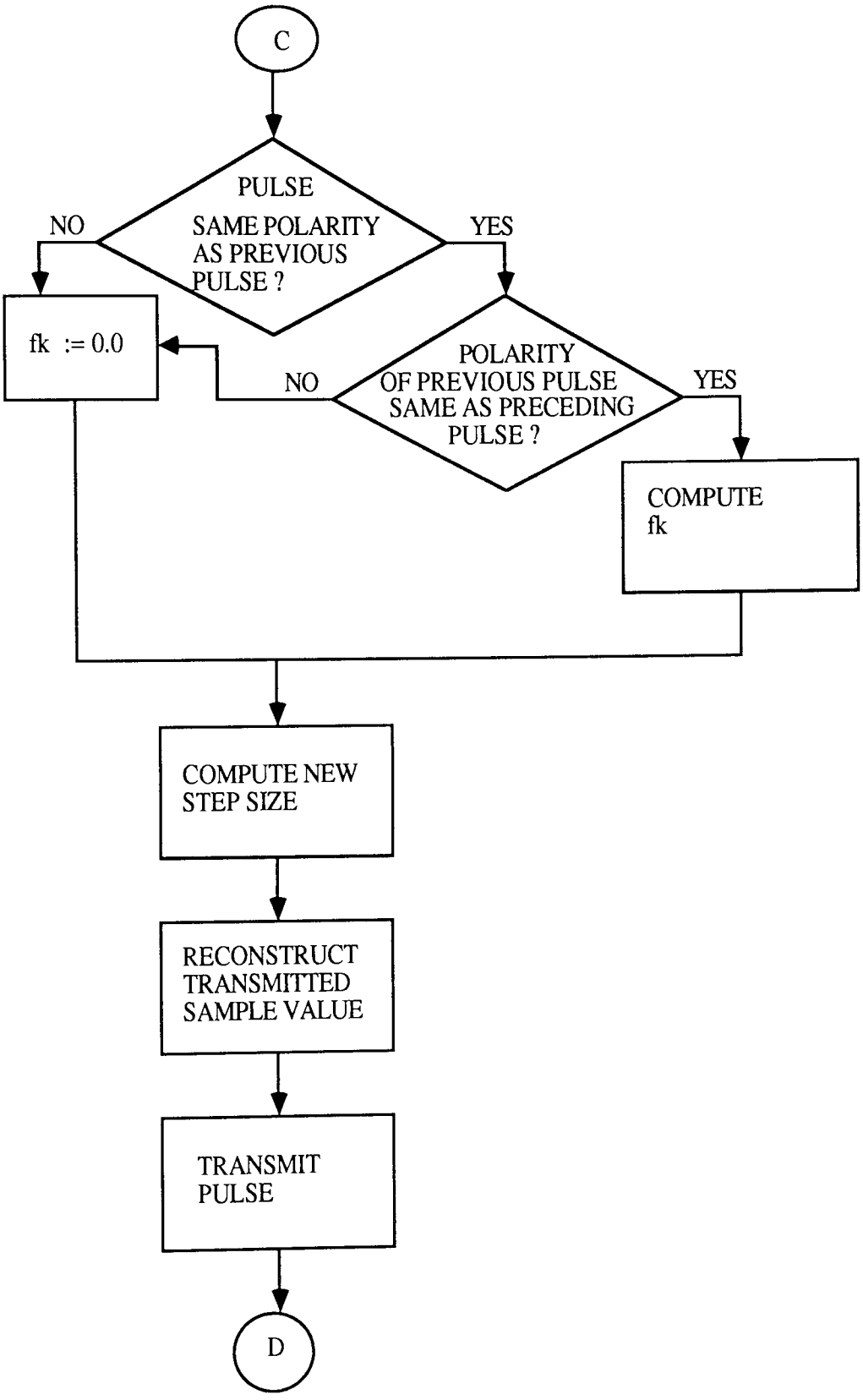
5. Goto step 1

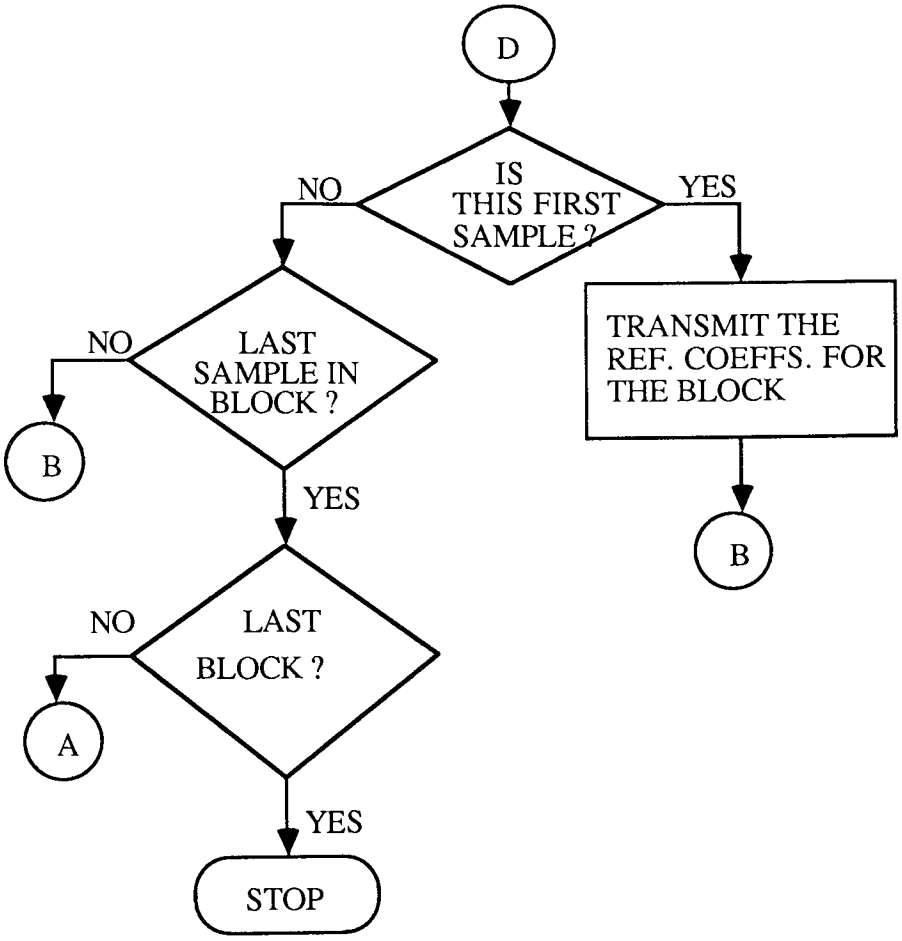
6. Continue until end of file

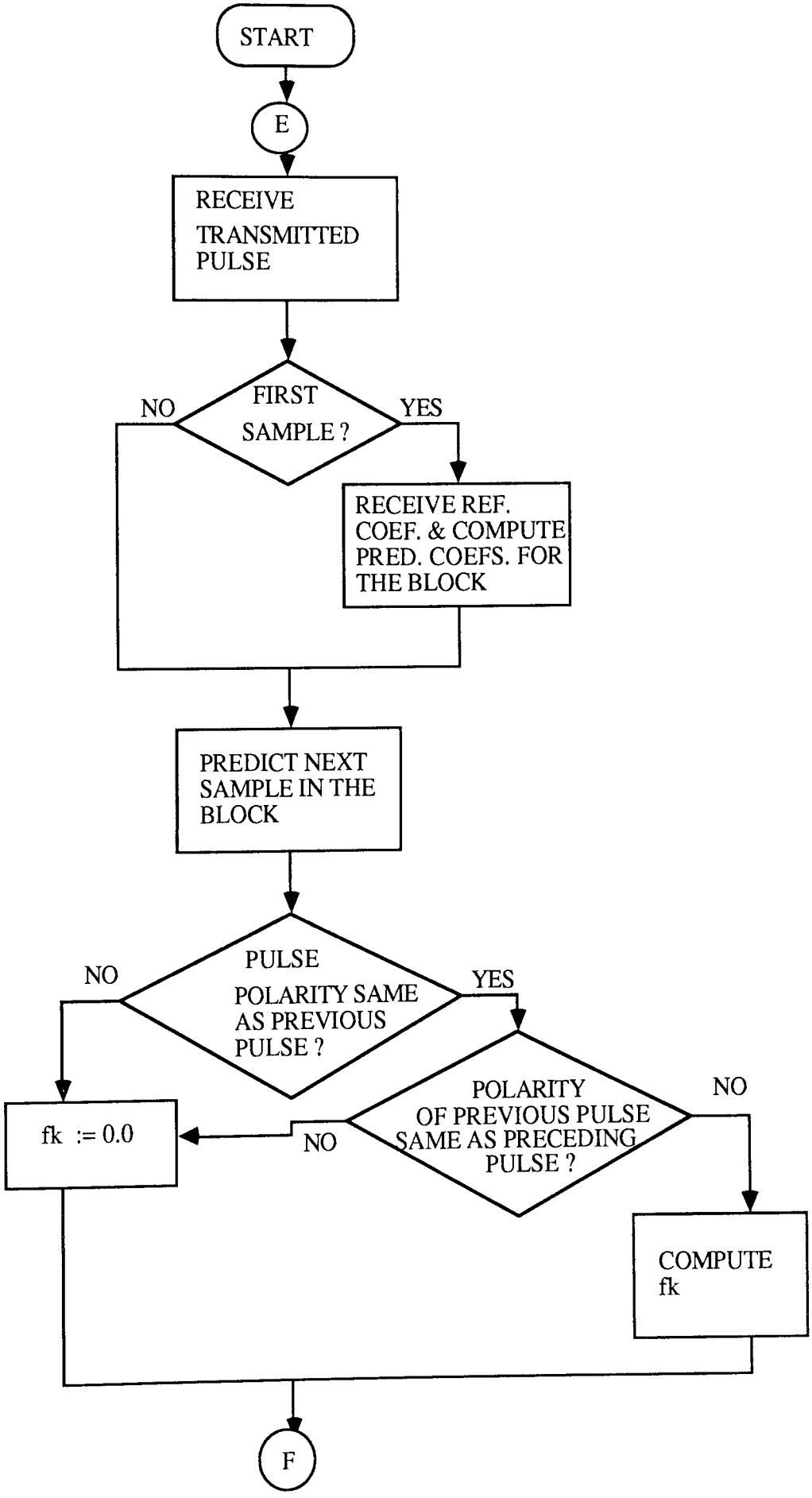
7. Halt

Fig. 4.2 illustrates the coding system based on the second algorithm.


START
DEFINE ALL CONSTANTS AND VARIABLES
INITIALISE VARIABLES & ARRAYS TO ZERO
OPEN THE INPUT & OUTPUT FILES
READ NEXT BLOCK OF 100 SAMPLES
READ ALL BLOCKS?
NO
YES
COMPUTE THE PRED. COEFFS. FOR NEXT BLOCK
ALL BLOCKS ?
NO
YES
A
A
CHOOSE NEXT BLOCK OF SAMPLES
B
PREDICT NEXT SAMPLE IN THE BLOCK
SUBTRACT THIS PREDICTED VALUE FROM THE SAMPLE
IS RESULT POSITIVE?
NO
YES
GENERATE NEGATIVE PULSE, P-
GENERATE POSITIVE PULSE, P+
C


C
PULSE
SAME POLARITY
AS PREVIOUS
PULSE ?
NO
YES
fk := 0.0
POLARITY
OF PREVIOUS PULSE
SAME AS PRECEDING
PULSE ?
NO
YES
COMPUTE
fk
COMPUTE NEW
STEP SIZE
RECONSTRUCT
TRANSMITTED
SAMPLE VALUE
TRANSMIT
PULSE
D


D
IS THIS FIRST SAMPLE ?
NO
YES
TRANSMIT THE REF. COEFFS. FOR THE BLOCK
B
NO
LAST SAMPLE IN BLOCK ?
B
YES
NO
LAST BLOCK ?
A
YES
STOP


START
E
RECEIVE TRANSMITTED PULSE
FIRST SAMPLE ?
NO
YES
RECEIVE REF. COEF. & COMPUTE PRED. COEFS. FOR THE BLOCK
PREDICT NEXT SAMPLE IN THE BLOCK
PULSE POLARITY SAME AS PREVIOUS PULSE ?
NO
YES
POLARITY OF PREVIOUS PULSE SAME AS PRECEDING PULSE ?
NO
NO
fk := 0.0
COMPUTE fk
F

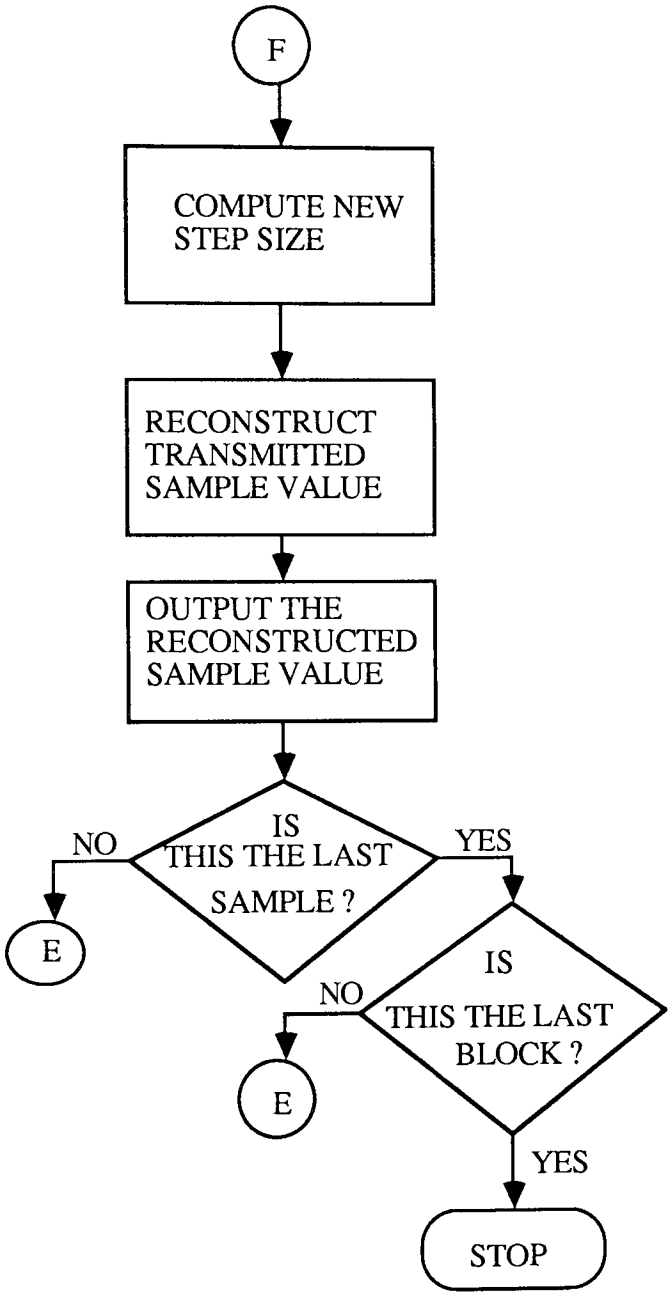
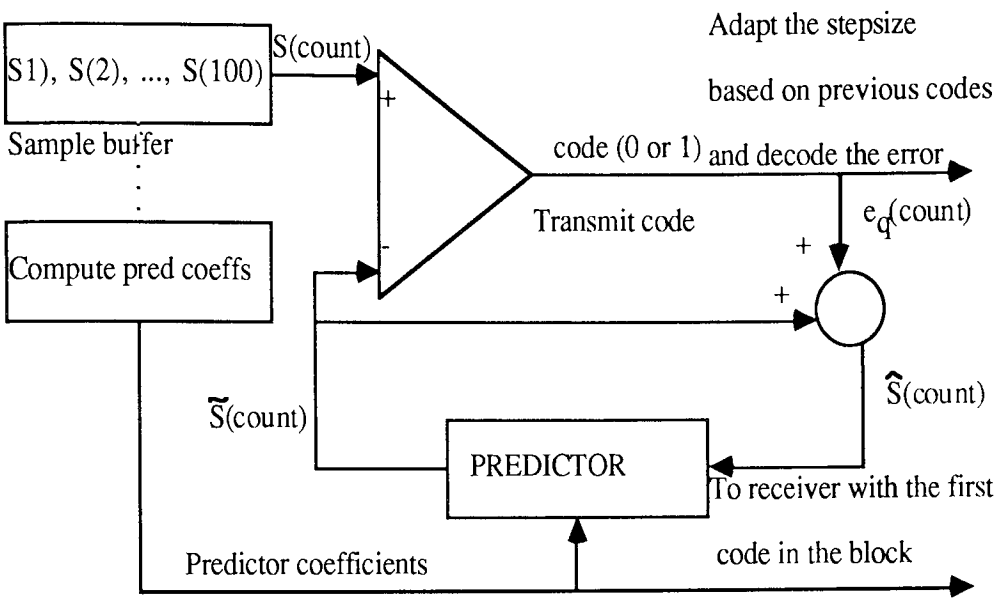



Fig 4.2. A flowchart representation of the Forward Adaptive coding process.




$\hat{S}$(count) is the reconstructed quantized version of the speech sample, S(count), where 'count' is the position of the speech sample in the buffer of speech samples. $\tilde{S}$(count) is the estimation of S(count).

Fig. 4.3 Block Diagram representation of the Forward Adaptive Coding Process.

### 4.4.2 Backward Adaptive Coding Algorithm

This algorithm reads a file (or a required number of samples in the file) into memory and then proceeds to code and transmit the samples sequentially using the adaptive predictive coding technique with the predictor coefficients adapted on a sample-by-sample basis as explained in section 4.3.2. The following software algorithm illustrates how this technique was implemented in this project. The flowchart representation of the coding process is given in figure 4.4. As in the case of the forward adaptive algorithm discussed above, the Turbo Pascal version of this algorithm is listed in the appendix.

**Transmit**

1. Open a file
2. Initialise counter

**count := 0**

3.Initialise the predictor coefficients

**a(1), a(2), a(3), a(4)** to zero

4. Read eight thousand samples from the file and store them in

**S(1), S(2), S(3), ....., S(8000)**

In this project the files were more than 8000 samples long but it was only possible to store 8000 in memory at a time, which was what was needed for speech.

5. Increment counter

**count := count + 1**

6. Compute an estimate of the next sample, S(count) and store it in

$\widetilde{S}$(count)

7. Compute difference as

Diff := S(count) - $\widetilde{S}$(count)

8. If Diff is positive then

**output binary 1**

**else**

**output binary 0**

9. Compute the stepsize (refer to section 4.3.2) and store in

**q(count)**

10. Reconstruct sample as

$\widehat{S}$(count) := $\widetilde{S}$(count) + q(count)

11.Compute new predictor coefficients (refer to section 4.3.1)

12.**Go to step 5** to the next sample in the block.

13. Continue until end of file, i.e., until count := 8000

14. Halt

**Receive**

1. If count = 1 then

Initialise the predictor coefficients as in step 3 above

2. Receive transmitted code (0 or 1)

3. Compute prediction value of the present sample as in the transmit above, and save prediction coefficients.

4. Determine new stepsize as in the transmit section

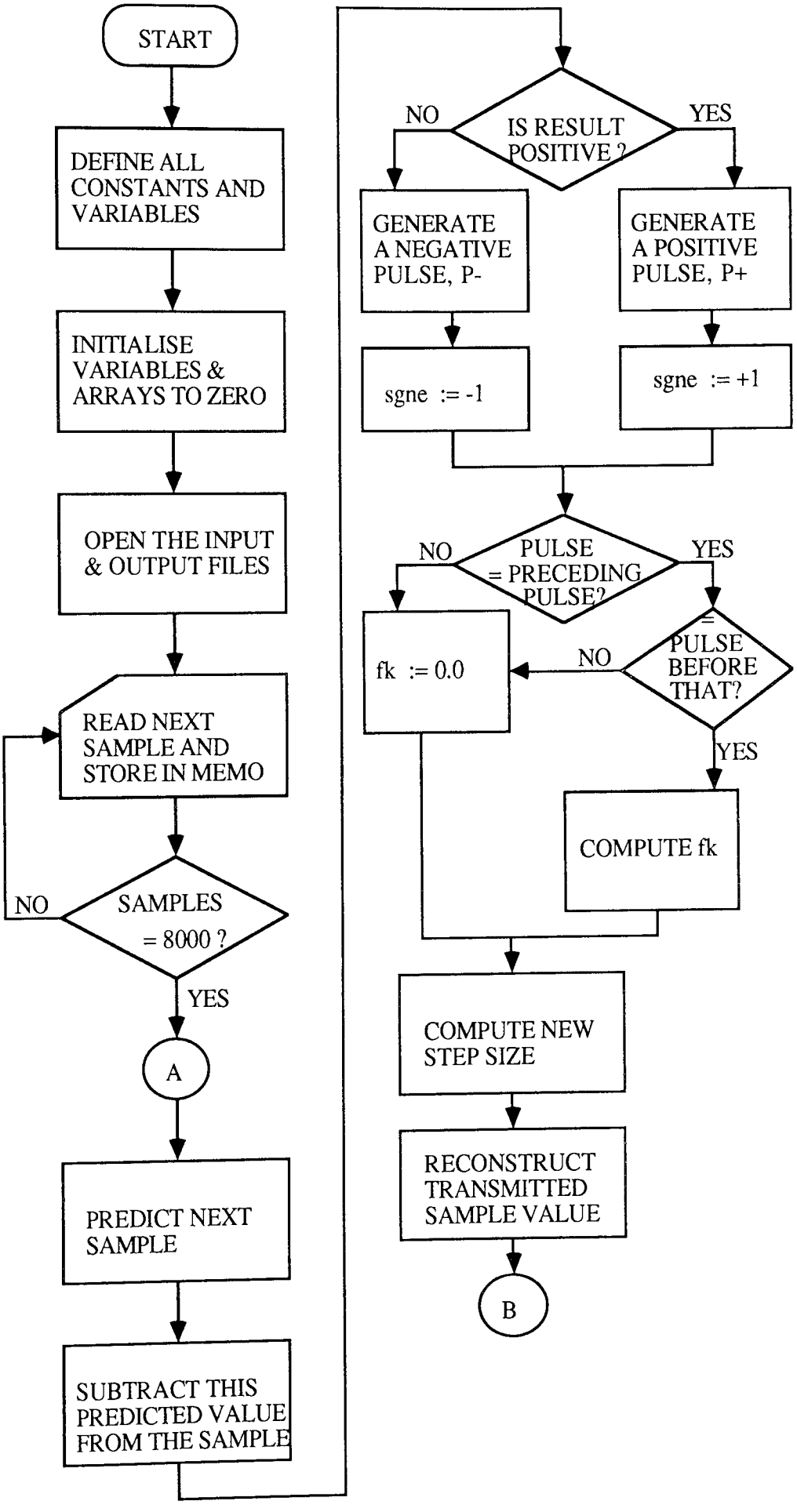
5. Reconstruct present sample as in step 10 above

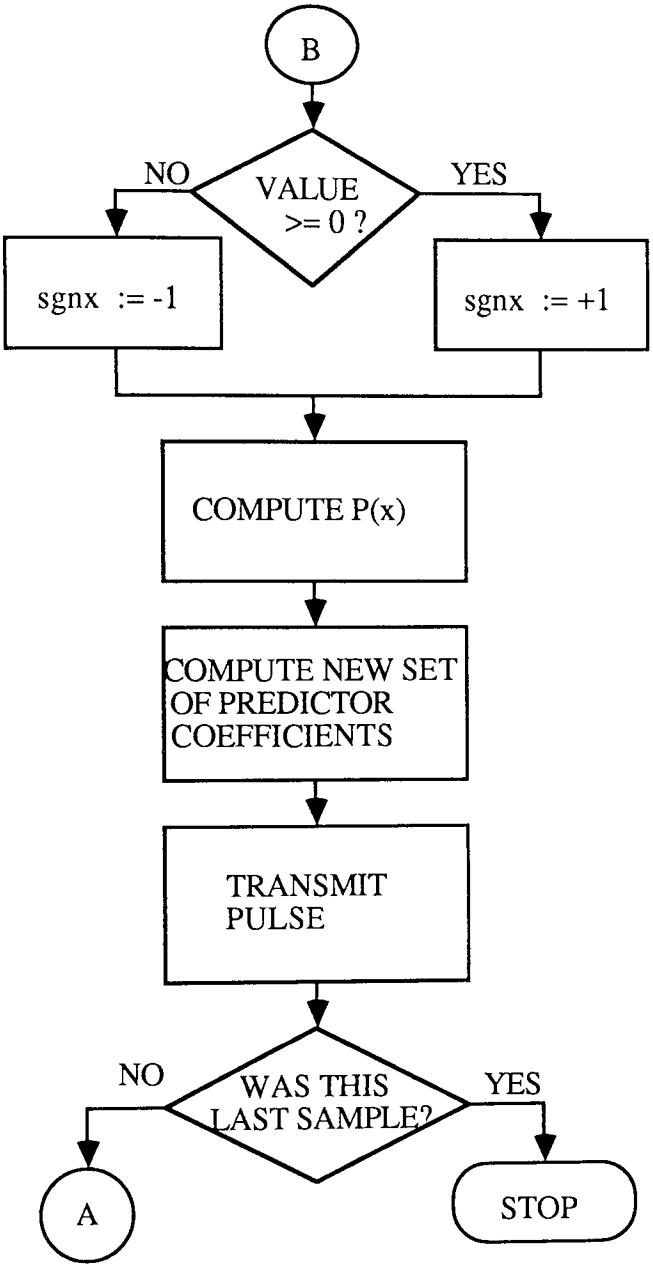
6. Compute new predictor coefficients as in step 11 above

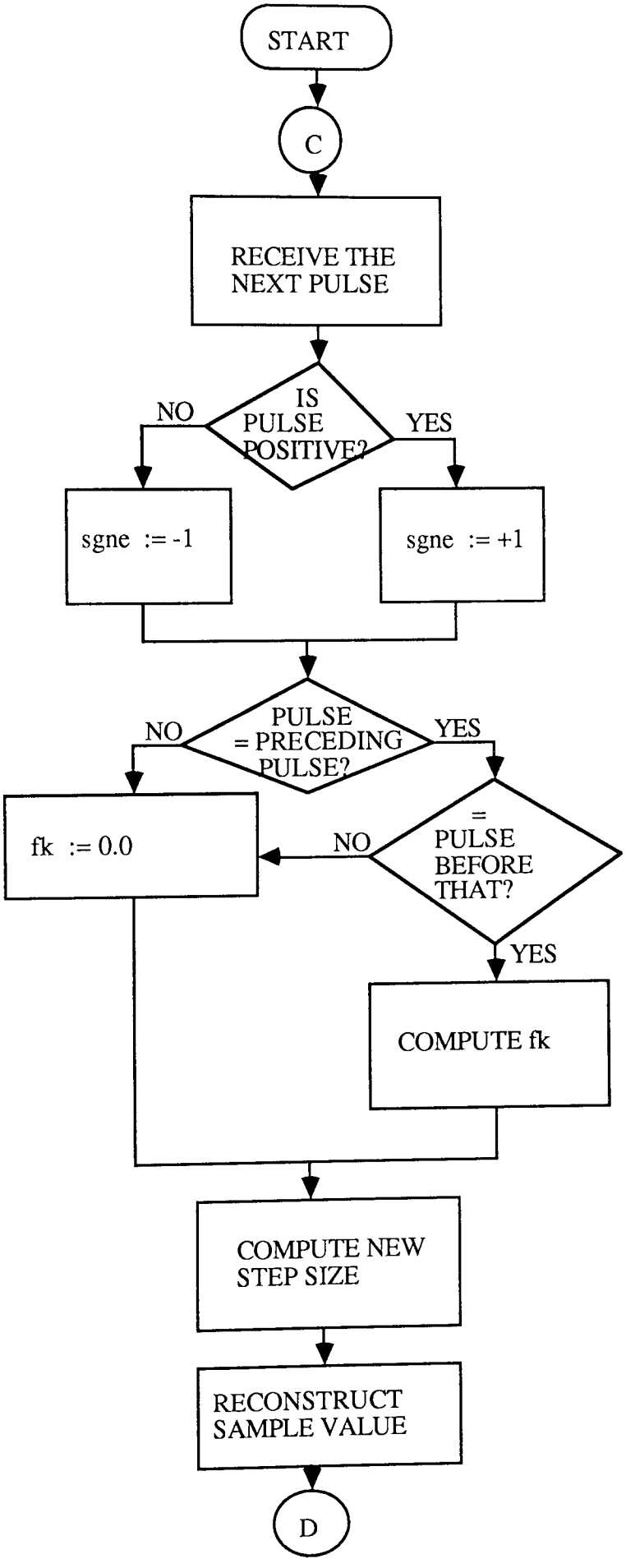
6. Goto step 2

7. Continue until end of file, i.e., until count := 8000

8. Halt


START
DEFINE ALL CONSTANTS AND VARIABLES
INITIALISE VARIABLES & ARRAYS TO ZERO
OPEN THE INPUT & OUTPUT FILES
READ NEXT SAMPLE AND STORE IN MEMO
NO
SAMPLES = 8000 ?
YES
A
PREDICT NEXT SAMPLE
SUBTRACT THIS PREDICTED VALUE FROM THE SAMPLE
NO
IS RESULT POSITIVE ?
YES
GENERATE A NEGATIVE PULSE, P-
GENERATE A POSITIVE PULSE, P+
sgne := -1
sgne := +1
NO
PULSE = PRECEDING PULSE?
YES
fk := 0.0
NO
PULSE BEFORE THAT?
YES
COMPUTE fk
COMPUTE NEW STEP SIZE
RECONSTRUCT TRANSMITTED SAMPLE VALUE
B


B
NO
VALUE
>= 0 ?
YES
sgnx := -1
sgnx := +1
COMPUTE P(x)
COMPUTE NEW SET
OF PREDICTOR
COEFFICIENTS
TRANSMIT
PULSE
NO
WAS THIS
LAST SAMPLE?
YES
A
STOP


START
C
RECEIVE THE NEXT PULSE
IS PULSE POSITIVE?
NO
YES
sgne := -1
sgne := +1
PULSE = PRECEDING PULSE?
NO
YES
fk := 0.0
=
PULSE BEFORE THAT?
NO
YES
COMPUTE fk
COMPUTE NEW STEP SIZE
RECONSTRUCT SAMPLE VALUE
D

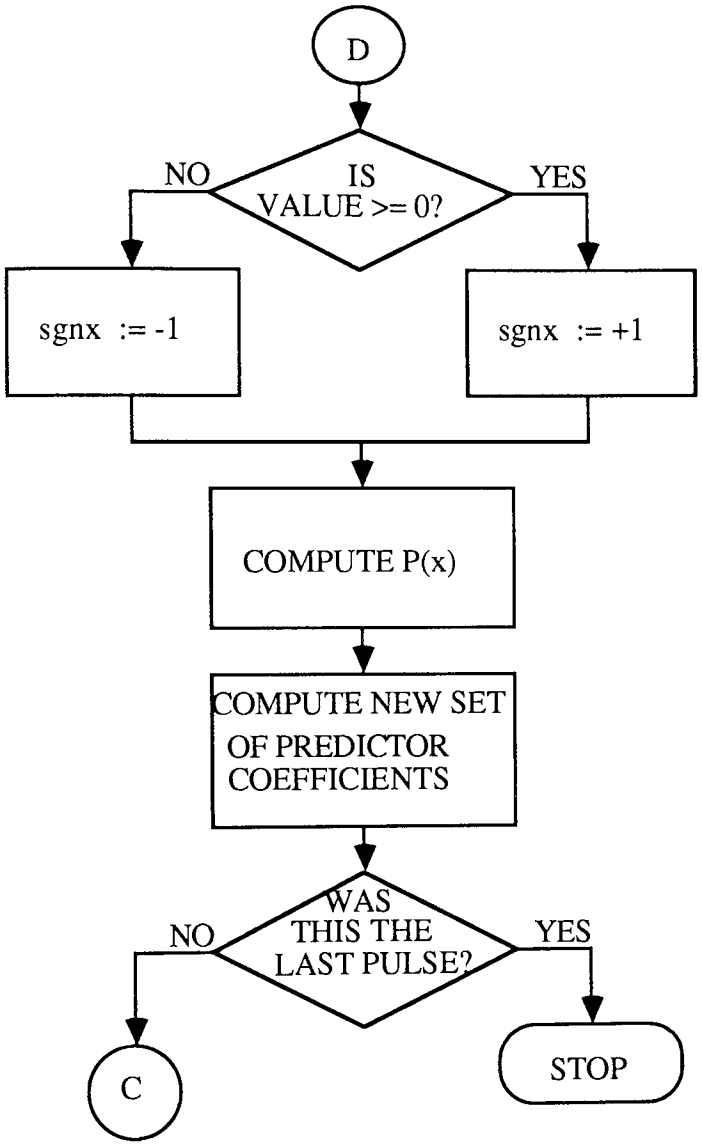



Fig.4.4 A Flowchart representation of the coding process based on the Backward Adaptive algorithm.

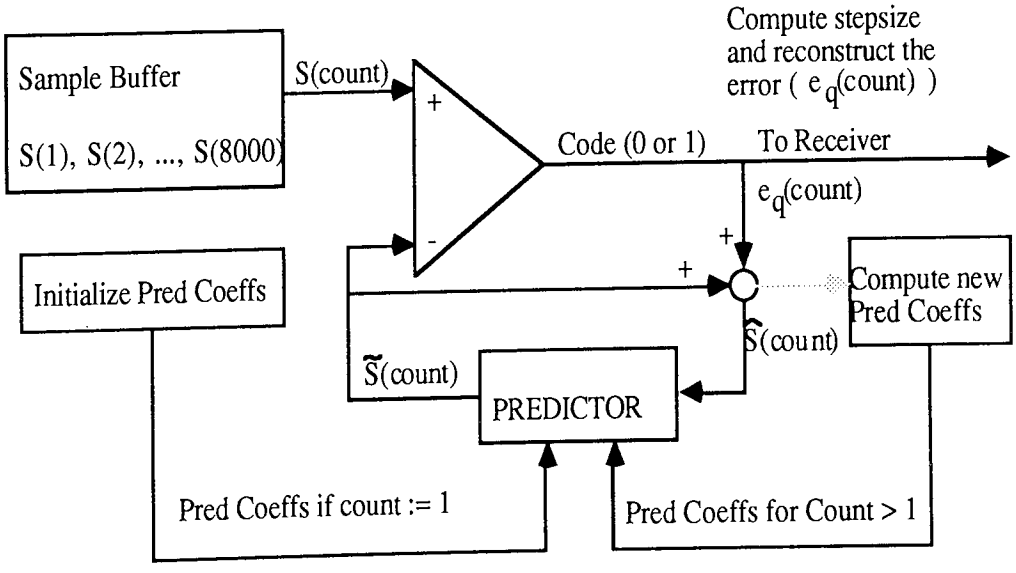



Fig. 4.5 A block diagram representation of the Backward Adaptive coder.

# CHAPTER FIVE

## EXPERIMENT

The experimental work of this project was conducted on an opus PC III IBM clone . The machine had a hard disk drive and a floppy disk drive. It had 640K bytes internal memory capability with provision for expansion to 1M byte. The main processor was a version of Intel's 8088 microprocessor.

The software consisted of programs written in Turbo Pascal version 4.0. This was divided into three basic operational levels: Input, process, and output. Speech was input to the computer via a data acquisition device designed earlier for an undergraduate project in speech recognition. It could accept and process only one word at a time, being originally intended for processing numbers from zero to nine.

The word to be processed was spoken through a telephone handset into the data acquisition device which then sampled and quantised it to about 64kbits/s, and the quantised signal was stored in a RAM memory. A program called "capture" read the word thus processed by the data acquisition device and stored it as a data file on the computer's hard disk. This was carried out for a selection of words such as "hello", "Aston", "good', "morning", "where", etc., and each word stored as a separate data file on the computer's hard disk.

The purpose of the experiment was to process each of the words "captured", as described above, in order to compress the data with the aim of obtaining a bit-rate as near as possible to 8kbits/s. Two adaptive differential pulse code modulation ( ADPCM ) techniques were tested. Differential pulse code modulation was used to take advantage of the correlative nature of speech samples, as was explained in chapter two, and thus to achieve the targeted bit-rate without significant loss in the speech quality. To enable the predictor to cope with the nonstationarity of the speech waveform the predictor coefficients

were made to adapt rather than remain fixed.

The two ADPCM techniques used were forward adaptive and backward adaptive ADPCM. The two techniques adapt the predictor coefficients to match variations in signal parameters with time of the speech waveform, thus enabling accurate prediction of the speech waveform. In both techniques the method used to compute the predictor coefficients is based on minimising the square of the error between a sample and its predicted version. However, the two differ mainly in the way the coefficients are adapted.

The forward adaptive technique adapts the predictor coefficients once over a period of time. This involves windowing, by which a finite number of samples is chosen and the predictor coefficients computed for this block of samples. The autocorrelation technique was used in this experiment, and the coefficients were computed using Durbin's recursive technique as explained in chapter four. Two drawbacks of the forward adaptive technique are a finite time delay which is equal to at least the length of a window, and a higher transmission bit-rate incurred as the result of transmitting the predictor coefficients as side information every time new ones are computed. On the other hand, the backward adaptive technique adapts the coefficients on a sample-by-sample basis so that no delay is involved. Also, no side information need be transmitted as the coefficients can be computed at the receiver using the same technique employed at the transmitter.

However, the coefficients are computed from input samples in the forward adaptive technique rather than from output samples already corrupted by quantization noise as is in the case of the backward adaptive technique. Also, the computed coefficients can be error-protected in the forward adaptive technique. In addition, as was experienced in this project, the backward adaptive algorithms are more complex and slower to execute than the forward adaptive ones.

In each case, the words stored earlier in separate data files were read and processed separately, and each processed word was stored separately in another data file. Another program read the words (processed and unprocessed) and

output them through an I/O device to an external board containing, among other devices, a digital-to-analogue converter and a loud speaker. The speech thus generated was listened to and graded for performance against the its unprocessed version.

## 5.1 EXPERIMENT

The experiment was conducted in two stages. The optimum parameters of each coder were determined in the first stage. The two coders were treated separately for this part of the experiment.

In each case the optimum parameters were determined through a series of listening tests conducted on coded speech using the coder, with its parameters altered each time. The set of parameters with the best performance was the set of optimum parameters for the coder. These were the working parameters of the coder for the next stage of the experiment.

Having determined the working parameters of each coder, the two coders were evaluated against each other with the aim of determining the coder that better achieves the twin-aim of this project: the ability to effectively encode speech at a low bit rate with minimal system complexity. This was the second stage of the experiment.

In both stages of the experiment, subjects registered their subjective opinion according to a five-point grade ( Table 5.1 ). In all, twenty subjects, chosen randomly from the University population, took part in the experiment.

The tests were conducted in Room N401 during periods of minimum noise and interference from other students. A subject listened to an input word stored in a file in the computer and then the coded version of the word, also from a file stored in the computer. The subject made a quality evaluation of the coded word against its input version. The words were drawn from a list of words such as "aston", "hello", "morning", "good", etc. The choice of words was not entirely random. The "s" in "aston", for example is difficult to reproduce because of its being silent.

In the second stage of the experiment, the subject also listened to a play

back of sentences coined from the single words, such as "morning from aston", by coining the words "morning", "from", and "aston", and "hello how are you", from the words "hello", "how", "are", and "you". This was to enable the evaluation of each coder in terms of its usefulness as an information, rather than as word, processor, which is what each coder will be used for in practice.

Table 5.1. The five-point subjective grade scale [63]

| Grade | Quality | impairment |
|---|---|---|
| 5 | Excellent | Imperceptible |
| 4 | Good | Perceptible but not annoying |
| 3 | Fair | Slightly annoying |
| 2 | Poor | Annoying |
| 1 | Bad | Very annoying |

### 5.1.1 Determination of the Optimum Parameters

#### The Forward Adaptive Coder

The parameters that define this coder are the predictor coefficients. These were computed using the autocorrelation technique based on the minimum mean square method ( chapter four ), and the coder updated every so often by computing a new set of coefficients each time. The optimum parameters determined in this experiment are the number of predictor coefficients and the technique of coding the coefficients for transmission. Thus the experiment was done in two stages; in stage one, the optimum number of predictor coefficients are determined, and then the technique of coding the set of coefficients thus determined for transmission to the receiver. In both cases decision is based on

the best subjective score.

Four sets of predictor coefficients of three, four, five, and over five, predictor coefficients were tested for the best subjective performance. The coding techniques considered were linear and log ratio. Both tests were based on the predictor output rather than on the coder output, which involved the delta modulator. This was to avoid complicating issues at this stage by involving the effect of quantization distortion.

## RESULTS

Table 5.2: Subjective Performance Evaluation of the Forward Adaptive Predictive Algorithm with varying number of predictor coefficients.

| No. of Pred. Coeff. / Quality | % Score 3 | % Score 4 | % Score 5 | % Score > 5 |
|---|---|---|---|---|
| Grade 5 | 56.9 | 65.5 | 66.0 | 66.0 |
| Grade 4 | 33.1 | 30.0 | 30.0 | 30.0 |
| Grade 3 | 8.1 | 3.0 | 3.0 | 3.0 |
| Grade 2 | 1.9 | 1.5 | 1.0 | 1.0 |
| Grade 1 | 0.0 | 0.0 | 0.0 | 0.0 |
| Average Grade | 4.45 | 4.60 | 4.61 | 4.61 |

Table 5.3. Subjective Performance Evaluation of Forward Adaptive Predictor Algorithm with predictor coefficients linearly- and log ratio-quantized.

| Coding method / Quality | % Score LINEAR | | % Score LOG RATIO | |
|---|---|---|---|---|
| | 3 BITS | 5 BITS | 3 BITS | 5 BITS |
| Grade 5 | 28.6 | 54.6 | 40.6 | 55.0 |
| Grade 4 | 28.4 | 33.4 | 44.4 | 33.0 |
| Grade 3 | 40.0 | 10.0 | 14.0 | 10.0 |
| Grade 2 | 3.0 | 2.0 | 1.0 | 2.0 |
| Grade 1 | 0.0 | 0.0 | 0.0 | 0.0 |
| Average Grade | 3.82 | 4.41 | 4.25 | 4.41 |

## CONCLUSIONS

1) A predictor with four coefficients performed significantly better than the one with three coefficients, but only slightly less than the one with five and more coefficients ( Table 5.2 ). Due to the need to save on transmission bit rate by minimising the amount of side information to be transmitted, the optimum number of coefficients for the coder was therefore chosen to be four.

2) The predictor performance with the coefficients linearly coded with five bits was better than its performance with the coefficients linearly coded with three bits. The performance of the coder improved slightly with its coefficients first transformed using the log ratio technique [54] and then linearly coded.On average, the predictor performance with the coefficients encoded with five bits was the same in either case ( Table 5.3 ). Because of simplicity, linear coding with five bits was chosen.

### 5.1.2 The Backward Adaptive Coder

The predictor coefficients of this coder are adapted according to the following algorithm ( equation 4.22 ):

$$a_k(i+1) = a_k(i) + P_i(x)e(i)x(i-k) \qquad 5.1$$

where

$$P_i(x) = \frac{A}{B + \frac{1}{M}\sum_{j=i-1-M}^{i-1} x_j^2} \qquad 5.2$$

and A and B are constants.

For a particular coefficient $a_k$, its value at the (i + 1)th instant, $a_k(i + 1)$, is equal to its previous value $a_k(i)$ plus $P_i(x)e(i)x(i - k)$. As shown in equation 5.2, the adaptation rate of the prediction algorithm is controlled by the variation in the mean square value of the immediate past M samples. In particular, when this mean square value increases, $P_i(x)$ is reduced and overcorrections of the $a_k(i+1)$ are avoided, preventing the occurrence of a large prediction error [62].

In this experiment, A and B were kept constant at 10.0 and 100.0 respectively, while M was varied from 40.0 to 100.0 in steps of 20. For each of these values of M, a number of words were coded and subjectively evaluated. The aim was to determine the value of M with the best subjective result to be the optimum M for the rest of the experimental work of this project. As in the case of the Forward Adaptive Coder, this part of the experiment did not involve the delta modulation part. Coefficient updating was done according to equation 5.1, involving prediction error and past input sample values. The aim again, as in the case of the Forward Adaptive Coder, was to determine the efficiency of the predictive algorithm and to later determine the effect of quantization noise on the algorithm of the coder. For the purpose of comparison, four coefficients were adopted for this part of the experiment, the same as for the Forward Adaptive Coder.

**RESULTS**

Table 5.4. Subjective Performance Evaluation of Backward Adaptive Predictor Algorithm with M, the number of immediate past samples, varied.

| M / Quality | % Score M = 40 | % Score M = 60 | % Score M = 80 | % Score M = 100 |
|---|---|---|---|---|
| Grade 5 | 30.6 | 36.4 | 53.2 | 55.4 |
| Grade 4 | 33.4 | 32.6 | 33.6 | 33.0 |
| Grade 3 | 28.0 | 25.0 | 10.0 | 10.6 |
| Grade 2 | 7.4 | 5.5 | 3.0 | 1.0 |
| Grade 1 | 0.6 | 0.5 | 0.2 | 0.0 |
| Average Grade | 3.86 | 3.98 | 4.36 | 4.42 |

## CONCLUSIONS

The performance grading of the algorithm for M = 100 is slightly better than that for M = 80, which is better than that for M = 60, and so on. It obviously shows sign of becoming better as the value of M increases. However, for computational reasons, the optimum value for M was chosen as 100.

### 5.1.2 Performance Evaluation of the two Coders

Having determined the optimum parameters of each coder in the first stage of this experiment, speech signals consisting of single words were input to each coder and subjective tests carried out on their outputs for subjective performance evaluation with the aim of determining the better of the two coders based on their overall performance and also, to determine the effect of quantization noise on the performance of each coder. In pursuit of this latter

aim, sentences formed by combining output words from the coder were also tested in this part of the experiment. This was to determine the usefulness of each coder as an information processing device.

## RESULTS

Table 5.5. Subjective Performance Evaluation of the two Coders

| CODER / QUALITY | FORWARD ADAPTIVE | | BACKWARD ADAPTIVE | |
|---|---|---|---|---|
| | WORDS | SENTENCES | WORDS | SENTENCES |
| GRADE 5 | 27.8 | 27.5 | 27.6 | 27.5 |
| GRADE 4 | 28.2 | 28.0 | 27.8 | 28.0 |
| GRADE 3 | 42.0 | 40.5 | 41.8 | 38.0 |
| GRADE 2 | 2.0 | 2.0 | 2.0 | 4.5 |
| GRADE 1 | 0.0 | 2.0 | 0.8 | 2.0 |
| AVE. GRADE | 3.82 | 3.77 | 3.79 | 3.75 |

## 5.2 CONCLUSIONS

I The output speech quality of both coders was rated, on average, between good and fair. At these grades, the impairments, mainly due to quantization distortion, are just perceptible and acceptable ( refer to Tables 5.1 and 5.5 ).

II The impairments in both coders were slightly more perceptible in sentences formed by coining words individually coded by the coder than in the coded words when listened to them singly.

III On average, both coders's performances were quite good, with the Forward Adaptive Coder's performance sightly better than that of the Backward Adaptive Coder in both the words and sentences ( refer Table 5.5 ).

IV The speech quality of the Forward Adaptive Coder was sightly degraded when the predictor parameters were quantised, ( refer to chapter four for transmission parameters ) and further when delta modulation was included ( compare Tables 5.2, 5.3, and 5.5). On average, the effect of quantising the predictor parameters on the coder performance was a drop in speech quality from an average grade of 4.6 to 4.41, a drop in speech quality of 4 per cent. This is quite slight considering the fact that the speech quality is still between excellent and good. Thus, one can safely conclude that the effect of quantising the predictor parameters on the quality of the coder output was negligible and that linear quantization was quite adequate for this experiment.

V The effect of quantizer distortion on the coder output was considerable. This can be seen in the drop of speech quality from 4.41, with parameters quantised but without delta modulation, to 3.82 ( for the same words ), including delta modulation, a drop in speech quality of 13 per cent ( compare tables 5.3 and 5.5). Also, it can be seen from comparing the results in the two tables that the impairment is more perceptible with delta modulation included than without delta modulation.This discrepancy between the performance of the algorithm and the coder can be explained by the binary nature of delta modulation, which is part of the function of the coder. Also, delta modulation in this project is done on speech signals that are originally sampled only slightly above the Nyquist rate rather than the far-in-excess-of Nyquist rate sampling requirement for conventional delta modulation ( refer to chapter two for full discussion on delta modulation ).

VI The Backward Adaptive Coder suffered a drop of 15 per cent, a drop in speech quality from 4.42 without delta modulation to 3.75 when delta

modulation is involved ( compare Tables 5.4 and 5.5 ). This is 8 per cent more than the overall effect of quantization noise on the Forward Adaptive Coder. This is as expected as the predictor coefficients were now updated based on information about past quantised sample values rather than the past input sample values of the first result. Thus, not only the signal value, but the prediction coefficients were also affected by the coarse quantization of the delta modulator. The sgn(error)-sgn(data) algorithm given in equation 4.24 was used in this part of the experiment. However, it achieved quite acceptable performance at the minimum bit rate ( 8kbps ) possible in this experiment. The overall transmission bit rate for the Forward Adaptive Coder was 9.6kbps ( 8kbps data plus side information of 20 bits every 100 sampling times = 8kbps + 1.6kbps ). Thus, as a low bit rate coder, the Backward Adaptive Coder is a better choice. However, it is more complex than the Forward Adaptive.

VII Another shortcoming of the Backward Adaptive Coder observed in this experiment, though not apparent in the results of Table 5.5, was that its processing time was longer by far than that of the Forward Adaptive Coder. If processing time was used as a criterion in determining impairment, the overall performance of this coder would have been further affected. It was not considered as a performance criterion in this part of the experiment because it is the author's belief that the problem is more associated with processor power than with the overall coder performance. With a more powerful processor it will cease to be of any significance.

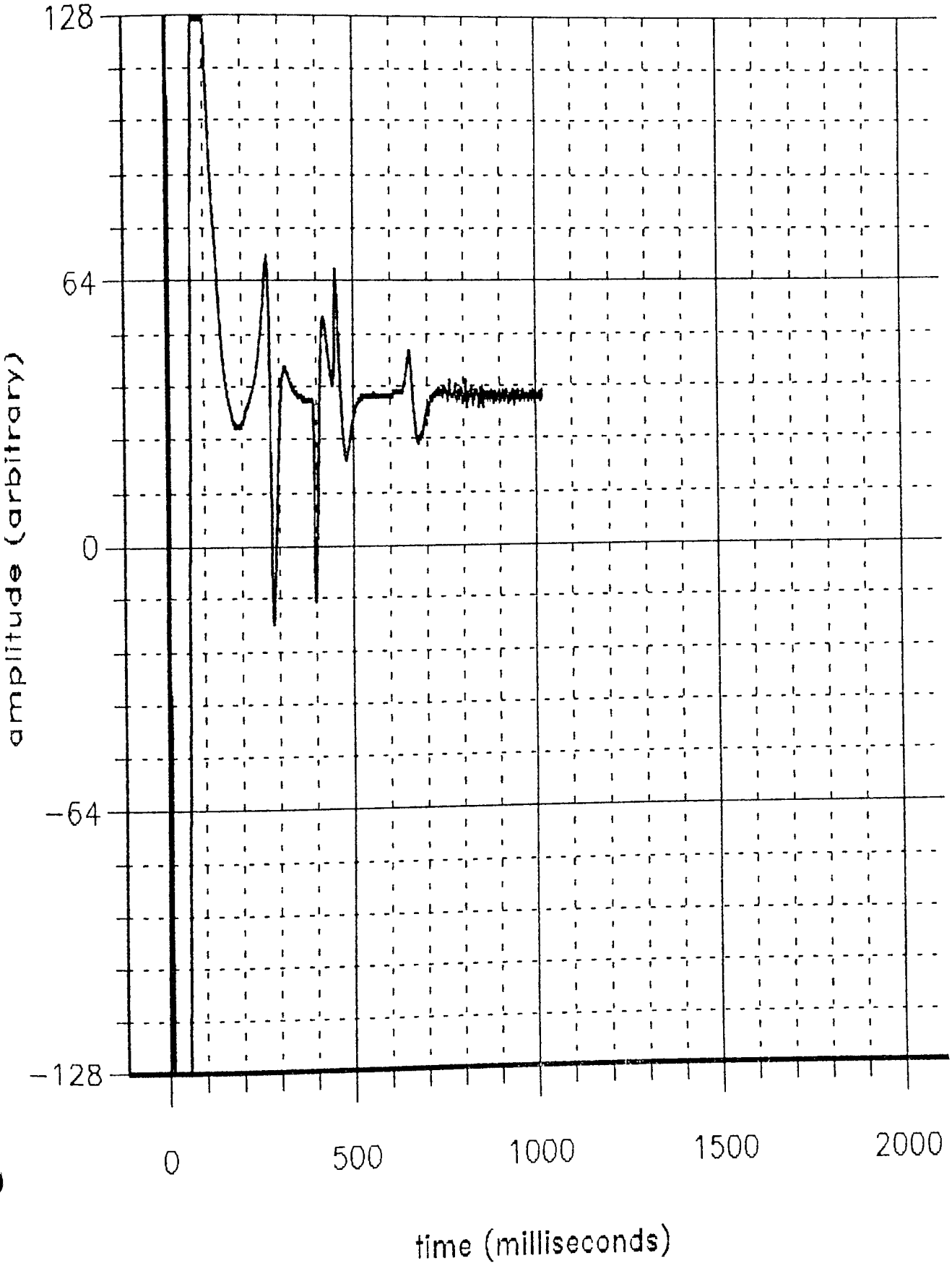
As part of the last part of this experiment, waveforms of the output speech signals from each coder were studied and compared with their corresponding input versions. They were obtained with the aid of the Logistix Graphic Display Utility. This is a spread sheet with graphic capability. It plots the graph of the data presented to it. A program called graphpre, written in turbo pascal, was used in preparing the files to be plotted by the graphic utility. The program processes 8000 samples of a given file (input and processed) into a graph of amplitude versus time and the graph then displayed using the graphic utility as

explained above. Hard copies of the graphs were studied. The aim here was to determine how well each coder could reproduce the input waveform and to investigate any peculiarity in each coders output by studying the speech spectrum.

Samples of these graphical displays are given in Fig.5.1. They may not give much information as to the speech quality, but are an illustration of the coders' ability to reproduce the input speech waveform fairly well, which is a significant achievement for a low bit rate coder. The sample waveforms are those of the input word "aston" and the forward and the backward coder output versions. Both coder outputs include quantization ( the effect of delta modulation ).

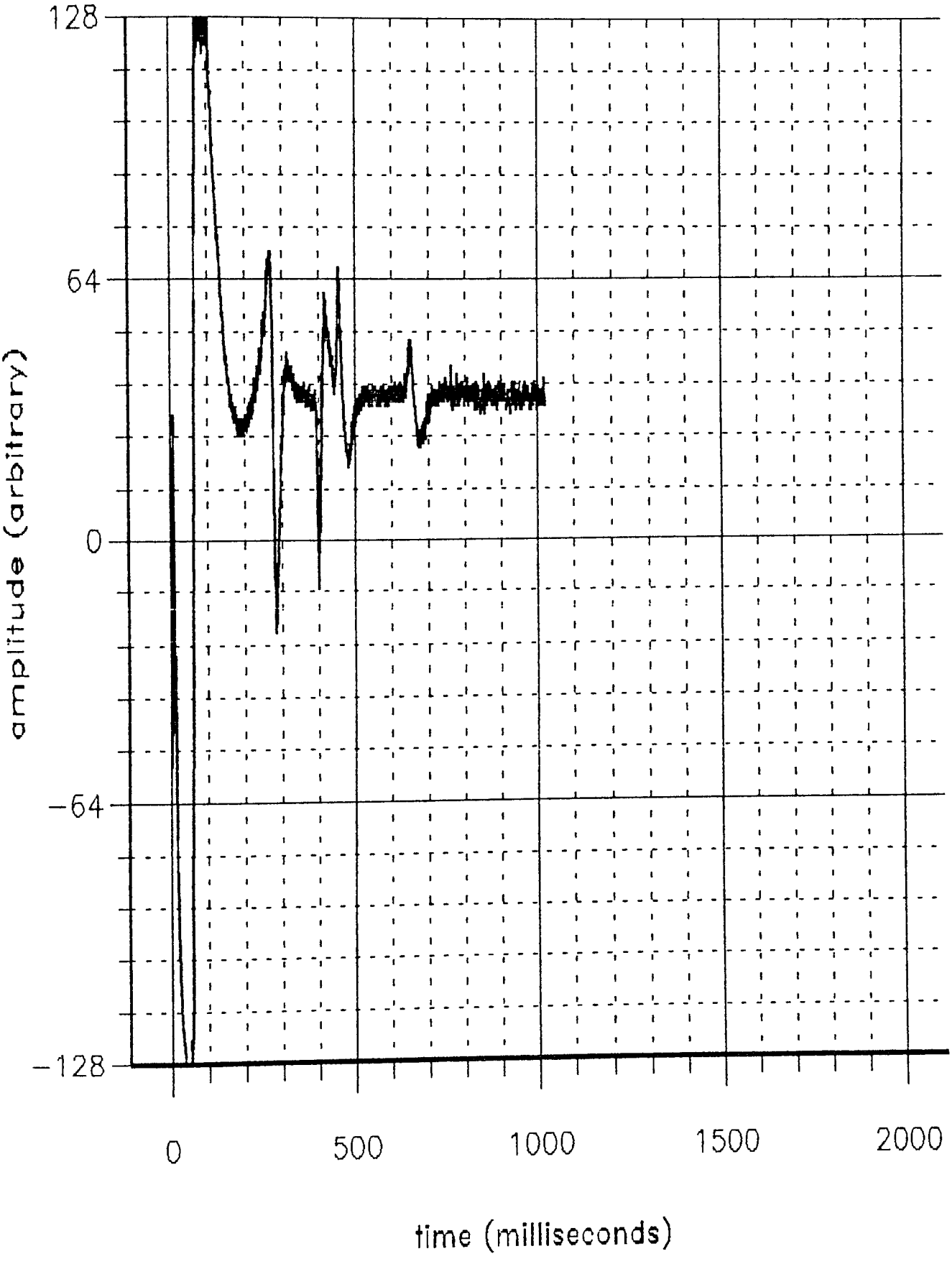
Fig. 5.1 Input and Output waveforms of the word "aston"

# (i) Input word
## Aston



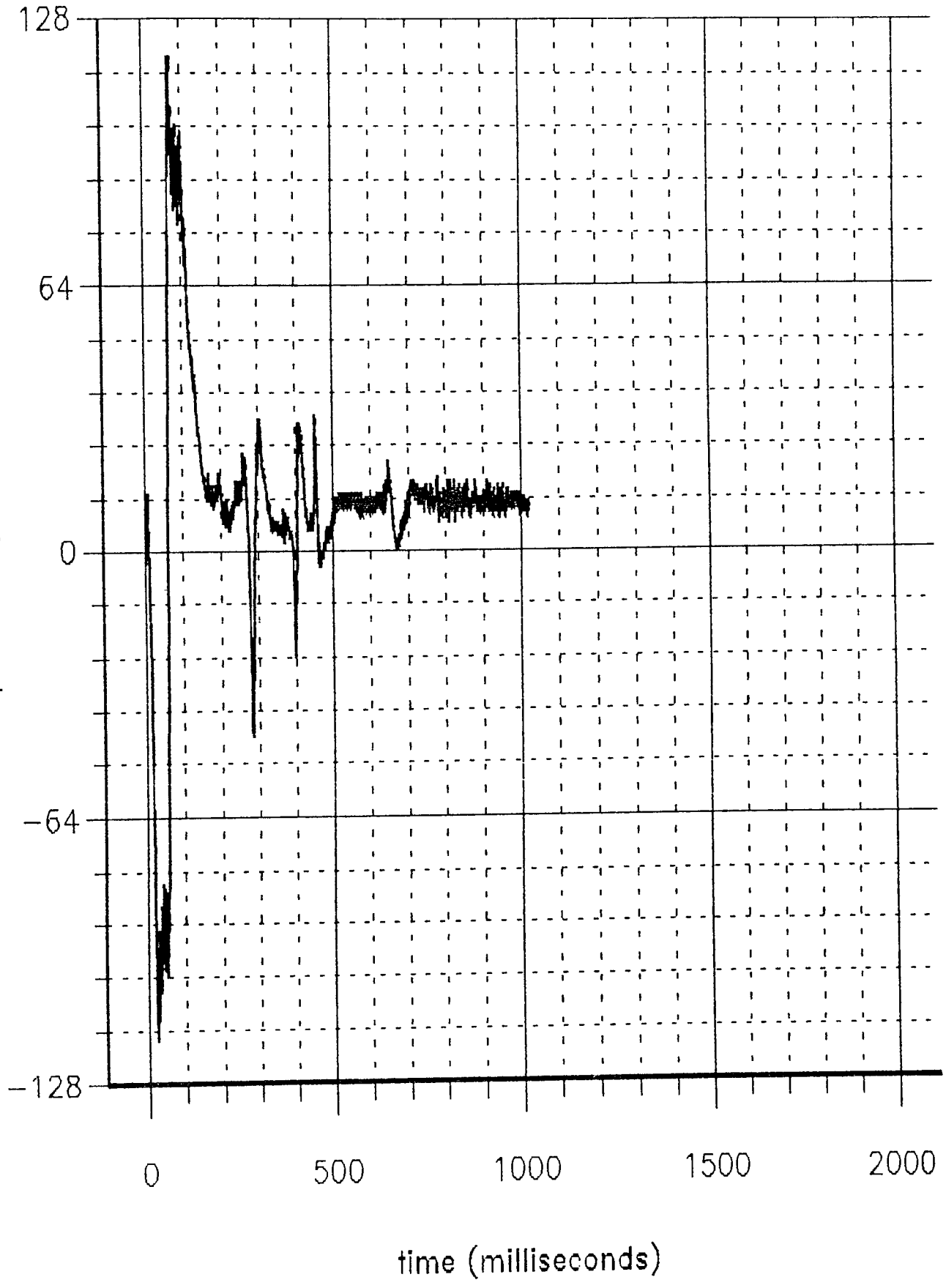
# (d) Forward Coder output with quantization

## Aston



# (e) Backward Coder output

## Aston



# CHAPTER SIX

# CONCLUSION

## 6.1 DISCUSSION

The aim of this project was to code speech at about 8kbps for communications purposes. To achieve this aim two coders, one based on the forward adaptive predictive algorithm and the other based on the backward adaptive predictive algorithm, have been designed and tested. The two algorithms and coder techniques are discussed in chapter four, and the performances of the two coders are discussed in chapter five.

The coders combine delta modulation with linear predictive coding. This linear predictor-delta modulation combination ensures that the two-fold requirement of the coders is achieved. These are simplicity and efficiency. The simplicity requirement is met by the use of delta modulation in the coders. Delta modulation is the simplest form of quantization (two-level quantizer). Another good quality of delta modulation beneficial to this project is robustness to transmission errors. This latter property of delta modulation thus eliminates the need for any coding to protect against transmission errors, thus reducing the amount of side information required for transmission. Thus delta modulation also enhances the low bit-rate target. The use of a linear predictor in place of the normal integrator used in conventional coders using delta modulation enhances the efficiency requirement.

Linear predictive coding is a major contender for low bit-rate digital speech coding. This technique is discussed in detail in chapter two. However, the technique on its own produces speech of vocoder quality and is not suitable for the type of speech quality aimed for in this project. Complexity is another drawback of linear predictive coding. On the other hand, conventional delta modulation is a waveform coding technique that yields speech of telephone

quality but with a much higher transmission bit-rate requirement than is aimed for in this project.

Thus neither of the techniques on its own prove suitable for this project. However, by combining the two, the advantages of both techniques have been made use of. In addition, both the predictor and the quantizer (delta modulator) in both coders are made adaptive. This further enhances the performance of the two coders. In this way, both coders have the ability to closely follow the waveform of the speech signal.

To provide a common basis for comparison, the same method of adapting the quantizer step size is used for both coders. However, the two coders differ in their method of adapting the predictor coefficients. The forward adaptive coder updates the predictor coefficients at the transmitter input, based on input speech samples. The backward adaptive coder, on the other hand, updates the predictor coefficients at the transmitter output end, based on the output reconstructed sample values. Moreover, forward adaptation of predictor coefficients is done over a period of time ( every 25msec in this project ) and requires that the predictor coefficients be transmitted to the receiver, thus adding to the transmission bit-rate. On the other hand, backward adaptation is done on a sample-by-sample basis. Since new coefficients are computed based on the data at the output, this coder does not require the transmission of the predictor coefficients to the receiver as they can easily be computed there using the same technique as at the transmitter output.

The performances of the two coders were discussed in chapter five. It is shown that the forward adaptive coder performs slightly better than the backward adaptive coder ( refer to Table 5.5 for subjective performance evaluation results for the two coders ). Also, although not apparent in the evaluation results, the forward adaptive coder was faster than its backward counterpart. One reason for this is the fact that the forward coding technique updates the predictor coefficients periodically while the backward algorithm updates them every sampling period. This characteristic of the backward coding algorithm makes it more computation intensive than its forward

counterpart. Also, these computations tend to be more complex the more efficient the technique is. In this project an approximate technique based on the the least mean square method was used. This is far simpler and faster than the more complex but more accurate recursive least squares method.

However, the backward adaptive coder is a better low bit-rate coder than the forward adaptive due to the fact that it does not involve the transmission of side information. In this project it achieved 8kbps for the speech quality aimed at in this project as compared to 9.6kbps for the same quality achieved by the forward adaptive counterpart. It should be noted that 8kbps was the minimum bit rate that could be achieved in this project as speech signals were sampled at 8kHz. Thus it is possible that the two coders could have performed the same if the bit-rate was the same in both coders. This was not tried in this project as it would mean redesigning the data acquisition device.

The quantizer step size in both coders was adapted every sampling period based on information of the data at the output. This helped to reduce the amount of side information in the forward adaptive coder and kept that for the backward adaptive coder at zero. The step size at the receiver was determined using the same technique as at the transmitter. The computational requirement on both coders as the result of this backward update of the step size was kept to a minimum due to the binary nature of delta modulation.

The performance of the two coders was generally graded as quite good, with the impairments, mainly due to quantization distortion, just perceptible but acceptable. In general, the quality of words coded by both coders was better than that of sentences formed by coining these individually coded words, where the impairments were slightly more perceptible.

The effect of quantization on the Forward Adaptive Coder was a drop in speech quality of 13 per cent. This included a 4 per cent drop due to the effect of quantising the predictor parameters and a further drop of 9 per cent in speech quality due to the effect of delta modulation ( compare Tables 5.2, 5.3, and 5.5 ). The Backward Adaptive Coder suffered a 15 per cent reduction in speech quality ( compare Tables 5.4 and 5.5 ), 2 per cent more than the overall

reduction for the Forward Adaptive Coder. This is as expected, as the predictor coefficients for this coder were updated using past quantised sample values.

In this project the speech signals were originally sampled at a rate only slightly above the Nyquist rate ( 8kHz ). Thus, the fact that the signals were successfully coded using techniques involving delta modulation, which normally requires a sampling rate far in excess of the Nyquist rate for distortionless reconstruction of the speech signal, is an achievement in itself. Samples of the speech waveforms obtained during the experiment are displayed in Fig.5.1. Although these waveforms, in themselves, may not give an indication of the quality of the coded speech signals, they are significant in that they illustrate how well each coder was able to reproduce the input speech waveform. This ability to reproduce the input speech waveform is a characteristic of waveform coders which produce high quality speech ( refer to chapter two for a discussion of the types of speech coding techniques ). Thus the quality of the output speech signal is closely associated with its envelope. In particular, the extent of distortion in the output speech signal can be determined from comparing its waveform with that of the input signal. The sample waveforms in Fig.5.1 show that both coders were able to reproduce the input waveforms quite well, further confirming the results of the subjective tests.

Thus the two aims of this project, the coding of speech at low bit-rates with speech quality suitable for communication purposes and with as minimum as possible system complexity, have been achieved. This achievement was, to a large extent, due to the purely-software design adopted in this project. The result of this was flexibility in the implementation of the coders. Parameters could be altered with very little design consideration. The software nature of the coders also meant that the performance of the coders depended, to a large extent, on the coding algorithm used and on the computer's processing power.

It has been demonstrated in this project that the two coding techniques adopted in this project are fairly robust to quantization noise, even with the coarse quantization used in this project. Both suffered only slight degradations in their output speech quality, mainly as the result of delta modulation.

## 6.2 RECOMMENDATIONS FOR FUTURE WORK

This project has been about speech coding. Thus, the results discussed above were obtained assuming ideal channel conditions, the only impairment been due to quantization distortion and the effect on each coder's algorithm of changes in its parameters. Thus a future area of research could be in determining the effect of other impairments due to transmission and channel errors. As the final results are intended for mobile communication purposes, this future work would involve mobile channel and terminal simulation, taking various environmental conditions into consideration. Any of the two algorithms developed in this project could be used for the future experiment. Alternatively, both algorithms could be incorporated to determine which of the two coders developed in this project is better suited for the transmission and channel requirement. So far, in this project, neither of the two coders had a clear edge over the other. What the Backward Adaptive Coder lacked in speed and output speech quality in comparison to its Forward Adaptive counterpart, it gained in transmission bit-rate saving.

The experimental work of this project was conducted within some constraints; severely limited computer memory, the absence of a mathematics co-processor to handle floating point operations more efficiently, and the rather low processing power of the computer ( an 8088 processor operating at 4.7kHz ) compared to faster processors common in more modern personal computers. In future, a more powerful computer with at least 1Mbyte of internal memory and with a mathematics co-processor fitted should be considered as the minimum requirement.

Also, it would help to design a data acquisition device so that the computer could access it as one of its peripheral devices without the need for a separate

input program. This would enable words to be input continuously, thus enabling the coding of sentences rather, rather than in isolation, as was only possible in this project.

## REFERENCES


[1] HOYLE, R.D., "Comparison of Digital Speech Coding Methods for Mobile Radio System", IEEE Journal on Selected Areas in Communications, Vol. SAC-5 No.5, June 1987, pp915-920.

[2] BASTIAN, M., "Voice-Data Integration: An Archetecture Perspective", IEEE Communications magazine, Vol.24, No.7, July 1986, pp8-12.

[3] IBE, O.C. and GIBSON, D.T., "Protocols for Integrated Voice and Data Local Area Networks", IEEE Communications magazine, Vol.24, No.7, July 1986, pp30-36.

[4] BELLAMY, John C. "Digital Telephony", John Wiley & Sons, 1982.

[5] PAPAMICHALIS, Panos E. "Practical Approaches To Speech Coding", Prentice-Hall, Inc., Englewood Cliffs, New Jersey, 1987.

[6] ABATE, J.E. "Linear and Adaptive Delta Modulation", Proceedings of the IEEE, Vol.55, No.3. March 1967, pp298-307.

[7] JAYANT, N.S. "Adaptive Delta Modulation with a One-Bit Memory", The Bell System Technical Journal, Vol.49, Number3, March, 1970, pp321-342.

[8] GIBSON, J.D. "Adaptive Prediction in Speech Differential Encoding Systems", Proceedings of the IEEE, Vol.68, No.4, April, 1980, pp488-525.

[9] GIBSON, J.D. "Adaptive Prediction for Speech Encoding", IEEE Acoustics, Speech, and Signal Processing Magazine, July, 1984, pp12-26.

[10] MILLAR, P.C., "A reduced delay 8-band sub-band coder, Br. Telecom. Technol. J. Vol.3, No.3, July 1985, pp56-61.

[11] FLANAGAN, J.L. et al. "Speech Coding", IEEE Transactions on Communications, Vol.COM-27, No.4, April, 1979, pp710-737.

[12] JAYANT, N.S. "Step-Size Transmitting Differential Coders for Mobile Telephony", The Systems Technical Journal, Vol.54, No.9,November, 1975, pp1557-1581.

[13] ATAL, B.S. and SCHROEDER, M.R. "Adaptive Predictive Coding of Speech Signals", The Bell System Technical Journal, October,

1970,pp1973-1986.

[14] ATAL, B.S and SCHROEDER, M.R. "Predictive Coding of Speech Signals and Subjective Error Criteria", IEEE Transactions on Acoustics, Speech, and Signal Processing, Vol.ASSP-27. No.3, June,1979, pp247-254.

[15] ATAL, B.S. "Predictive Coding of Speech at Low Bit Rates",IEEE Transactions on Communications, Vol.COM-30, No.4, April, 1982, pp600-614.

[16] ALEXANDER,S.T. "Adaptive Signal Processing _Theory and Applications", Springere-Verlag, Newyork Inc. 1986.

[17] HUANG, Y.F., "A recursive estimation algorithm using selective updating for spectral analysis and adaptive signal prpcessing", IEEE Transactions on Acoustics, Speech, and Signal Processing, Vol. ASSP-34, pp1331-1334, 1986.

[18] TRUSSEL, H.J., and COMBETTS, P.L., "Stability of the linear prediction filter: A set theoretic approach", Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing, New York, Vol.4, pp2288-2291, 1988.

[19] MAKHOUL, J. "Linear Prediction: A tutorial review", Proceedings of the IEEE, Vol. 63, pp561-580, 1975.

[20] MIYANAGA, Y., MIKI, N., NAGAI, N., and HATORI, K., "A speech analysis algorithm which eliminates influence of pitch using the model reference adaptive system", IEEE Transactions on Acoustics, Speech, and Signal Processing, Vol. ASSP-30, pp88-96, February 1982.

[21] ATAL, B.S, and HANAUER, S.L, "Speech analysis and synthesis by linear prediction of the speech wave", J. Acoust Soc. Amer., Vol.50 pp637-655, August 1971.

[22] ATAL, B.S., and SCHROEDER, M.R., "Linear prediction analysis of speech based on a pole-zero representation", J. Acoust., Soc. Amer., Vol.64 pp1310-1318, November 1978.

[23] ROSE, Rchard C. and BARNWELL, Thomas P., "Design and

performance of an Analysis-by-Synthesis Class of Predictive Speech Coders", IEEE Transactions on Acoustics, Speech, amd Signal Processing, Vol.38, No.9, September 1990.

[24] SCHROEDER, M.R., and ATAL, B.S., "Coder excited linear prediction: High quality speech at very low bit rates", Proc. International Conference on Acoustics, Speech, and Signal Processing, April 1985, pp937-940.

[25] MAKHOUL, J., and BEROUTI, M., "Predictive and residual coding of speech", J. Acoust. Soc. Amer. Vol.66 No.6 December 1979.

[26] KROON, P., DEPRETERE, E.F., and SLUYTER, R.J., "Regular pulse excitation: A novel approach to effective and efficient multi-pulse coding of speech", IEEE Transactions on Acoustics, Speech, and Signal Processing, Vol. ASSP-34, No.5 pp1054-1063, October 1986.

[27] DAVIDSON, G. and GERSHO, A., "Complexity reduction method for vector excitation coding", Proc. IEEE International Conference on Acoustics, Speech, and Signal Processing Tokyo, Japan, 1986, pp3055-3058.

[28] SINGHAL, S. and ATAL, B.S., "Improving performance of multi-pulse LPC coders at low bit rates", Proc. IEEE International Conference on Acoustics, Speech, and Signal Processing, San Diego, CA, 1984, paper 1.3.

[29] GRAY, R.M., "Vector quantization", IEEE Acoustics, Speech, and Signal Processing magazine, Vol.1, No. pp4-29, April 1984.

[30] CUPERMAN, V. and GERSH, A., "Vector Predictive Coding of Speech at 16kb/s", IEEE Transactions on Communications, Vol. Com-33, No.7, July 1985, pp685-695.

[31] CHANG, Pao-Chi and GRAY, R.M., "Gradient Algorithms for Designing Predictive Vector Quantizers", IEEE Transactions on Acoustics, Speech, and Signal Processing, Vol. ASSP-34, No.4, August 1986,pp679-689.

[32] SAVVIDES, V. and XYDEAS, C.S., "A New Approach to Low Bit Rate Speech Coding", 5th International Conference on Digital Processing of

Signals in Communication, IERE Univeristy of Technology, Loughborough, United Kingdom, Sept. 1988.

[33] BOYD, I., SOUTHCOTT, C.B., and CROWE, D.P., "Implementation of a 9.6 kbps Multipulse-excited LPC Speech Codec For an Aeronautical Telephone Service", same conf. as in [8] above.

[34] ATAL, B.S., and REMEDE, Joel R., "A new model of LPC Excitation for producing Natural-Sounding Speech at Low Bit Rates", IEEE International Conference on Acoustics, Speech, and Signal Processing, April 1982 pp614-617.

[35] JAIN, V.K., and HANGARTNER, R., "Efficient Algorithm for Multipulse LPC Analysis of Speech", IEEE International Conference on Acoustics, Speech, and Signal Processing, May 1984, pp1.4.1-1.4.3.

[36] TRANSCOSO, I.M., GARCIA-COMEZ, R., and TRIBOLET J.M., "A study on Short-Time phase and Multipulse LPC", same conference as [11] above, pp10.3.1-10.3.4

[37] SHARAKI, Yoshinao and HONDA Masaaki, "LPC Speech Coding Based on Variable-Length Segment Quantization", IEEE Transactions on Acoustics, Speech, and Signal Processing, Vol.36, No.9, September 1988, pp1437-1444.

[38] OZAWA, Kazunori and ARASEKI, Takashi, "High Quality Multipulse Speech Coder With Pitch Prediction", IEEE International Conference on Acoustics, Speech, and Signal Processing, Tokyo, April 1986, pp1689-1692.

[39] GIGSON, Jerry, D., "Sequential Adaptive Backward Prediction in ADPCM Speech Coders", IEEE Transactions on Communications Vol-26 No.1 January 1978.

[40] TRANCOSO, I.M., and ATAL, B.S., "Efficient Search Procedures for Selecting the Optimum Innovation in Stochastic Coders", IEEE Transactions on Acoustics, Speech, and Signal Processing, Vol.38, No.3 March 1990, pp385-395.

[41] LIND, L.F., ATTKINS, P.M., and CHALLENER, P., "An improved implementation of adaptive quantizers for speech waveform encoding

schemes", Br. Telecom Technol. J. Vol.6 No.2, April 1988, pp41-48.

[42] KROON, Peter and DEPRETTERE, Ed F., "Experimental Evaluation of Different Approaches to Multi-Pulse Coder", IEEE International Conference on Acoustics, Speech, and Signal Processing, May 1984, pp10.4.1-10.4.4.

[43] HUANG, Z., YANG, X., ZHU, X., and KUH, A., "Homomorphic linear predictive coding: a new estimation algorithm for all-pole speech modelling", IEE Proceedings Vol.137, pt.1 No.2, April 1990.

[44] CIOFFI, John M. and KAILATH, Thomas, "Fast, Recursive-Least-Squaresn Transversal Filters for Adaptive Filtering", IEEE Transactions on Acoustics, Speech, and Signal Processing, Vol.ASSP-32 No.2, April 1984.

[45] AYANOGLU, Ender and GITLIN, Richard D., "Tandem Transcoding without Distortion Accumulation for Memoryless and Predictive Vector Quantizers", * pp0295-0300.

[46] COSTANTINI, C., and PARLADORI, G., "A simplified method for RPE Coding with Improved Performance, * pp0280-0284

[47] YONG, Mei and GERSHO, Allen, "Vector Excitation Coding with Dynamic Bit Allocation", * pp0290-0294

[48] LIU, Y.J., "On Tree Codebook Design for Vector Quantization of Speech", * pp0586-0589.

[49] WATTS, Lloyd and COPERMAN, Vladimir, "A Vector ADPCM Analysis-By-Synthesis Configuration For 16 kbit/s Speech Coding", * pp0275-0279.

[50] TZENG, Forrest F., "Multipulse Excitation Codebook Design and Fast Search Methods for CELP Speech Coding", * pp0590-0594.

[51] MALONE, Kevin T. and FISCHER, Thomas R., "Trellis-Searched Adaptive Predictive Coding", * pp0566-0570.

[52] GIBSON, Jerry D. "Principles of Digital and Analog Communications", Macmillan Publishing Company, New York, 1989.

[53] GRAY, Jr. A.H. and MARKEL, J.D. "Quantization and Bit Allocation in Speech Processing", IEEE Transactions on Acoustics, Speech, and Signal Processing, Vol.ASSP-24, No.6, December, 1976, pp459-473.

[54] VISWANATHAN, R. and MAKHOUL, J. "Quantization Properties of Transmission Parameter in Linear Predictive Systems" IEEE Transactions on Acoustics, Speech, and Signal Processing, Vol.ASSP-23, No.3, June, 1975, pp309-321.

[55] GOODMAN, David J. "Delta Modulation Granular Quantization Noise", The Bell System Technical Journal, May-June, 1969, pp1197-1218.

[56] GOODMAN, D.J. and GERSHO, A. "Theory of an Adaptive Quantizer", IEEE Transactions on Communications, Vol. COM-22, No.8, August, 1974, pp1037-1045.

[57] MARKEL, J.D. and GRAY, Jr., A.H. "A Linear Predictiction Vocoder Simulation Based Upon the Autocorrelation Method", IEEE Transactions on Acoustics, Speech, and Signal Processing, Vol.ASSP-22, No.2, April, 1974k, pp124-134.

[58] MORGAN, D.R. and CRAIG, S.E. "Real-Time Adaptive Linear Prediction Using the Least Mean Square Gradient Alagoariathm", IEEE Transactions on Acoustics, Speech, and Signal Processing, Vol.ASSP-24, No.6, December, 1976, pp494-507.

[59] TREICHLER, J.R., JOHNSON R.C. Jr., and LARIMORE, M.G. "Theory and Design of Adaptive Filters", John Wiley & Sons, Inc., 1987.

[60] SINGH, A. and TRIEBEL, W.A., "The 8088 Microprocessor: Programming, Interfacing, Software, Hardware, and Applications", Prentice-Hall, Inc., 1989.

[61] TREMBLAY, Jean-Paul and BUNT, R.B., "Introduction to Computer Science: An Algorithm Approach", McGraw-Hill Book Company, 1989.

[62] XYDEAS, C.S., EVCI, C.C., and STEEL, R., IEEE Trans. on Comms. Vol.COM-30, pp 1942-1954, August, 1982.

[63] DAUMER, William, R., "Subjective Evaluation of Several Efficient Speech Coders", IEEE Trnsactions on Communications, Vol.COM-30, No.4,

April 1982.

* IEEE International Conference on Acoustics, Speech, and Signal Processing March 1988

## BIBLIOGRAPHY

[1] BELLANGER, Maurice "Digital Processing of Signals; Theory and Practice", April, 1986, A Wiley Interscience Publication.

Chapter 11; Adaptive Filtering

Chapter 13; Application in Telecommunications.

[2] BAYLESS, J.W. et al. "Voice Signals: bit-by-bit", IEEE Spectrum, October, 1973, pp28-34.

[3] GOLD, B. "Digital Speech Networks", Proceedings of the IEEE, Vol.65, No.12, December, 1977, pp1636-1658.

[4] O'NEAL, Jr. J.B. "Delta Modulation Quatization Noise Analytical and Computer Simulation Results for Guassian and Television Input Signals", The Bell Systems Technical Journal, January,1966, pp117-141.

[5] WASSEL, I.J. et al. "Embedded Delta Modulation", IEEE Transactions

on Acoustics, Speech, and Signal Processing, Vol.36, No.8, August, 1988,pp12-1242.

[6] COLANTONIO, E.S., "Microcomputers & Applicaations", Copyright 1989 by D.C Heath and Company.

[7] SINNEMA, W. "Digital, Analog, and Data Communication, Second Edition", Prentice-Hall, Inc., 1986.

[8] SAVITCH, W.J. "Turbo Pascal 4.0/5.0: An Introduction to the Art and Science of Programming, second edition", The Benjamin/Cummings Publishing Company, Inc.

[9] WILLEN, D.C., and KRANTZ, J.I., "8088 Assembler Language Programming: The IBM PC", Second Edition, Howard W. Sams & Companay, 1987.

[10] MASTON, R.M., "Audio Circuits Manual", Heinemann Professional Publishing Ltd., 1989.

[11] HELD, G. "Communicating with the IBM PC Series:Concepts, Hardware, Software, Networking", John Wiley & Sons Ltd., 1988.

[12] BYLANSKI, P. and INGRAM, D.G.W., "Digital transmission Systems", Peter Peregrinus Ltd., 1980.

[13] INOSE, H. "An Introduction to Digital Communication Systems", Peter Peregrinus Ltd., 1979.

[14] GAYAKWARD, R.A., "Op-Amps and Linear INtegrated Circuits", Prentice-Hall, Inc., 1988.

[15] PROAKIS, J.G. and MANOLAKIS, D.G., "Introduction to Digital Signal Processing", Macmillan Publishing Company, 1988.

[16] SARGENT III, M. and SHOEMAKER, R.L., "The IBM Personal Computer from the Inside Out", Addison-Wesley Publishing Company, Inc., 1986.

[17] O'CONNOR, P.J., "Understanding Digital Electronics: How Microcomputers and Microprocessors Work", Prentice-Hall, Inc., 1984.

[18] HEFFER, D.E., KING, G.A., and KEITH, D., "Basic Principles and Practice of Microprocessors", Edward Arnold Publishers Ltd., 1981.

[19] MORRIS, N.M., "An Introduction to 8086/88 Assembler Language Programming for Engineers", McGraw-Hill Book Company, 1987.

[20] WILLIAMS, G.B., "Troubleshooting on Microprocessor Based Systems", Pergamon Press, 1984.

[21] JAYANT, N.S., and NOLL, P., "Digital Coding of Waveforms: Principles and Applications to Speech and Video", Prentice-Hall, Inc., 1984.

[22] SLATER, M., "Microprocessor Based Design: A Comprehensive Guide to Hardware Design", Mayfield Publishing Company, 1987.

[23] KERSHAW, J.D., "Digital Electronics: Logic and Systems", PWA-Kent Publishing Company 1988.

[24] SOUTHCOTT, C.B., BOYD, I., COLEMAN, A.E., and HAMMETT, P.G., "Low bit rate Speech Coding for Practical Applications", British Telecom. Technology Journal Vol. 6 No. 2, April, 1988.

[25] BARKER, Forrest, "Communications Electronics: Systems, Circuits, and Devices" Prentice-Hall International, 1987.

[26] JONES, R.H., and STEELE, N.C., "Mathematics in Communications Theory", Ellis Horwood Ltd. 1989.

[27] PEEBLES, Peyton Z., "Digital Communication Systems", Prentice-Hall International, 1987.

[28] PROAKIS, John G., Digital Communications", McGraw-Hill.

[29] LENK, John D., "Handbook of Data Communications", Prentice-Hall International 1984.

[30] MARTIN, James, "Telecommunications and the Computer", 2nd Edition, Prentice-Hall, Inc.

[31] GRIFFITHS, J.M. (Editor), "Local Telecommunication", IEE Telecommunications Series 10, Peter Peregrinus Ltd.

[32] LEA, Wayne A., "Trends in Speech Recognition", Prentice-Hall Inc. 1980.

[33] BAUCHAMP, K. and YUEN, C., "Data Acquisition For Signal Analysis", George Allen & Unwin Ltd. 1980.

[34] RABINER, Lawrence R. and GOLD, Bernard, "Theory and

Application of Digital Signal Processing", Prentice-Hall Inc. 1975.

[35] CLAYTON, G.B., "Data Converters", The Macmillan Press Ltd. 1982.

[36] FLANAGAN, James L. and RABINER, Lawrence R. ( Editors ), "Speech Synthesis", Dowden, Huchinson & Ross, Inc.

[37] FALLSIDE, Frank and WOODS, William A. ( Editors ), "Computer Speech Processing", Prentice-Hall International 1985.

[38] STERNS, S.D. and WIDROW, B., "Adaptive Signal Processing", Prentice-Hall Inc., 1985.

[39] SHORT, K.L., "Microprocessors and Programming Logic", Prentice-Hall International Editions.

[40] GIBSON, J.D., "Backward Adaptive Prediction as Spectral Analysis within a Closed Loop", IEEE Transactions on Acoustics, Speech, and Signal Processing, Vol. ASSP-33, No.4, October 1985, pp1166-1175.

[41] CHIEN, E.S.K., et al, "Cellular Access Digital Network CADN): Wireless Access to Networks of the Future", IEEE Communications magazine, Vol.25, No.6, June 1987, pp22-30.

[42] CARR, Joseph J., "Microprocessor Interfacing Handbook, A/D and D/A". TAB Books Inc., 1980.

[43] HALL, Douglas V., "Microprocessors and Digital Systems", McGraw-Hill Inc. 2e.

[44] JACKSON, L.B., "Digital Filters and Signal Processing", Kluwer Academic Publishers, 1986.

# APPENDIX

# PROGRAM LISTINGS

## A1 FORWARD ADAPTIVE PREDICTOR ALGORITHM

```
program Speech_Codec;
label
jmp;
const
block_size      = 100;    * samples of speech in a block; indicates rate of
predictor *               * update *

max_blocks      = 80;     * number of blocks of 100 speech samples to be *
                          * processed *
filt_order      = 4;      * order of the predictive filter; i.e., a predictor with
4                         * coefficients *
delta_max       = 10.0;   * upper limit of quantizer step size *
delta_min       = 0.0;    * lower limit of quantizer step size *
beta            = 0.95;   * decay factor of the quantizer step-size
                          *adaptation algorithm *
type
file_size       = array[1..max_blocks,1..block_size] of integer;
auto_coef       = array[0..filt_order] of real;
ref_coef        = array[1..filt_order] of real;
adpt_coef       = array[1..max_blocks,1..filt_order] of real;
pred_coef       = array[1..filt_order,1..filt_order] of real;

var
P, P1: pred_coef;
a, a1: adpt_coef;
```

```
x, x1: ref_coef;
r, e: auto_coef;
S, Y, Yr: file_size;
B:  array[1..filt_order,1..4] of integer;  *stores the encoded reflection*
                                                    *coefficients*
sg: array[1..filt_order] of integer;  *stores the sign bit of each refection
                                                    *coefficient*
count, k, j, i, bk_1, bk_2, buffer: integer;
sum, delta, n1, fk: real;
infile, outfile: text;
fname1, fname2: string;

begin {program}
count := 0;

{ neutralize arrays }
for count := 1 to max_blocks do
begin
for i := 1 to block_size do
S[count,i] := 0;
Y[count,i] := 0;
for j := 1 to filt_order do
a[count,j] := 0.0;
end;

{ open and read file }
writeln( 'Enter input file name:' );
readln( fname1);
writeln( 'Enter output file name:' );
readln( fname2 );
assign( infile, fname1); assign( outfile, fname2 );
```

```
reset( infile);                    * to read from infile *
rewrtie( outfile );* to write to outfile *
while not eof( infile ) do
begin { read the number of blocks each containing the number of speech samples
}
for count  := 1 to max_blocks do
for i  := 1 to block_size do
readln( infile, S[count,i] );
end;

{Now compute the predictor coefficients for each block}
for count  := 1 to max_blocks do
begin
for k  := 0 to filt_order do
begin
{compute the autocorrelation coefficients}
sum  := 0.0;
for i  := 1 to block_size - 1 - k do
sum  := sum + S[count,i] * S[count,i + k];
r[k]  := sum;
end;
e[o]  := r[o];
for k  := 1 to filt_order do
begin
{compute the reflection coefficients and then the predictor coefficients}
sum  := 0.0;
for j  := 1 to k - 1 do
sum  := sum + P[k - 1,j] * r[k - j];
if e[k - 1] = 0.0 then
x[k]  := 0.0
else
```

```
x[k] := (r[k] - sum)/e[k - 1];
P[k,k] := x[k];
for j := 1 to k - 1 do
P[k,j] := P[k - 1,j] - x[k] * P[k - 1,k - j];
if k = filt_order then
for j := 1 to filt_order do
a[count,j] := P[k,j];
e[k] := (1 - x[k] * x[k]) * e[k - 1];
end;

{encode the reflection coefficients for transmission}
for k := 1 to filt_order do
begin
n1 := x[k];
if n1 >= 0.0 then
begin
sg[k] := 0;        *indicates positive sign*
n1 := n1;
end
else
begin
sg[k] := 1;        *sign is negative*
n1 := abs(n);      *consider only the magnitude of the coefficient*
end;
for t := 1 to 4 do
{encode the coefficient with four bits to make five bits including the sign bit}
if n1 >= exp(-t * ln(2.0)) then
begin
B[k,t] := 1;
n1 := n1 - exp(-t * ln(2.0));
end
```

```
else
begin
B[k,t] := 0;
n1 := n1;
end;
end;

{Now code and transmit the blocks of speech samples. Transmit also the
refection} {coefficiebnts for each block with the first sample information of that
block}
for count := 1 to max_blocks do
begin
for i := 1 to block_size do
begin
sum := 0.0;
for j := 1 to filt_order do
sum := sum + a[count,j] * Y[count,i - j];
if count = 1 then
begin
if i = 1 then
Y[count,i] := S[count,i];      *the reconstructed value of the first sample of the*
                               *first block of samples is equal to the sample,
                               *since the prediction is zero*


else
goto jmp;
end
else
goto jmp;
```

```
jmp: begin
if S[count,i] - sum >= 0 then
bk  := +1
else
bk  := -1;          *program outputs +1 if the prediction is positive and -1 if it is*
                    *negative*

{compute a new step size}
if bk  =  bk_1 then
begin
if bk_1  = bk_2 then          *an indication of slope overload noise*
fk  := (1 - beta) * (delta_max - delta_min)
else
fk  := 0.0;
end
else                                *no slope overload*
fk  := 0.0;
delta  := beta * delta + (1 - beta) * delta_min + fk;
Y[count,i]  := trunc(sum + delta * bk);
end;
bk_2  := bk_1;
bk_1  := bk;
end;
{At the receive, receive the transmitted codes and reconstruct the transmitted
sample.}
{If the codes are those for the first sample of the block of samples, then receive
and} {decode a new set of reflection coefficients. Compute the corresponding
predictor}
 coefficients and use them to predict the first and subsequent sample values in the
block }
for i := 1 to block_size do
```

```
begin
if i := 1 then
begin
for k := 1 to filt_order do
begin


for t := 1 to 4 do
sum := sum + exp(-t * ln(2.0));
if sg[k] := 0 then
x1[k] := + abs(sum);
else
x1[k] := - abs(sum);
P1[k,k] := x1[k];
for j := 1 to k - 1 do
P1[k,j] := P1[k - 1,j] - x1[k] * P1[k - 1,k - j];
if k := filt_order then
for j := 1 to filt_order do
a1[j] := P1[k,j];
end;
{Now predict and reconstruct the transmitted sample vlaue}
for j := 1 to filt_order do
sum := sum + a1[j] * Yr[count - j];
Yr[count,i] := trunc(sum + delta * bk); *asuming errorless transmission, delta*
                                        *is computed as for the transmitter*

writeln(outfile,Yr[count,i]; *stores reconstructed sample value in the output
                             *file*
                             *saves the last three previous bits*

end;
close(infile); close(outfile); *close the input and output files*
```

end. {program}

## A2 BACKWARD ADAPTIVE PREDICTOR ALGORITHM

```
program Speech_Codec;
{This program opens a file of speech samples, reads 8000 samples known as}
{Max_Samples, from the file, and stores the samples in a memory location called
SEG} {(for segment). ADPCM coding is then done on each sample in the
segment, using the }
{backward adaptive technique to update both the predictor coefficients and the}
{quantizer step size.}

const
Max_Samples    = 8000;
Filter_Order   = 4;
A              = 10.0;
B              = 100.0;
em             = 100.0;
delta_max      = 10.0;
delta_min      = 0.0;
beta           = 0.95;

var
Seg, Eval: array[1..Max_Samples] of integer; *stores 8000 speech samples*
a: array[1..Filter_Order] of real; *stores the predictor coefficients*
Rval, P: array[1..Max_Samples] of integer; *stores the reconstructed sample
```

```
values*
In_File, Out_File: text;        *the input and output files respectively*
count, order:                   *sample and predictor coefficient number*
                                *respectively*
step_size, sum_sq, zg_sq, err: real;
sgne, sgnx, sgnx1:  integer; *indicates the polarity of of the calculated error and
                                *the previous reconstructed sample sample*
                                *respectively*

bit, bit_1, bit_2, k: integer;
sum, accum, Pval, buffer, fk: real;
FileNameR, FileNameW: string;
begin
{neutralize variables}
count := 0;
Pval := 0.0;
sum := 0.0;
stp_size := 0.0;
order := 0;
for count := 1 to Max_Samples do
begin
Seg[count] := 0;
Rval[count] := 0;
Eval[count] := 0;
end;
for k := 1 to Filter_Order do
a[k] := 0.0;

{Open a file and read 8000 samples from it. Open another file to store the
processed}
{samples later}
```

```
writeln('Enter file to be processed:');
readln(FileNameR);
writeln('Enter name of output file:');
readln(FileNameW);
assign(In_File, FileNamwR); assign(Out_File, FileNameW);
reset(In_File); rewrite(Out_File);
while not eof(In_File) do
for count := 1 to Max_Samples do
begin
sum := 0.0;
for k := 1 to Filter_Order do
sum := sum + a[k] * Eval[count -k];
Pval := sum;
err := Seg[count] - Pval;
if err >= 0 then
sgne := +1
else
sgne := -1;
bit := sgne;
if bit = bit_1 then
begin
if bit_1 = bit_2 then
{sign of slope overload noise}
fk := (1 - beta) * (delta_max - deta_min)
else
fk := 0.0;                    *if no slope overload detected*
end
else
fk := 0.0;
step_size := beta * step_size + (1 - beta) * delta_min + fk;
Eval[count] := trunc(Pval + step_size * bit);
```

```
bit_2 := bit_1;
bit_1 := bit;
for j := count - 1 - em to count - 1 do
sum_sq := sum_sq + Eval[j] *Eval[j];
zg_sq := sum_sq/em;
P[coun] := A/(B + zg_sq);
for k := 1 to Filter_Order do
begin
{Update the predictor coefficients}
if Eval[count - k] >= 0 then
sgnx := +1
else
sgnx := -1;
a[k] := a[k] + P[count] * sgne * sgnx;
writeln(Out_File,Eval[count]; *writes the reconstructed sample value in the*
                                *output file*
end;
close(In_File); close(Out_File);
end.
```