LOW RATE ENCODING OF
AUTOREGRESSIVE SOURCES

by

© MADAN L. SETHIA, B.Sc. (Cal.), B.Sc. Eng. (U.K.)

A Thesis

Submitted to the Faculty of Graduate Studies
in Partial Fulfilment of the Requirements
for the Degree
Master of Engineering

McMaster University
March 1981

MASTER OF ENGINEERING (1981)
(Electrical Engineering)

McMaster University
Hamilton, Ontario

TITLE: Low Rate Encoding of Autoregressive Sources

AUTHOR: Madan L. Sethia, B.Sc. (Cal.), B.Sc. Eng. (U.K.)

SUPERVISOR: Dr. J. B. Anderson

NUMBER OF PAGES: vi , 99

SCOPE AND CONTENTS: Various methods of digital encoding of autoregressive sources are examined. The relevance of the results to low rate waveform encoding of speech is stressed.

ABSTRACT

Various approaches, traditional as well as non-traditional, are utilized to encode Gaussian and Laplacian distributed general autoregressive sources at rates of 1 and 2 bit per source letter. The performance of the traditional DPCM encoder is evaluated. At these low rates, DPCM turn out to be rather ineffective from a data compression point of view. Underlying laws governing the performance loss caused by the quantiser non-linearity in the predictor loop are detected experimentally. It is found that tree searching improves the performance substantially and the gain is a very well behaved function of some well known source statistics. Effect of tree searching on mismatched source predictor is examined: The results indicate that tree searching is not a substitute for a matched predictor. The performance of an intution-based smoothing filter in cascade with the DPCM encoder is evaluated when the predictor is matched as well as when mismatched to the source. Such smoothing is not helpful. Finally, a certain random coding scheme is used at rate 1. The performance of such an information theoretic inspired scheme is compared with the tree searched DPCM. Wherever appropriate, the relevance of results to low rate waveform encoding of speech is stressed.

ACKNOWLEDGEMENTS

I wish to register my special thanks to my supervisor, Dr. J. B. Anderson for his help, guidance and encouragement in course of this work.

I wish to convey my sincere appreciation to Linda Hunter for typing the entire manuscript.

TABLE OF CONTENTS

CHAPTER 3 (cont'd.)

CHAPTER 1

INTRODUCTION

The first half of this century was an era of prolific growth in analogue radio communication. Since then, there has been a gradual but accelerating trend from analogue to digital communication. Earlier, the emphasis was mainly on telephony. Digital techniques in satellite communications are now commonplace. More recently, terrestrial digital radio has appeared.

Discussions of manifold advantages of digitally coding an analogue signal are widespread in literature [1]. In brief, digital systems offer ruggedness, the capability of regenerating the signal as many times as required without cumulative increase in distortion as long as fading is not too severe, an attractive power-bandwidth product, and a uniform format for all kinds of signals. These, along with the evolution of solid state electronic technology needed to support the development of efficient, flexible and error-free digital communication systems, are the reasons of this phenomenal growth in digital communication.

The theoretical foundations of digital communication were laid toward the end of the first half of this century by the celebrated "Mathematical theory of communications" papers of Claude E. Shannon [2]. Since then, relentless effort has been directed toward refinement, embellishment and practical realization of the theory; so uniquely perceived and deduced by Shannon.

1.1 Digital Transmission of Speech

An area of current research interest is the digital transmission of speech [1, 3]. Low cost, low rate digital voice coders that produce high quality, highly intelligible speech while retaining the robustness to impairment are still not a practical reality.

For an analogue source such as speech to be transmitted by digital means, the source must be converted to digital form. Digital transmission of speech has progressed along two entirely different lines. In the so-called vocoders, only certain important features of speech are extracted, encoded and sent over the channel. On the receiving side, speech is synthesized from the received features. The method, though efficient from data compression point of view, is very complex. The other approach is waveform encoding. In such schemes, a time-continuous waveform is sampled at Nyquist rate and the sampled amplitudes are digitised and sent over the channel. The receiver recovers the original waveform from the received samples. Though considerably less complicated than the vocoder, waveform encoders have so far not been able to fully utilize the redundancy of speech.

The earliest waveform digitisation scheme is Pulse Code Modulation and it is still in wide use. For sources with correlated samples, PCM is not very effective from the data compression point of view. Delta Modulation attempts to reduce redundancy due to memory but suffers from slope overload. An amplitude adaptive form of Delta modulation which lessens the slope overload problem, when used to encode speech requires 30 K bits/sec. for a telephone quality reproduction. One promising technique is the so-called differential PCM which assumes the source (e.g., speech) to be an auto-regressive source and quantises the deviations from the predictions based on previous samples in an LMS (least mean square error) sense.

1.2 Scope of Thesis

In the remainder of this chapter, we shall formally introduce auto-regressive sources and differential PCM in some detail. In Chapter 2, general concepts of rate distortion theory will be discussed, some of which if not directly applicable to the situation at hand, are at least instructive where no other more relevant theory exists.

In Chapter 3, we shall discuss and investigate the effect of the quantiser non-linearity which lies in the predictive loop of DPCM, when used to encode auto-regressive sources with both normal as well as Laplacian distributions at low rates of 1 bit and 2 bit per source letter. At these low rates, the presence of memory has a strongly adverse effect on the performance of an encoder. In Chapter 4, we shall use a multipath search algorithm (the M-algorithm) as an encoder along with the DPCM code generator to find if tree searching can fix the damage that the quantiser causes at these low rates. It very nearly does. Effects of tree searching on mismatched source-predictor pairs will also be studied. In these, the predictor structure is not matched to the source correlation.

According to rate-distortion theory, a source-encoder pair, even if matched, is not optimal. The theory requires that a certain smoothing filter be cascaded with the matched decoding filter to achieve optimal digitisation at high rates. Unfortunately, at low rates, we have an additional problem in that the theory does not tell how to find such a filter. In Chapter 5, we shall use a smoothing filter derived on intuitive grounds in cascade with the matched or mismatched decoding filter to encode auto-regressive sources. We find that such smoothing has relatively little effect on SNR performance.

Finally, in Chapter 6, we use random tree codes to encode auto-regressive sources at low rates of 1 bit per source letter. We shall find the comparison between the SNR performance of random codes and predictive quantiser codes at this low rate interesting.

This work will never involve speech directly; nonetheless, our whole effort is directed toward encoding of speech-like sources. Speech has been frequently and quite successfully modelled as an auto-regressive source [See 33 among others]. Low rate encoding of speech is a matter of considerable interest to communications engineers. Recently, a lot of emphasis has been placed on tree encoding of auto-regressively-modelled speech at low rates and there are reports of substantial performance improvements. However, the origin of the gain has been mysterious: Does tree-searching replace and enhance prediction, or does the gain have a pure "coding" origin? We establish here that the tree coding gain is, in fact, independent of gains due to prediction, and that it primarily depends on the encoder bit rate.

1.3 Auto-regressive Sources

The auto-regressive model constitutes one of the most commonly used representations of practically occurring time series, both stationary as well as non-stationary. Many sources of interest such as speech, music, and pictures to name a few, are submused under the general category of auto-regressive sources.

In this model, the current value of the process is expressed as a finite, linear aggregate of previous values; that is, the current value is regressed on the previous values of itself.

A pth order time-discrete auto-regressive (AR(p)) source $\{x_t, t = 0, 1, 2, \ldots\}$ is described by the difference equation,

$$x_t = \sum_{i=1}^{P} a_i x_{t-i} + e_t \qquad (1.1)$$

Where a's are called auto-regression constants and $\{e_t\}$ is a sequence of independent random variables drawn from some fixed distribution. P is called the order of the auto-regressive source. Fig. 1.1 shows the block schematic diagram of a recursive circuit that generates x_t.

Define an operator z such that,

$$x_{t-1} = z^{-1} x_t \quad \text{or,} \quad x_t = z\, x_{t-1}$$

This will permit us to rewrite (1.1) as,

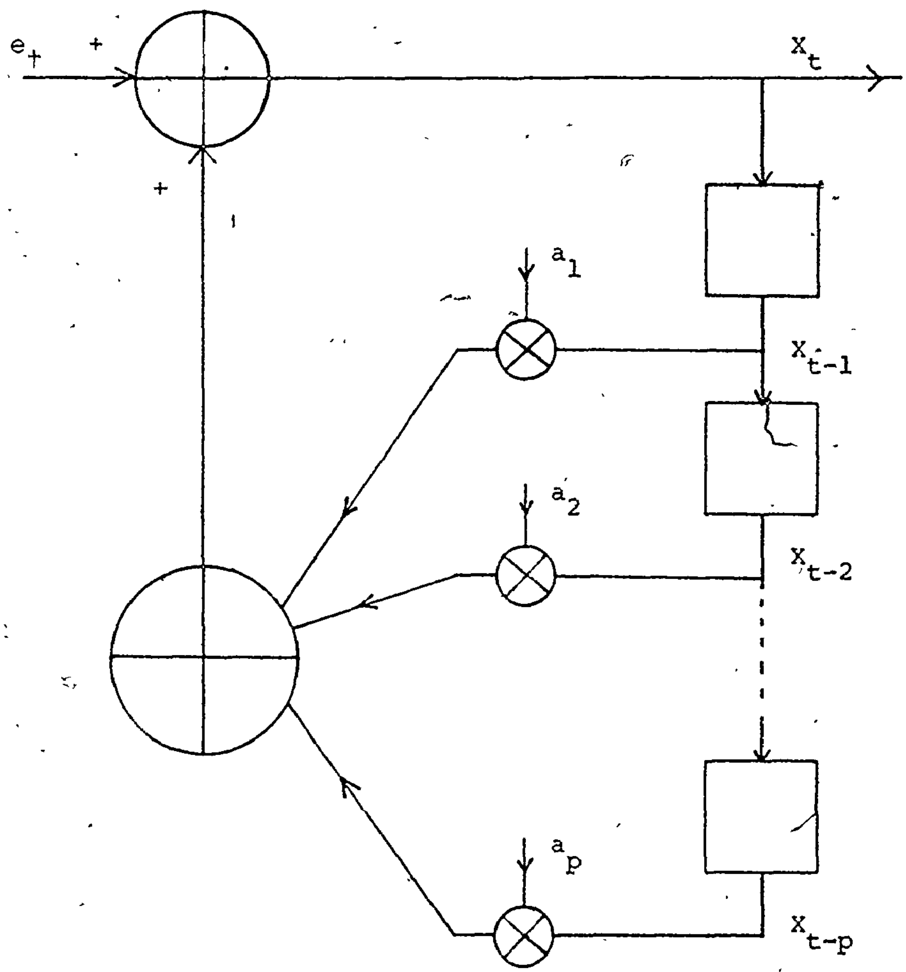

Fig.1.1 Recursive circuit that generates an autoregressive time series.

$$x_t = \frac{1}{1 - a_1 z^{-1} - a_2 z^{-2} \dots a_p z^{-p}} e_t \qquad (1.2)$$

The behaviour of a typical realisation of a real auto-regressive source is solely determined by the magnitude of the roots {G} of the pth order polynomial [34].

$$a(z) = 1 - a_1 z^{-1} - a_2 z^{-2} \dots a_p \cdot z^{-p} \qquad (1.3)$$

Depending on the location of the poles in a complex plane, AR(p) sources are broadly classified in four major categories:

Type 1: $|G| < 1$, all G; i.e., all roots lie in a unit circle. This type is asymptotically stationary and the transients in the response die out in an exponential fashion. This model has been frequently used to model speech, and due to the ease with which it lends itself to experimental and analytical tools, it has found considerable favour. The model is stationary in the mean, i.e., the time series fluctuates about a fixed mean.

Type 2: One or more roots lie on unit circle but not all. This type exhibits non-stationarity. Though capable of modelling quasi-stationary sources like speech more precisely than Type 1, they have not been used very frequently. This model can take account of fluctuations in the mean of a time series.

Type 3: All roots lie on the unit circle, $|G| = 1$, all G. This type is also non-stationary but the variance of X_t grows only algebraicly. More commonly known as a Weiner sequence or unimpeded random walk, the problem of encoding these has been studied by Berger [14] among others.

Type 4: One or more roots lie outside the unit circle. This type is also non-stationary but here the series breaks loose, i.e., the variance of x_t grows exponentially with time. The explosive growth is limited only by the system energy. Even though physical phenomena such as bectirial growth and nuclear chain reactions fall in this category, this type has not been very popular because of its pathological behaviour.

When a communications engineer talks about an auto-regressive source, he generally implies Type 1, and it is this type with which we concern ourselves mainly in this thesis. Unless indicated otherwise, by auto-regressive source, we shall imply this type.

1.4 Philosophy of Differential PCM

The purpose of a source-encoding scheme is to minimise the information content of the source output subject to some fidelity criterion, before it is handed over to the channel for transmission. In case of auto-regressive sources, encoding involves one more step than is needed for encoding sources with independent identically distributed outputs.

1.4.1 Prediction

An auto-regressive source is characterised by some form of memory as is evident from the difference equation,

$$x_t = \sum_{i=1}^{p} a_i x_{t-i} + e_t$$

Thus, if the past history of source outputs is known, it is possible to say something about the next source output. A data compression scheme must naturally take advantage of this situation. Instead of encoding x_t itself, an attempt should be made to encode the difference between the current value x_t and the prediction based on the previous values assumed by the process. This is the basis of Differential Pulse Code Modulation and distinguishes it from simple Pulse Code Modulation. The net effect of encoding the prediction error is a reduction in the power of input to the quantiser and consequently in improved performance. The performance improvement achieved by Differential PCM over PCM at high quantiser rates is called the Signal-to-Noise Improvement Ratio [5, 6] denoted by SNI.

Prediction, in addition to the fundamentally important role it plays in compression of sources with memory, plays an equally important role in forecasting, which has become a part of our daily life. So important is its role in time series analysis that one can hardly overstate it.

1.4.2 Quantisation

An N level quantiser has an input x, the domain of which is a real line, and an output that can assume only one of N discrete

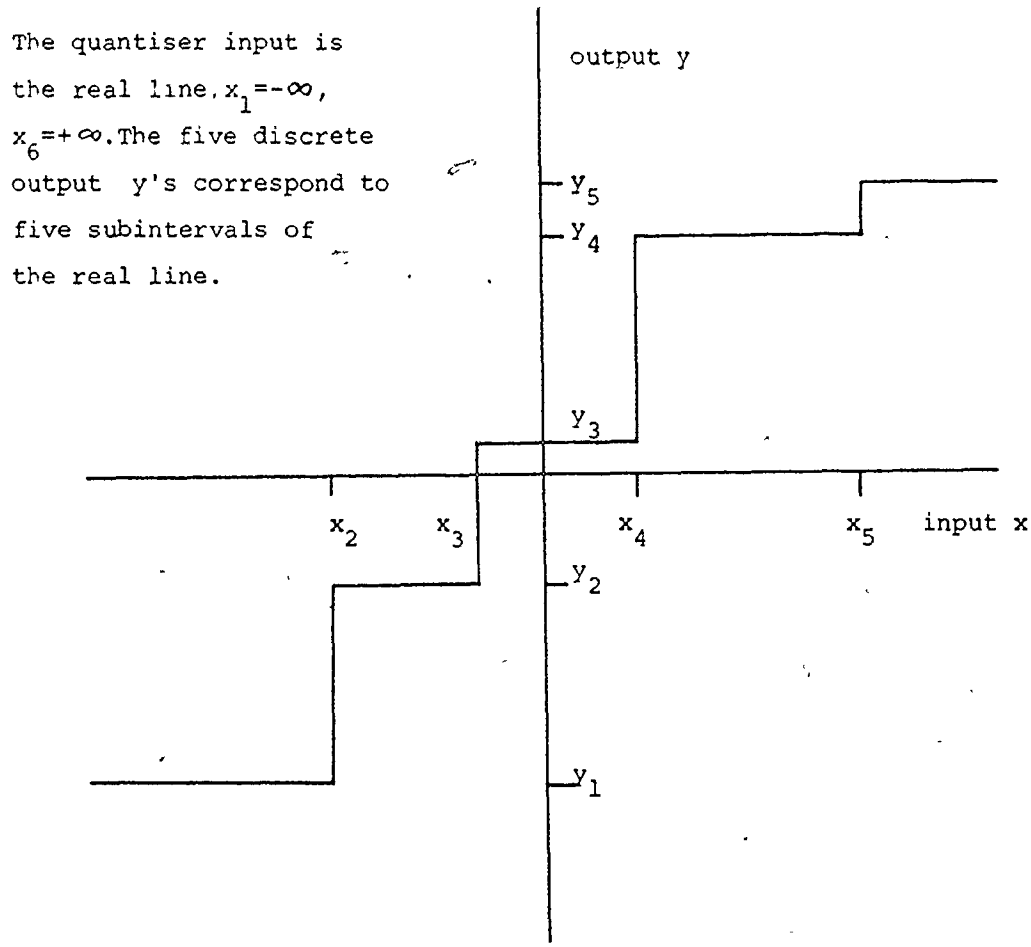

Fig.1.2 A five level quantiser characteristic

real values $Y = y_1 < y_2 < y_3 \ldots < y_N$. The quantiser subdivides its input range into N intervals by thresholds $x_1 < x_2 < x_3 < \ldots x_{N-1}$ such that the quantiser output is y_K if and only if $x_{K-1} < x \leq x_K$. The input-output characteristic of a five-level quantiser is shown in fig. 1.2.

In a uniform quantiser, the spacing between successive thresholds is equal. At coarse quantisation, a uniform level quantiser is nearly optimal for Gaussian sources but as the number of quantisation levels increases, the gap between uniform quantiser and the optimal non-uniform quantiser broadens. It is never very great for Gaussian sources but for Laplacian and Gamma distributed sources a non-uniform quantiser is desirable even with coarse quantisation.

The representation of a continuous amplitude signal by certain discrete levels inherently introduces an error, giving rise to the so-called quantisation noise. The signal-to-quantisation noise ratio of a standard differential PCM system is given by the well known 6 dB/octave law [6, 7]. For Gaussian sources the law states,

$$S/N = 6.02\,n + SNI - 7.2 \text{ (in dB)} \qquad (1.4)$$

where 2^n is the number of quantisation levels. n is sometimes called the rate, i.e., the number of binary symbols required to encode the amplitude of one sample. The law is an approximation and holds for n above 5 or 6. It holds exactly for uniformly distributed sources without the 7.2 term in equation (1.4) at all rates. For PCM or a source without memory the SNI becomes zero. Exact upper-bounds for S/N performance of DPCM and PCM are derived by O'Neal [7].

1.5 DPCM Code Generation and Code Trees

Even though suboptimal from rate distortion theoretic point of view, DPCM has attracted considerable interest because of its simplicity of implementation and its attractive signal-to-noise ratio gains.

A block diagram of DPCM encoder is shown in fig. 1.3. The similarity between the filter in fig. 1.3 and the recursive filter of fig. 1.1 that generates the auto-regressive time series is clear. In fig. 1.3, source output x_t is fed to the encoder and in the mean time the Linear predictor in the dotted box makes an estimate of the current input x_t on the basis of previous inputs to the encoder. The difference e_t between the current input x_t and the predicted value $\hat{x}_t$ is fed to the quantiser to produce the encoder output $e_{q(t)}$. The dotted box in fig. 1.3 considered with $e_{q(t)}$ as the input and $\underline{x}_t$ as the output is the DPCM decoder filter (or DPCM code generator). The output $\underline{x}_t$ of the DPCM code generator filter can be mapped onto a tree-like structure as we shall see.

In fig. 1.3 the operation of the recursive filter inside the dotted box can be expressed as,

$$\underline{x}_t = \frac{1}{1 - a_1 z^{-1} - a_2 z^{-2} \dots a_m z^{-m}} e_{q(t)} = \frac{1}{a(z)} e_{q(t)}$$

$$\underline{x}_t = (1 + c_1 z^{-1} + c_2 z^{-2} + \dots + c_k z^{-k} + \dots) e_{q(t)} \qquad (1.5)$$

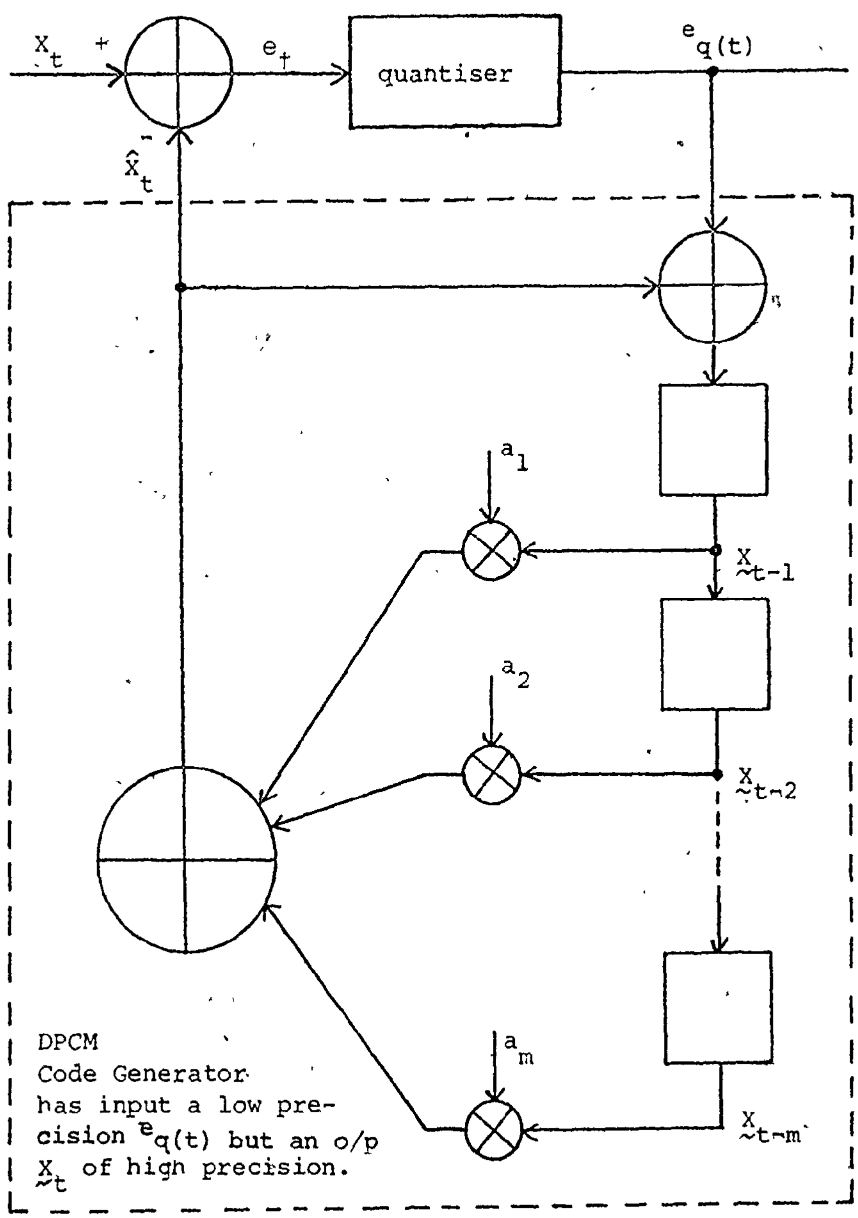

Fig.1.3 Schematic diagram of DPCM encoder.

where c's are the impulse response of $1/a(z)$. In practice, the sequence of c's converge quite rapidly, therefore we can turncate it at some k. The result is a transversal realization of the filter in the DPCM circuit. This realization will have a longer, but still quite manageable order. Fig. 1.3a shows the transversal realization of DPCM encoder.

Rewriting (1.5) in matrix notations,

$$\underline{x}_t = [1, c_1\ c_2\ \ldots\ c_k] \begin{bmatrix} e_{q(t)} \\ e_{q(t-1)} \\ \cdot \\ \cdot \\ e_{q(t-k)} \end{bmatrix} \tag{1.6}$$

$$= f: \begin{bmatrix} e_{q(t)} \\ e_{q(t-1)} \\ \cdot \\ \cdot \\ e_{q(t-k)} \end{bmatrix}$$

where,

$$f = [1\ c_1\ c_2\ \ldots\ c_k]$$

Assuming zero initial conditions, at $t = 1, 2, \ldots, t$, we have

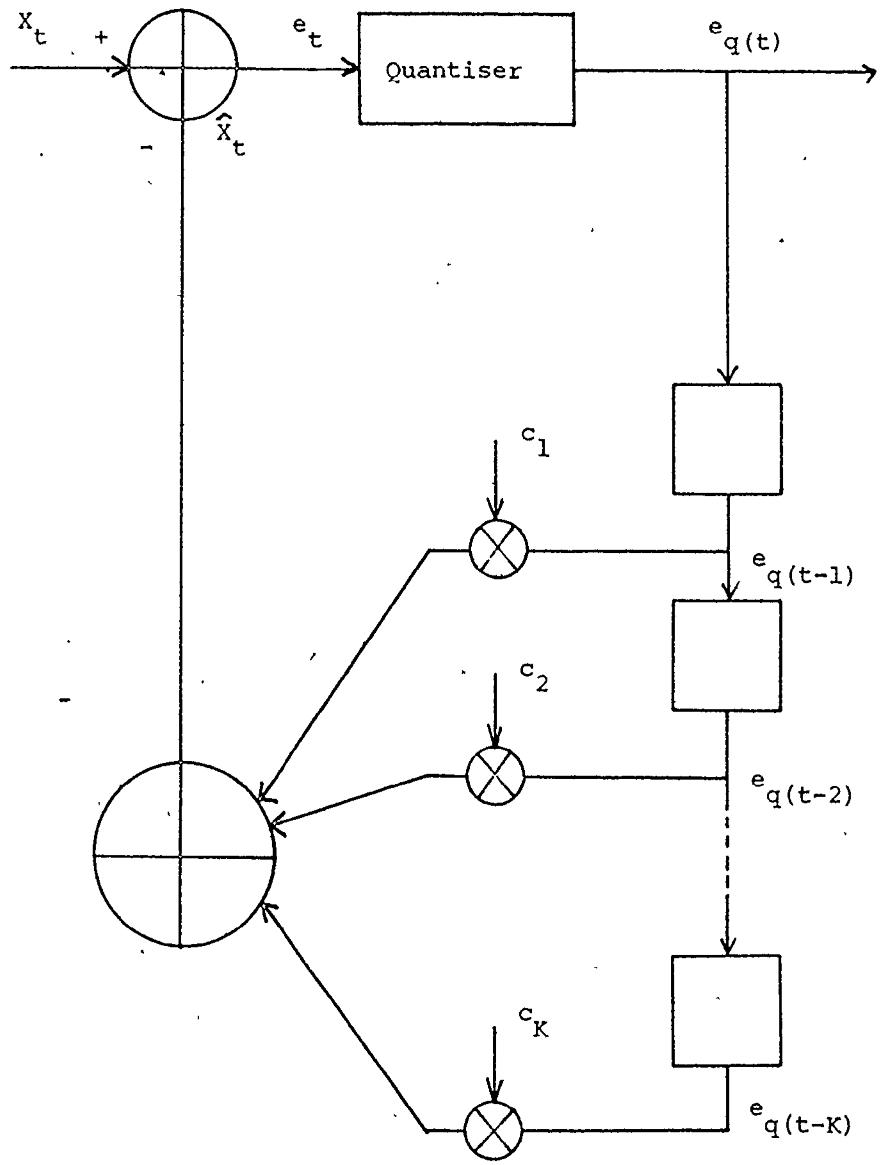

Fig. 1.3a The transvarsal realization of DPCM encoder. c's are the impulse responce of $1/a(z)$.

$$\underline{x}_1 = f: \begin{bmatrix} e_{q(1)} \\ 0 \\ 0 \\ . \\ . \\ 0 \end{bmatrix} ; \quad \underline{x}_2 = f: \begin{bmatrix} e_{q(2)} \\ e_{q(1)} \\ 0 \\ . \\ . \\ 0 \end{bmatrix}$$

$$\underline{x}_t = f: \begin{bmatrix} e_{q(t)} \\ e_{q(t-1)} \\ . \\ . \\ . \\ e_{q(t-k)} \end{bmatrix}.$$

Clearly, the code generator maps the k-dimensional input vector onto real line $\mathbb{R}$, $f: e_q^{\ k} \rightarrow \mathbb{R}$. As an example, a rate one system accepts at each time a binary symbol either 0 or 1 (in practice a 0 and 1 implies inputs +1 and -1 times the quantiser step size, respectively) and outputs a reproduction symbol $\underline{x}_t$. Since the current output depends only on current input and previous inputs, the output of such a code generator can be expressed on a tree-like structure shown in fig. 1.4. An input zero specifies an upper branch of bifurcation while a one specifies the lower one.

A path through the tree is specified by its pathmap. For instance, the path map 1010 specifies moving up at the first branching level, down at second, up at third and again down at the fourth to produce the output indicated along the branches traversed: f(000...0),

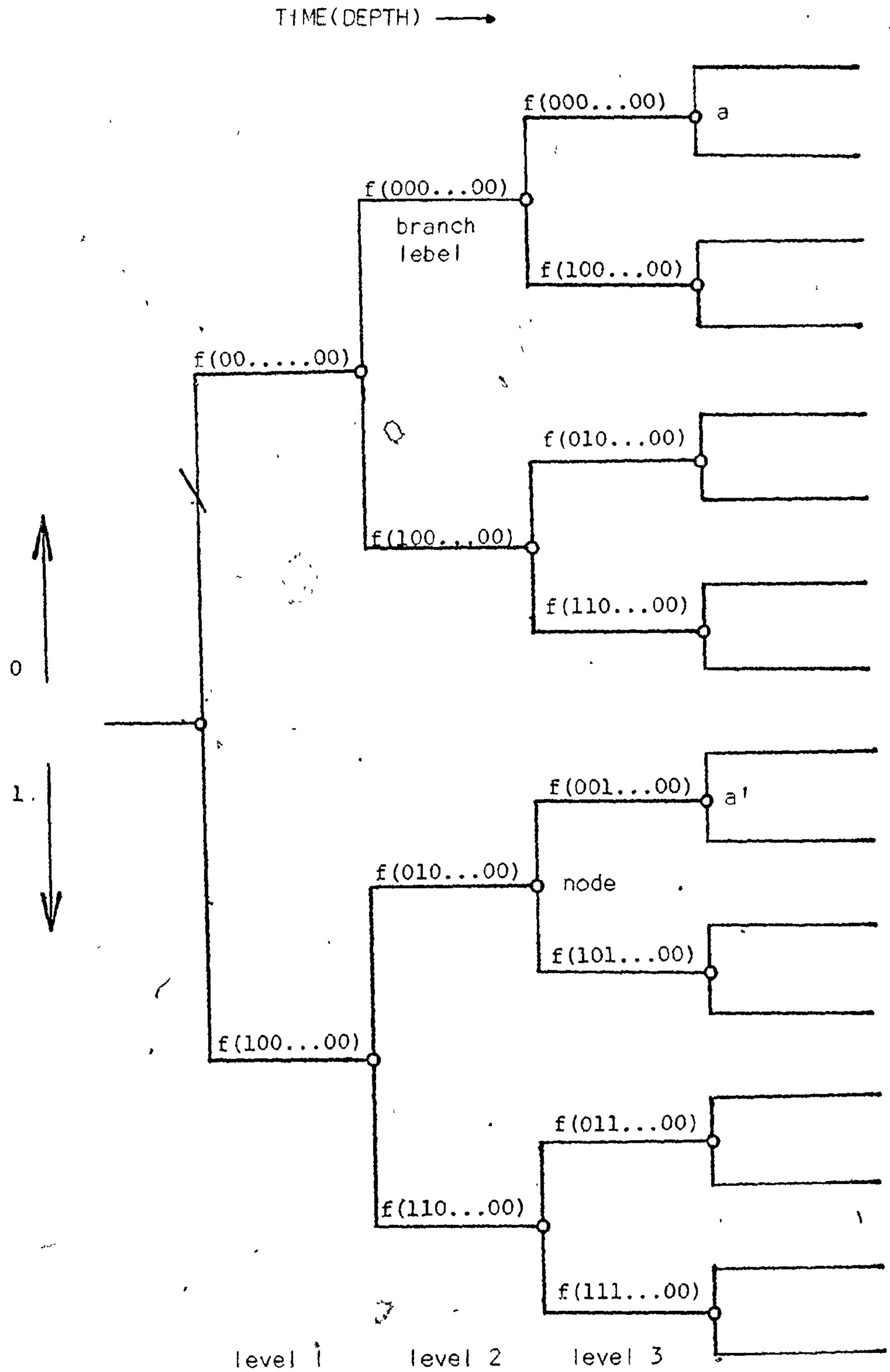

Fig. 1.4 The binary rate one code tree

f(100...0), f(0100...0), f(1010...0). f operates on k digits; one current digit and k-1 past digits. K is called the constraint length. If b branches come out of a node and β symbols correspond to each branch then the rate of tree code is,

$$R = \frac{\log_2 b}{\beta} \text{ bits/source symbol}$$

Based on the error between the current source output x_t and the current generator output $\underline{x}^*_t$, the encoder makes a final decision to branch either upward or downward. From the source coding point of view, this single line of decision developed by DPCM code generator is its clearest shortcoming. In a multipath search, the encoder pursues several paths at any given time and chooses among them at some latter point based on some fidelity criterion usually called the path metric Most commonly used path metric is the cumulative squared error. We shall return to methods of searching in sec. 4.2 and 4.3.

CHAPTER 2

REVIEW OF RATE DISTORTION THEORY

All practical channels have a finite capacity, which is to say that they can only carry a finite amount of information in a given time, whereas most sources of information in real life have infinite entropy because the space of their probable outputs occupies a part of the real line $\mathbb{R}$. The implication is that we cannot communicate the information generated by an analogue source to a user with exact reproduction. Some distortion is inevitable. Rate distortion theory serves to lower-bound this distortion.

2.1 Basics of Rate-Distortion Theory

The aforementioned character of physical world leads to the necessity of some kind of matching device between the channel which carries the information but only at a finite rate and the source that generates the information at an infinite rate or at least at some higher rate. Source encoding achieves just this end. A source encoder in general maps a sequence of outputs x_t of a discrete-time source into a set of preselected messages. It partitions the space of source outputs, usually the real line, into a set of equivallence classes and tells the channel encoder to which of these classes the source output belongs. The whole operation can be conveniently described by the mapping $f : \mathbb{R}^n \to A^k$, where blocks of n source output symbols are the input to the encoder, blocks of k symbols are its output, and A is the encoder output alphabet. In case of non-continuous source outputs, $\mathbb{R}$ can always be replaced by the appropriate discrete space.

If $A = \{0, 1\}$, we define the rate of encoding to be the average number of binary digits at the encoder output per source letter. In fixed rate systems, the same number of source symbols are input to the encoder each time and a fixed number of binary digits appear at the output of the encoder. A lower rate results in higher compression because fewer digits are used to encode a given source symbol. Finally, use of term time-discrete in describing a source encoder is completely general, since a band-limited time-continuous source can always be converted into a time-discrete source without loss of any information, by sampling or some other convenient transformation.

In arriving at its ultimate performance bounds, rate-distortion theory makes use of random coding arguments; it does not, however, make clear how practical codes may be designed. The approach inherent in random coding is to populate the tree branches with random deviates drawn from an appropriate distribution. Unfortunately, this results in a time-varying decoder and storage and computation that grow exponentially with n. Though much research has been directed toward overcoming these difficulties, much remains to be done to find practical codes. Encoding schemes such as PCM, DPCM and Delta Modulation, the so-called ad-hoc schemes as the information theoretician prefers to call them, have been very popular. There are good reasons for this. DPCM, which is suboptimal from rate distortion theory point of view, has performed quite well for sources with memory and particularly well when used in conjunction with some kind of search algorithm [23]. PCM and Delta Modulation have great simplicity and at the same time give good performance. Vocoders, though complex, give similar performance at a much lower rate.

2.2 Some Theorems

Analytical evaluation of predictive quantisation (the DPCM encoder) has had only a limited success at low rates. The difficulty arises from the combination of the nonlinearity introduced by the quantiser and the memory inherent in such systems. The problem of optimal DPCM at low rates for auto-regressive sources of order higher than one is still to be solved, a theorem by Gray [5] does give some insight into this. Gray's theorem does not apply to DPCM, but rather shows the existence of a code with structure as yet unspecified at low rates.

Theorem 2.1

Let $\{x_t\}$ be a p-th order auto-regressive source generated by an i.i.d.* $N(0, \sigma^2)$† sequence $\{z_t\}$ and the auto-regression coefficients $a_1, a_2, \ldots a_p$. Then MSE (mean squared error) rate distortion function of $\{x_t\}$ is given parametrically by,

$$D_\theta = \frac{1}{2\pi} \int_{-\pi}^{\pi} \min \left[\theta, \frac{1}{g(\omega)}\right] d\omega$$

$$\text{and} \quad R(D_\theta) = \frac{1}{2\pi} \int_{-\pi}^{\pi} \max \left[0, \frac{1}{2} \log \left\{\frac{1}{\theta g(\omega)}\right\}\right] d\omega$$

The rate distortion function R(D) is the lower bound on rate required to encode the sequence with fidelity D.

$$\text{and} \quad g(\omega) = \frac{1}{\sigma^2} \left| \left[1 + \sum_{k=1}^{p} a_k e^{-j\omega k}\right] \right|^2$$

* independent identically distributed

† zero-mean normal density with variance σ^2

A second theorem relates the rates required for autoregressively correlated sources to that required for memoryless Gaussian source.

Theorem 2.2

Let $\{x_t\}$, $\{z_t\}$ and $g(\omega)$ be as in Theorem 2.1 and let $R_x(D)$ and $R_z(D)$ denote the MSE rate-distortion fucntions of $\{x_t\}$ and $\{z_t\}$, respectively. Then where Δ is the essential supremum of $g(\omega)$, we have

$$R_x(D) = R_z(D) \quad \text{for } 0 \le D \le \frac{1}{\Delta} = D_0$$

$$\text{and} \quad R_x(D) > R_z(D), \quad \text{for } D > \frac{1}{\Delta}$$

This result is remarkable in that it implies that a suitable coding strategy can always be found such that the rate required to reproduce a weighted sum of z_t with some $D \leq \frac{1}{\Delta}$ is the same as z_t itself, despite the larger variance of the weighted sum.

By low rates for autoregressive sources, we imply rates which result in a D which is either greater than or in the vicinity of $1/\Delta$. At these low rates, Theorem 2.2 says that all codes must suffer some loss in performance; for DPCM, the situation is perhaps made worse by the destructive interaction of the quantiser nonlinearity and the feedback. At rates higher than the so-called low rates, it is at least possible in principle to recover completely the damage caused by the quantiser.

Quantitative evaluation of the performance loss caused by the quantiser in the predictive loop of practical DPCM is still an unanswered question for general autoregressive sources. It is one of the goals of this thesis to answer this question through experimental analysis, and in particular to see if DPCM performance depends on Δ, the essential infimum $1/\Delta$ of the spectrum $1/g(w)$ in theorem 2.2.

Δ can be conveniently calculated from the following relation,

$$\Delta = (1 + |a_1| + |a_2| + \dots + |a_p|)^2 \qquad (2.1)$$

Here a's are the autoregression coefficients of the source.

2.3 Some Comparison Between Rate Distortion Theory and Quantiser Codes.

The principal original application of information theory to data compression system design was a demonstration by Gish and Pierce [8] and Goblick and Holsinger [9] that for Gaussian memoryless sources, entropy coded PCM performed within 0.25 bit of theoretical limits at rate 2 and upwards. It is clear that little is to be gained by more sophisticated techniques of coding. Even at rate 1, where entropy coding and simple PCM become essentially the same, the performance gap was similar. Although [9] showed that at low rates simple PCM performed quite near the R(D) lower bounds, however, the gap steadily widened as the rate increased.

The situation is entirely different with Laplacian and Gamma distributed sources. The performance gap between theoretical lower bounds and simple PCM in these cases is very wide even at low rates (see,e.g. Noll and Zelinski [11]). And this gap widens with increasing rate. Entropy coding, which lies about halfway between rate distortion theoretic coding and simple PCM, has given large performance gains over simple PCM at rate 2 and 3 [11] for these distributions. However, the performance gap of 6.76 dB and 3.61 dB for Gamma and Laplacian distributed sources [11], respectively between rate distortion theoretic lower bounds and the simple PCM (and hence entropy coding) at rate 1 has not been exploited.

Sources with memory are of greater importance than memoryless sources since the potential gains of data compression are usually more for the latter as they have more redundancy to remove. Our simulations indicate that at high rates DPCM performs quite well when encoding Gaussian autoregressive sources. Further improvements of the order of

2.8 dB for Gaussian sources and 5.6 dB for Laplacian sources are possible by entropy coding the quantised error output of the DPCM encoder (see O'Neil [12]). But there is no rigorous counterpart of the results of [8, 9] for these correlated sources. Whether such systems perform close to the optimum is an important open problem in information theory. Many sources of interest such as speech are best modelled as Laplacian or Gamma distributed autoregressive sources. Very little is known about what scheme will work well at low rates with these as well as the Gaussian AR(p) sources. the topic of interest to us in this work. While at rate 2, entropy coding is able to reduce the gap between theory and simple DPCM but at rate 1 suitable techniques are yet to be devised to improve the poor performance of existing systems.

2.4 Tree and Trellis Coding

Tree codes have already been introduced in sec. 1.5. To illustrate the trellis structure, consider the constraint length k to be 3 (to make matters simple) in the code tree of sec. 1.5. The branch labels for the unlabelled branches of the tree at level 4 in fig. 1.4 will be determined by the mapping f which will operate on K(=3) inputs which include 2 previous inputs corresponding to level 2 and 3 and 1 input corresponding to level 4. Nodes a and a' in fig. 1.4 specify two paths differing only in the input corresponding to level 1. Since the branch labels at level 4 independent of input at level 1, the branches emanating from nodes a and a' will have the

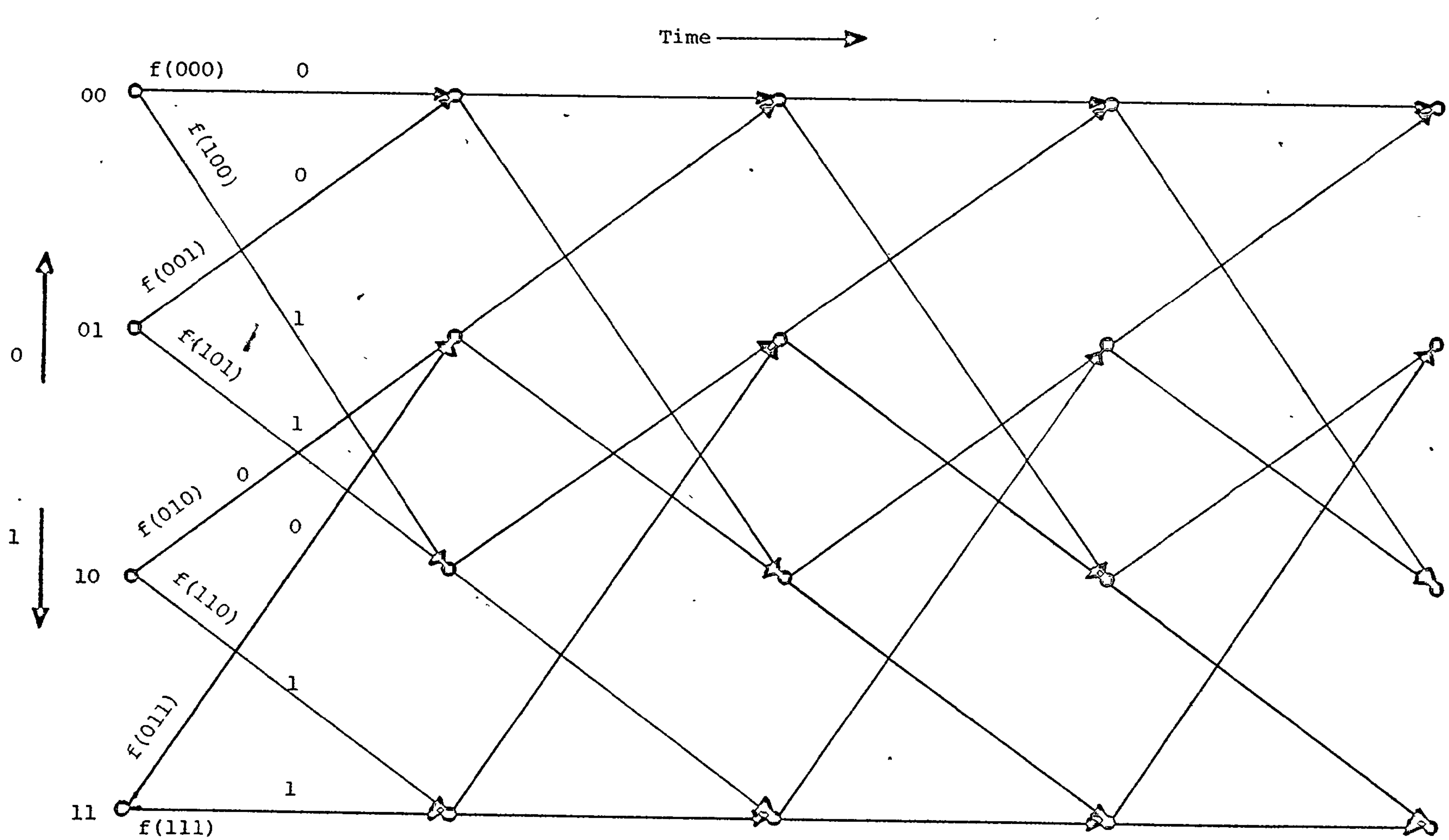

Fig.2.1 A binary trellis matched to a decoder with a constraint length three.

same labels on them. Clearly nodes a and a' can be joined together. Similar argument can be extended to other node pairs as well. The resulting tree diagram with collapsing branches is called a Trellis diagram and the code a trellis code. Fig. 2.1 shows a trellis diagram.

Recently many results have been proved in information theory that show that nearly optimal performance can be achieved by a class of data compression systems called tree or trellis encoding systems. Jelinek [10] broke the traditional block approach (In block coding the encoder accepts at each time a non-overlapping block of source output letters and outputs a non-overlapping block of encoder output digits) of rate distortion theory and proved that codes having a tree structure can achieve the optimal performance with memoryless time-discrete sources. Davis and Hellman [13] further generalized the results of Jelinek. Gallager [18] and Anderson and Jelinek [30] developed algorithms capable of achieving a distortion as close to the lower bounds as desired with tree source codes. Viterbi and Omura [31] and Gray [36] proved corresponding results for trellis source codes. In essence, these results establish the existence of tree or trellis codes which are capable of achieving the optimal performance.

In practice, tree or trellis code design generally involves designing a good linear or nonlinear digital filter (possibly time variant). Once a good code is found, the function of the encoder is to find a good path through this tree (or trellis). In an ideal case, the encoder, with increasing tree depth, must explore an exponentially increasing number of code words in order to find the one that matches the source word best. Therefore, in a practical situation, one has to use a suboptimum method of exploring the tree (or trellis) in a sparse but intelligent manner. It is important to stress that the nature of an efficient search algorithm for tree (or trellis) encoding of source outputs differs fundamentally from that of an efficient search algorithm for sequential channel decoding. The reason is that in sequential decoding, the tree path corresponding to the transmitted word is the only one that is acceptable, whereas in tree (or trellis) encoding of sources there are usually many acceptable paths. Additionally, the search algorithm should be chosen to suit the source. Thus there are two design problems in practical encoders: The code and the search algorithm.

The search algorithm may take one of two basic forms. In a block search, the encoder inputs a block of source symbols, then searches for an encoding that will yield the best reproduction. Once the best path is found, the encoder outputs the whole string and proceeds to the next non-overlapping block. In case of an incremental search, the encoder inputs a string of source symbols and searches for the best path; then the first step along this best path is outputted. The next input symbols is then read in: the best path is found again and the first step is outputted for the second overlapping block.

Though most of the results in rate distortion theory have been arrived at by using a block approach, it has been found through simulation that incremental searches generally produce better results for a given search effort. It should be pointed out as well that the incremental search bears a natural relationship to the tree structure. A tree introduces statistical dependencies between code words in that if two code words share a common path in the lower echelons of the tree, then the corresponding computation need not be repeated. An incremental search capitalizes on this feature of a tree code while the block approach does not.

2.5 Smoothed R(D) Codes

Berger [14] proves the existence of an optimum code with a tree structure by the following theorem.

Theorem 2.3

Let $\{x_t\}$ be a type 1-3 (see Chapter 1) autoregressive source generated by an i.i.d. $N(0, \sigma^2)$ sequence and autoregression coefficient, $a_1, \ldots, a_p$. Let $R(\cdot)$ denote the MSE rate distortion function of $\{x_t\}$ and let Δ denote the essential supremum of $g(\omega)$ defined in Theorem 2.1. Then, given $\varepsilon > 0$ and $D \leq 1/\Delta$ there exists a tree code for $\{x_t\}$ of sufficiently large block length with rate less than $R(D) + \varepsilon$ and MSE per letter distortion less than $D + \varepsilon$.

The corresponding decoder is shown in fig. 2.2. The coefficients $b_0, \ldots, b_p$ are determined by the relationship,

$$|B(z)|^2 = \sigma^2 - D|A(z)|^2 \tag{2.2}$$

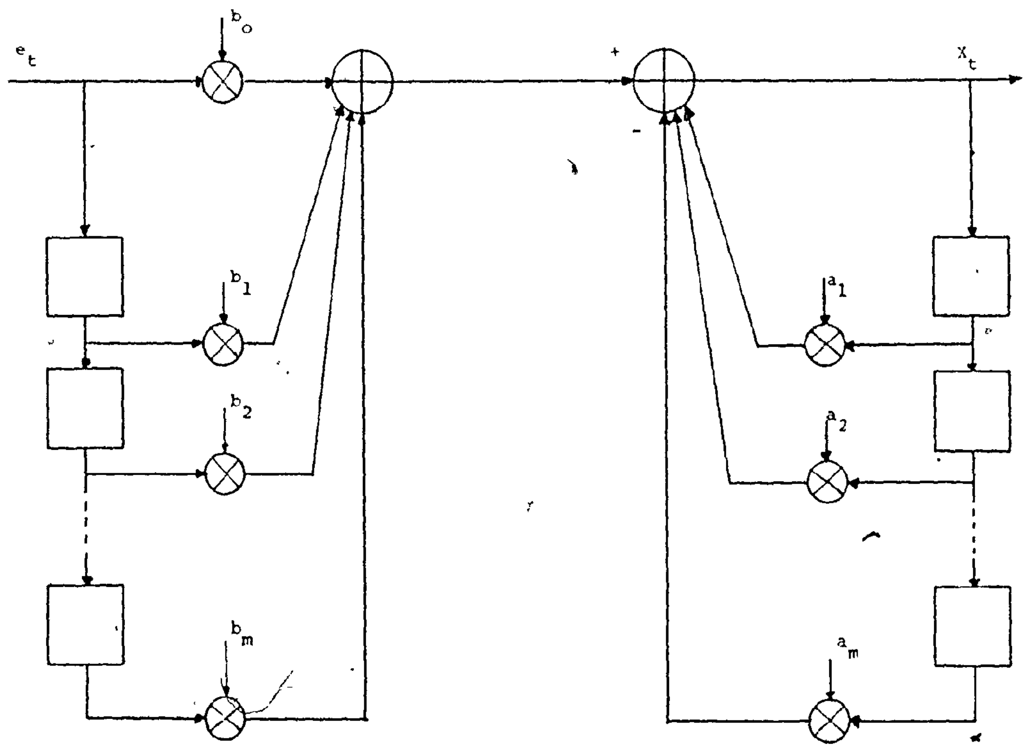

Fig.2.2 Circuit to generate optimal random tree codes for autoregessive sources at high rates.

Here A(z) is the pth order polynomial described by (1.3). By calculating the frequency response, one can see that the b's constitute a smoothing filter. For $D \leq 1/\Delta$, the right-hand side of (2.1) is almost everywhere positive and the Fezer-Riesz representation of non-negative trigonometric polynomials [29] assures us that b's can always be found to satisfy the above relationship. At $D > 1/\Delta$ it is clear that (2.1) fails. We can still conjecture that the decoder filter which is just matched to the source with no 'B' filter may not be optimal at these low rates as well. In Chapter 5, we shall discuss a smoothing filter designed intuitively and investigate its performance.

CHAPTER 3

AN EXPERIMENTAL METHOD AND THE PERFORMANCE OF ORDINARY DPCM

A standard DPCM digitiser is shown in Fig. 1.3. The prediction error e_t is fed to the quantiser and the quantiser outputs a quantised version $e_{q(t)}$ of e_t. Apart from being the final output of the encoder, $e_{q(t)}$ is also the input to the LMS predictor shown in the dotted box in Fig. 1.3, whose next prediction is based on this quantised prediction error $e_{q(t)}$. Clearly if $e_{q(t)}$ is in error because of quantisation noise, the predicted value of the next sample will be suboptimal. When the quantisation is coarse the feedback effect may be severe. In what follows, we shall experimentally analyse the effect of the quantiser non-linearity at low rates when encoding auto-regressive sources with random innovations drawn from both Gaussian and Laplacian distributions.

In a realistic experimental analysis involving probabilistic sources, the concept of confidence intervals is vitally important.

3.1 Confidence Intervals

Before embarking upon the notion of confidence intervals, for the sake of completeness, we shall briefly discuss some of the concepts which shall be used frequently in the ensuing work.

To attach quantitative meaning to the performance of an encoder, we introduce one of the most commonly used measures, the signal-to-quantisation noise ratio,

$$\mathrm{SNR} = \frac{\langle x_t^2 \rangle}{\langle (x_t - \underline{x}_t)^2 \rangle} = \frac{\sum_{k=1}^{K} x_k^2}{\sum_{k=1}^{K} (x_k - \underline{x}_k)^2} \tag{3.1}$$

or in dB

$$\mathrm{SNR\ (dB)} = 10 \ \mathrm{Log}_{10} \ \mathrm{SNR}$$

Here x_t and $\underline{x}_t$ are those shown in Fig. 1.3. We shall denote the above SNR as segmentwise SNR to distinguish it from the sample average SNR (or SNR in brief).

$$\mathrm{SNR} = \frac{1}{N} \sum_{n} \mathrm{SNR}_n \tag{3.2}$$

where SNR_n denotes the n-th segment SNR, and N is the sample size.

Likewise, the estimated sample variance is defined as,

$$\sigma_x^2 = \frac{1}{N} \sum_{n} (\mathrm{SNR}_n - \mathrm{SNR})^2 \tag{3.3}$$

and the standard deviation σ_x as,

$$\sigma_x = \{\frac{1}{N} \sum_{n} (\mathrm{SNR}_n - \mathrm{SNR})^2\}^{1/2} \tag{3.4}$$

When doing experiments with probabilistic sources, the tendency has been to talk about the performance of an encoder in terms of sample averages alone. The sample averages at times may give correct SNR figures but it may happen with highly correlated sources that from experiment to experiment the SNR fluctuates wildly with large deviation on either side about the sample average. Such an average will be largely meaningless from an experimental point unless we can form an estimate of its variance. In general, therefore, we associate a certain interval about a sample average and associate with this interval a likelihood that the outcome of other experiments will stay within this interval. The necessity for such intervals of confidence becomes even more urgent when comparing the performance of two (or more) different encoders whose performances in the sense of sample averages are not too far apart. Statistically speaking, "apart" is meaningful only if considered in conjunction with confidence intervals.

Tchebyceff's Inequality allows us to make a probability statement about an estimator and how it is distributed around a population mean without knowledge of the form of this distribution. One of the most important tools of statistics, the inequality, provides a theoretical basis for when a particular distribution is assumed or justified.

If $\sigma^2 = VAR(x)$ exists and $E(x) = m$, then,

$$P\{|(x - m)| \geq \varepsilon\} \leq \frac{E\{(x - m)^2\}}{\varepsilon^2} = \frac{\sigma^2}{\varepsilon^2} \qquad (3.5)$$

Equation (3.5) is the classical statement of confidence intervals. If x is an estimator of m, then the inequality provides the probability that interval $(x - \varepsilon, x + \varepsilon)$ contains m.

An even stronger statement due to central limit theorem states that,

$$P \left\{ \frac{\left| \sum_n x_n - Nm \right|}{\sigma N^{1/2}} \leq w \right\} = \Phi(w) \qquad (3.6)$$

where ϕ is the Gaussian distribution. This stronger version is valid whenever $\{x_k\}$ are identically distributed and the variance exists [15]. But in our case, there are good reasons to believe that all the moments exist. This implies that even for a relatively small sample size use of CLT is justified. Setting $w = 1$ and $x_n = SNR_n$ in (3.6), we derive the one sigma (one standard deviation) confidence interval. Provided the distribution of $\Sigma x_n/\sqrt{N}$ is close to Gaussian, the average of experiments $\Sigma x_n/N$ lies within $\pm\sigma/\sqrt{N}$ of its true mean with probability ~ 0.68. Although we shall not make use of them, the "two sigma" confidence interval is perhaps more common in statistical analysis: Here the interval is $\pm 2\sigma/\sqrt{N}$, and the average of experiments lies in this range with probability ~ 0.95. In either case, one must have knowledge of σ.

Use of theorem (3.5) or (3.6) clearly requires the knowledge of σ which in turn depends on knowledge of the sampling distribution of an estimator, information we cannot possess. However, σ^2 for a population may be estimated from samples by the unbiased estimator of σ^2,

$$\sigma_x^2 = \frac{1}{N-1} \sum_{j=1}^{N} (x_j - \bar{x})^2$$

If the variance of sampling distribution is not known, and is estimated by σ_x^2 then the statistic

$$t = \frac{\bar{x} - m}{\sigma_x} = \frac{\frac{1}{N} \sum_n x_n - m}{\sigma_x}$$

in equation (3.6) does not follow a strictly normal distribution. The probability statement

$$P \left\{ \frac{\sum_n x_n - mN}{\sigma N^{1/2}} \leq 1 \right\}$$

holds with probability less than 0.68, or saying it another way, at a probability ~ 0.68 and using σ_x instead of σ the confidence limits $\bar{x} \pm \frac{\sigma_x}{\sqrt{N}}$ are too narrow.

The basis for such discussion lies in a few theorems concerning the χ^2-distribution (chi-squared distribution). However, we can overcome most of the objections by having a sufficiently large number of observations (degrees of freedom in chi-square sense) that are independent. An important observation about the distribution of t is that it becomes essentially described by the Normal law with degrees of freedom greater than 120 [16]. For most calculations, however, the significance figures will not be seriously disturbed by use of the theorem (3.6) with an estimate of variance whenever the degrees of freedom are about 20 or more.

3.2 Linear Least Mean Square Prediction

Figure 1.3 shows an m-tap differential PCM encoder. The encoder is fed with a sequence of time-discrete inputs $\{x_t\}$ whose variance is assumed to be unity for convenience. At the same time, the predictor makes an estimate $\hat{x}_t$ of the current input x_t based on previous input to the predictor,

$$\hat{x}_t = \sum_{i=1}^{m} a_i \underline{x}_{t-i} = \sum_{i=1}^{m} a_i(x_{t-i} + q_{t-i}) \tag{3.7}$$

with the resulting prediction error,

$$e_t = x_t - \hat{x}_t = x_t - \sum_{i=1}^{m} a_i(x_{t-i} + q_{t-i}) \tag{3.8}$$

q_t is called the quantisation noise.

Define the signal-to-quantisation noise ratio of a DPCM encoder to be

$$S/N_D = \frac{E\{x_t^2\}}{E\{q_t^2\}} = \frac{1}{\sigma_q^2} = \frac{1}{\sigma^2} \cdot \frac{\sigma^2}{\sigma_q^2}$$

By taking logarithms, one gets

$$SNR = -10 \text{ Log}_{10} \sigma^2 + 10 \text{ Log}_{10} \frac{\sigma^2}{\sigma_q^2}$$

Here $\sigma^2 = E\{e_t^2\}$ and $\sigma_q^2 = \{q_t^2\}$.

The total SNR can be expressed as

$$SNR = SNI + SNR_q \tag{3.9}$$

where SNI is called the signal-to-noise improvement ratio. Ideally, this is the gain obtained by DPCM over PCM when encoding sources with memory. SNR_q is the SNR obtained by a simple PCM system when used to encode the sequence $\{e_t\}$.

From equation (3.8) the prediction error power becomes,

$$\sigma^2 = E\{(x_t - \hat{x}_t)^2\} \simeq E\{(x_t - \sum_{i=1}^{m} a_i x_{t-i})^2\} \tag{3.10}$$

where the equality becomes exact in the limit of fine quantisation.

In matrix notation, we can rewrite (3.10) as,

$$\sigma^2 = 1 - 2A^T G + A^T R A \tag{3.11}$$

where,

$$A = \begin{bmatrix} a_1 \\ a_2 \\ \cdot \\ \cdot \\ a_m \end{bmatrix} \qquad G = \begin{bmatrix} r_1 \\ r_2 \\ \cdot \\ \cdot \\ r_m \end{bmatrix}$$

$$R = \begin{bmatrix} 1 & r_1 & r_2 & \cdot & \cdot & \cdot & \cdot & r_{m-1} \\ r_1 & 1 & r_1 & \cdot & \cdot & \cdot & \cdot & r_{m-2} \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & 1 & r_1 \\ r_{m-1} & r_{m-2} & r_{m-3} & \cdot & \cdot & \cdot & r_1 & 1 \end{bmatrix} \qquad (3.12)$$

Here $r_k = E\{x_t \cdot x_{t-k}\}$ is the auto-correlation coefficient of sequence $\{x_t\}$ at the kth lag.

Minimisation of σ^2 with respect to A yields

$$A_{opt} = R^{-1} G \qquad (3.13)$$

$$\& \qquad \sigma^2_{min} = 1 - A^T_{opt} G = 1 - G^T R^{-1} G \qquad (3.14)$$

When the coefficients $\{a_i\}$ are chosen to satisfy (3.13), we call the predictor "matched" to the source. Any other values of $\{a_i\}$ will give rise to a "mismatched" source-predictor pair, and a lower SNR.

Clearly the above results deduced by assuming that the quantisation noise is negligible will not hold at low rates. The quantisation noise will be fairly substantial at rates 1 and 2, too much so for it to be ignored in comparison with x_t. The problem of DPCM encoder performance at these low rates has not been solved analytically. Our approach in this work will be to investigate the DPCM encoder at these low rates through experimental analysis when (3.9) & (3.10) do not hold, for the case of Normal as well as Laplacian distributions.

Having collected sufficient data, we shall compare them with what we should have obtained had the above simplification been true.

3.3 Optimum Quantiser

For our purpose, we shall refrain from discussing the optimality of a quantiser and rather refer the reader to available literature on the subject. Normally distributed memoryless sources have been analysed by Max [20], while quantisation of Laplacian and Gamma distributed sources has been discussed by Paez and Glisson [21]. Optimum Quantiser for general auto-regressive sources is a question still very much to be answered. However, first order auto-regressive sources have been dealt with by D. Arnstein [22] at low rates.

Our approach in this work will be to find the optimum quantiser through simulation and rather attack the other aspect of the problem, i.e., to investigate the extent of performance degradation caused by the optimum quantiser at low rates and the means of redressing it in the predictive loop of DPCM.

3.4 Experimental Design

Broadly speaking, the experimental setup involves three major steps, as shown in Fig. 3.1:

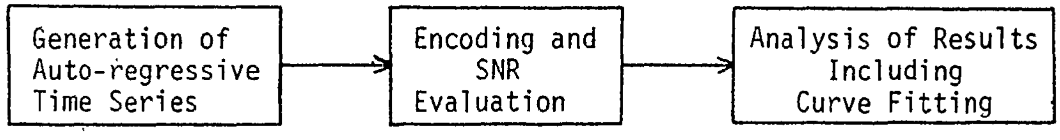

Fig. 3.1 The Experiments Involve Three Major Steps

Generation of auto-regressive time series: An auto-regressive time series is generated by passing the random deviates of appropriate distribution through the colouring filter of Fig. 1.1 with autoregression coefficients from one to three taps chosen to span the range of autocorrelation. Gaussian random deviates N(0,1) are generated using the IMSL subroutine GGNMP [32] while the Laplacian deviates are drawn from the distribution $L(\mu = 0, \sigma^2 = 1)$

$$P_L(x) = \frac{\alpha}{2} e^{-\alpha|x|} \tag{3.15}$$

where $\sigma = \frac{\sqrt{2}}{\alpha}$

With α so chosen, the resulting shape of the distribution is believed to closely resemble the distribution of speech. The random deviates with such a distribution are obtained through appropriate transformation of uniformly distributed random deviates which are generated using the IMSL subroutine GGUBS [32].

The important issues at this stage are, firstly, how many of random deviates should make up a segment and secondly, what should be the size N of the sample. The answer to the first is quite straight forward. The segment length should be large enough so that the sample average SNR remains more or less fixed as we vary the sample size N. A small segment length tends to inflate the overall sample average. The reason for this becomes quite obvious if we consider an extreme case where each segment consists of only one source letter. In this situation, it is highly likely that the value of some particular source letter comes very close to a quantiser output, reducing the quantising noise to a very small value and in effect producing an extremely high SNR for that segment of one source letter. This high value will tend to swamp all other segment SNR's, causing the overall sample average to appear much higher.

Based on these considerations, we shall use segments of length 1000 source letters. Furthermore, in order to keep the statistics manageable, i.e., to be able to use central limit theorem as discussed in sec. 3.1 to calculate the confidence intervals, we shall use a segmentwise SNR sample size N = 20.

Encoding: Statistical consideration of sec. 3.1 clearly requires the independence of each segment SNR. Therefore, the encoding is performed using a run of 1000 source letters at a time and the DPCM encoder of Fig. 1.3. The encoding performance is evaluated in terms of SNR as defined by (3.1). All the initial conditions are set to zero before starting the encoding of the next segment. (This is particularly important for experiments in coming Chapters). When performing experiments with matched source-predictor pairs, the predictor taps are chosen the same as the auto-regression coefficients of the source undergoing encoding.

The best quantiser step-size is found by performing the encoding at different step sizes, fitting a SNR vs. size curve, and taking the best SNR so obtained as the SNR corresponding to the optimum quantiser. When working with Normally distributed sources, we employ the uniform quantiser, since the performance gap between the uniform quantiser and the optimal non-uniform quantiser is very small. When dealing with Laplacian distribution, we shall find the optimal non-uniform quantiser, since the gap between uniform and optimal non-uniform quantiser SNR performance in this case is significant.

Curve fitting: Generally, this involves finding an appropriate source-predictor statistic which is likely to have played an important part in a particular situation and plotting that particular performance criterion as a function of this statistic for various sources. This is done to detect an underlying law in the results. However, this is only possible if some pattern can be made out of the results. Wherever possible at least an attempt will be made.

3.5 Results: The SNR Loss

In most cases, the SNR obtained through simulations fell well short of (SNR_q + SNI) expected at high rates from (3.9). We shall define the SNR loss caused by the quantiser non-linearity at low rates as (SNR - (SNR_q + SNI)); i.e., the SNR obtained minus the SNR that we should have achieved were it not for the quantiser.

To make the results more meaningful, an attempt will be made to fit a curve to the loss as a function of an appropriately chosen source statistic, since the performance degradation varied a great deal from one source to another. The most obvious course would be to look for a relation between this loss and SNI. But we could find

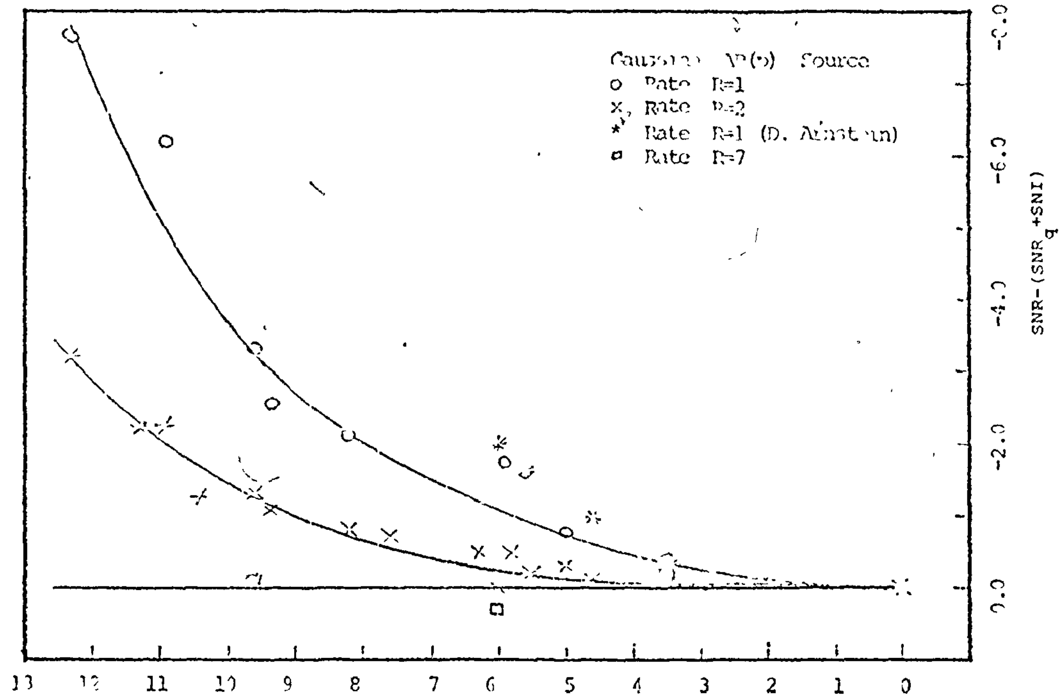

Fig.3.1 DPCM SNR Loss as a funcion of $-Log_{10}(1/\Delta)$

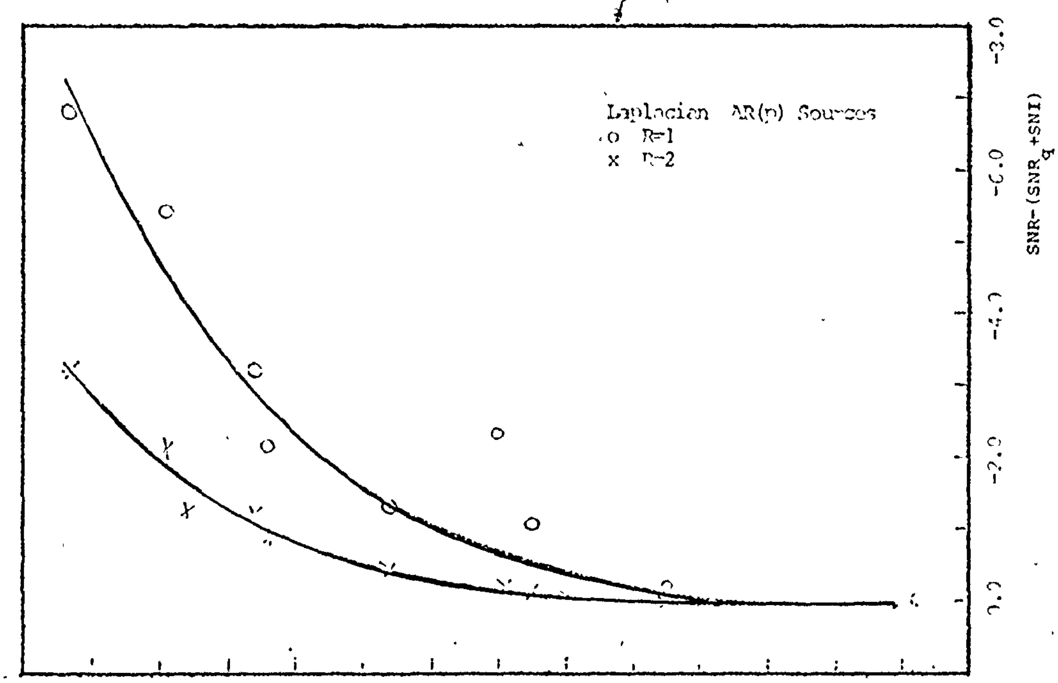

Fig.3.2 DPCM SNR Loss as a function of $-Log_{10}(1/\Delta)$

no definitive pattern in the results for various sources.

Next, the SNR loss was plotted as a function of 10 Log_{10} $\{1/\Delta\}$ the critical distortion in Gray's theorem [see Sec. 2.2]. The results are shown, for both Gaussian as well as Laplacian sources in Fig. 3.2 and Fig. 3.3, respectively. The fitted curve follows the plotted points well. To check the goodness of fit, the points corresponding to D. Arnstein's [22] single tap auto-regressive sources are also shown for Gaussian distribution. These points support the presence of strong secondary effects that the fitted curve does not account for, since the points are too wildly scattered about it. (The one sigma confidence intervals are small, too small to be shown well on the plot.)

The above criterion appeared a little too complicated for this particular situation, therefore some simpler function of AR taps deserves attention. Another likely source statistic is $A^T A = \sum_{k=1}^{P} a_k^2$. The resulting plotted points and the fitted curve are shown in Fig. 3.4 and Fig. 3.5 for Gaussian and Laplacian distributed sources respectively. Theoretically calculated points from D. Arnstein [22] for one tap Gaussian auto-regressive sources are also shown and these seem to agree well with the fitted curve.

The plots for rates 1 and 2 diverge from the rate 7 plots when the correlation between successive source letters increases. The performance drop at rate 1 for large $A^T A$ is rather substantial. Thus it is not unreasonable to expect that proper source encoding strategy should give a proportionately large SNR improvement with tightly correlated sources. Sources like speech, which often exhibit very

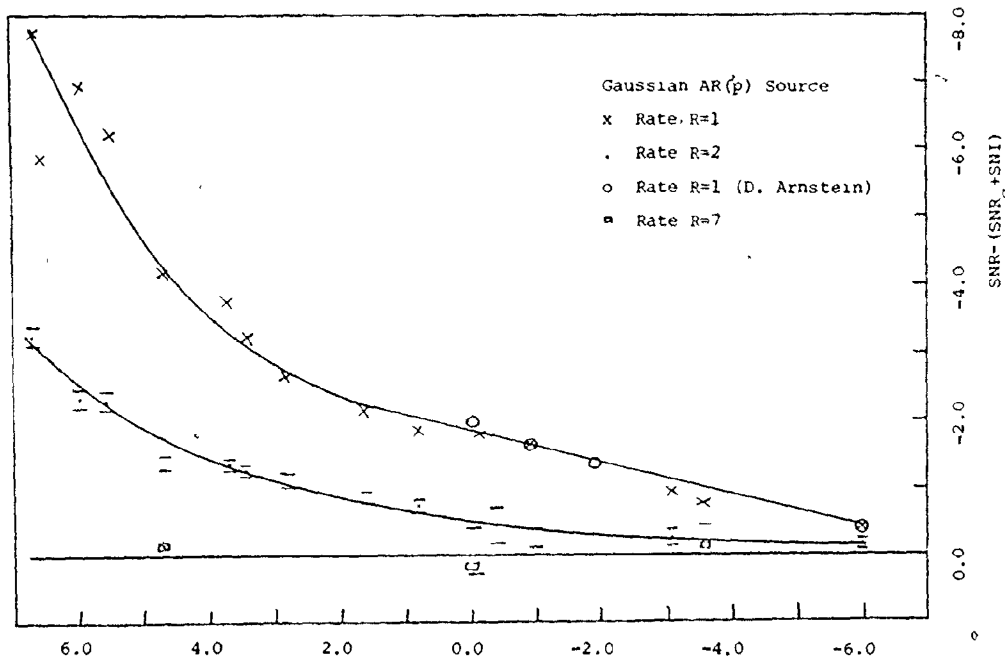

Fig.3.3 DPCM SNR Loss as a function of $10\mathrm{Log}_{10}(\sum_{i=1}^{p} a_i^2)$

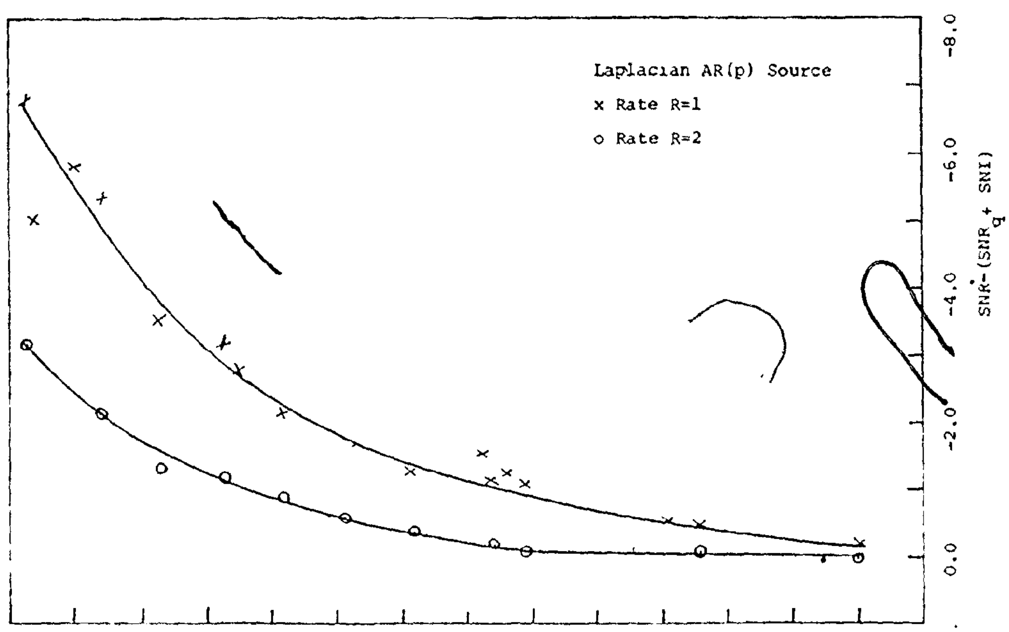

Fig.3.4 DPCM SNR Loss as a function of $10\mathrm{Log}_{10}(\sum_{i=1}^{p} a_i^2)$

tight correlation are probably not encoded very efficiently at low rates by even the best adaptive DPCM systems. Another rather unexpected result is that the results do not discriminate in a substantial way between Laplacian and Gaussian distributions, keeping in mind the limitations of the experimental setup. However, the loss appeared less in case of Laplacian sources by about one dB at very tight correlation.

CHAPTER 4

TREE ENCODING OF AUTOREGRESSIVE SOURCES

We discussed in Chapter 1 that the output of traditional DPCM code generator can be mapped onto a tree-like structure shown in fig. 1.4. However, the traditional DPCM code generator pursues only a single path through this tree. In a superior approach the encoder performs a multipath search. Numerous search algorithms are known which pursue several paths through the tree at any time and choose among them at a later point. In the ensuing work, we shall use such a tree search algorithm as the encoder along with the traditional DPCM encoder to encode autoregressive sources.

4.1 Implications of Previous Work

Fuelled by rapid strides made by the rate distortion theory, there has been a recent interest in the application of tree encoding to sources like speech and images. Real speech is a source whose exact distribution is not known and which exhibits quasi-stationarity. The early impetus was provided by a paper by Anderson and Bodie [23] who used a tree search algorithm along with traditional DPCM encoder of fig. 1.3 to encode speech at rates of 1 and 2 bits per source letter. The scheme being non-adaptive did not take into account the quasi-stationary nature of the source. With the quantiser step-size fixed, it did not correct for local power variations in speech. Yet, a moderate degree of search yielded a SNR improvement of the order of 3 dB over the traditional DPCM.

The SNR improvement due to multipath search was attributed to among other things, mismatched source predictor pair on a local basis or an unmatched stepsize. The last possibility was discounted by Jayant and Christensen [24] who studied a fixed predictor and an adaptive quantiser along with tree searching and still reported a comparable gain. Wilson and Hussain [25] used an adaptive predictor/quantiser scheme and reported a similar gain at comparable intensity of search. Most recently, Chan and Anderson [26] applied M-algorithm along with maximum entropy method (MEM) to compute a new predictor every 64 ms, a short duration designed to account for the quasi-stationarity of speech, and they reported SNR gain over traditional DPCM ranging from 1.5 to 2.5 dB. These last two studies tend to agree against the gain coming from a mismatched source and predictor.

There has been a good deal of speculation as to the origin of this quite substantial performance improvement at a rather moderate search effort. In non-adaptive schemes, the improvement could be attributed to non-stationarity of speech. But in the case of adaptive schemes, the improvement has been attributed to many different causes including the quality of adaptation and the distribution of speech, which resembles the Laplacian distribution more closely than the designed for Gaussian distribution. But the recent work of Chan and Anderson questions the belief that this relatively large gain is due to a local mismatch between the source and predictor. However, several questions remain unanswered.

The results of Chapter 3 are rather illuminating in this regard. These results establish beyond any reasonable doubt that the quantiser in the predictive loop of DPCM at low rates does lower the performance considerably below that expected from estimation theory and raises many questions about the effectiveness of traditional DPCM at these low rates even with a perfectly matched source-predictor pair when the source is tightly correlated. The possibility exists that tree-searching might fix a fair amount of damage done by the quantiser non-linearity in the predictive loop of DPCM. This would be quite consistent with the results of many authors, including Chan and Anderson, whose results indicate that SNR improvement due to tree searching is largely independent of adaptation.

To put all these speculations to rest and locate the real origin of the performance improvement, in what follows we shall discuss and apply a tree search algorithm to this end. The algorithm along with the traditional DPCM code generator will be used to encode autoregressive sources at low rates as well as high rates. The performance of such a scheme will be evaluated in detail for the case when predictor is matched to the source as well as the case when predictor is mismatched to the source in order to find if tree searching recovers the performance degradation due to mismatched source predictor and to what extent. Furthermore, autoregressive sources will be generated by using random deviates drawn from both Gaussian and Laplacian distributions to determine the role of the distribution.

4.2 Tree Encoding

Traditional DPCM penetrates the code tree through a single path approach. The code generator outputs the final output digits (corresponding to a tree branch) on the basis of previous branching decisions and the current input to the encoder. In this single path search the encoder retains only one path through the tree and decisions are made instantaneously without any regard for the future behaviour of the path. Clearly the encoder can make more efficient use of information available to it if it can delay its final decision for the time being, retain all the options available to it at a given level in the tree, and choose eventually a path (code word) that satisfies the appropriate fidelity criterion best. Since such an exhaustive search will find the best path through a tree while the traditional encoder finds a path (some other) through the tree, the performance of the new system cannot be worse than that of the traditional system and actually a significant gain may result. Even if only a few good paths are retained at a given level instead of all the possibilities, performance can only improve. However, the overall improvement might be reduced compared to the exhaustive case.

4.3 M-algorithm

Many effective algorithms are known, each differing from the others in how paths are retained or dropped. Accordingly, their effectiveness varies. Some are more effective at lower intensity of search while others are more effective at higher search intensity and of course the source plays its part too in deciding which particular algorithm is more suited to it. At low intensity of search, the M-algorithm, with its modest complexity and synchronous operation has been found quite effective when encoding speech. Since in this work we are more interested in the effectiveness of tree searching rather than the merits of a particular algorithm that accomplishes it, we will not go into great deal of detail about search algorithms. We choose M-algorithm as a benchmark degree of searching.

Though it is not uncommon to find detailed description of M-algorithm in literature, we shall briefly describe it to achieve a measure of self-containedness.

Being a breadth-first algorithm, the M-algorithm views all the branches it will ever view at a given tree-level before penetrating any further into the tree. The scheme pursues a fixed number M of tree-paths at any given tree-level. At each level of the tree, all the bM branches are extended out of the M saved paths from the previous level, and only the best M of these bM extended paths (b is the number of branches out of a node) are saved for the next level. The M-algorithm (as do all other breadth-first algorithms) operate synchronously; i.e., the lengths of all the retained paths at any given instant are the same.

In the steady-state operation of the algorithm, this common path length is called the memory-length L of the search and in practice is required to be a small number due to memory constraints. The pathmap symbols can vary over this length. At the Nth source letter input to the encoding algorithm (N-L) becomes the point of decisions among path alternatives leading to level N. Here path-map symbols corresponding to the (N-L)th branch are released as final output. This implies that the algorithm delays its decision by L source letters after it has received an input source letter.

4.4 Experimental Setup

Much of the material discussed in the experimental setup of previous chapters carries on to the present case with the following additions and modifications.

Unless indicated to the contrary, throughout this thesis we shall use M = 4 and L = 32. It has been reported that M = 4 extracts much of gain that can be had from M-algorithm and our work will only strengthen this point. We do have a few runs using M = 8 and it is clear that the SNR improvement achieved in going from M = 4 to M = 8 is only a small fraction of that achieved in going from M = 1 to 4. L = 32 is larger than common in applications, but when encoding a source with very tight correlation between successive letters, it is essential to have a sufficiently large memory length.

In such a situation, it takes quite a while for the source letter correlation to die out effectively so that a final decision can be made by the algorithm. A premature turncation of memory length in these cases will adversely affect the performance of the search. This follows from the fact that a final decision at a particular level (N-L) in the tree has already been made while the effect of this decision is still felt in the upcoming branches (> N).

Now turning to autoregressive sources, which include both with Laplacian as well as Gaussian distributions, we encode in exactly the same fashion as in sec. 3.4. We employ again a sample size of 20 with each segment consisting of 1000 source letters and one sigma confidence intervals.

Furthermore, to facilitate comparison with other works which are relevant to this work, the well known signal-to-quantisation noise ratio as defined in section 3.1, will be used as the metric for the search algorithm. The SNR so defined is equivalent to the cumulative squared error. The metric will be independent across runs, i.e., the search algorithm will find a path of length 1000 in the tree which has the greatest metric and then set the metric back to zero and start afresh to encode a new segment of 1000 source letters. This is essential in a statistical sense as discussed in sec. 3.1; the segmentwise SNR's that constitute the sample SNR's must be independent.

Since the SNR improvement involves the difference between the SNR's with and without searching, two distinct statistical quantities appear with different standard deviations. The confidence interval for the difference (the SNR improvement) is simply the sum of confidence intervals of the two quantities.

4.5 Analysis of Results

From a number of experiments with varying degrees of AR order

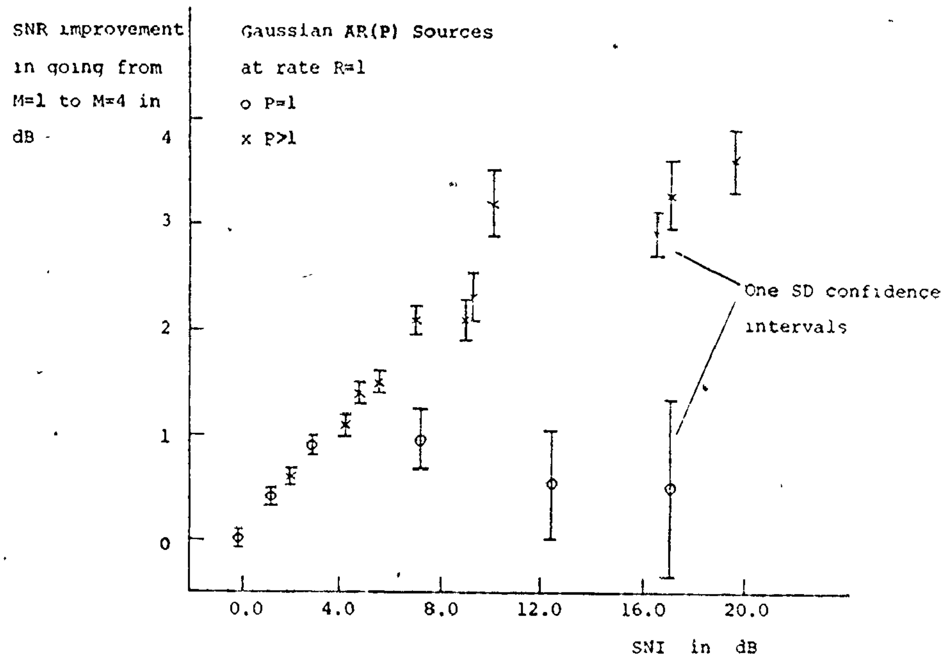

Fig 4.1 SNR improvement as a function of SNI

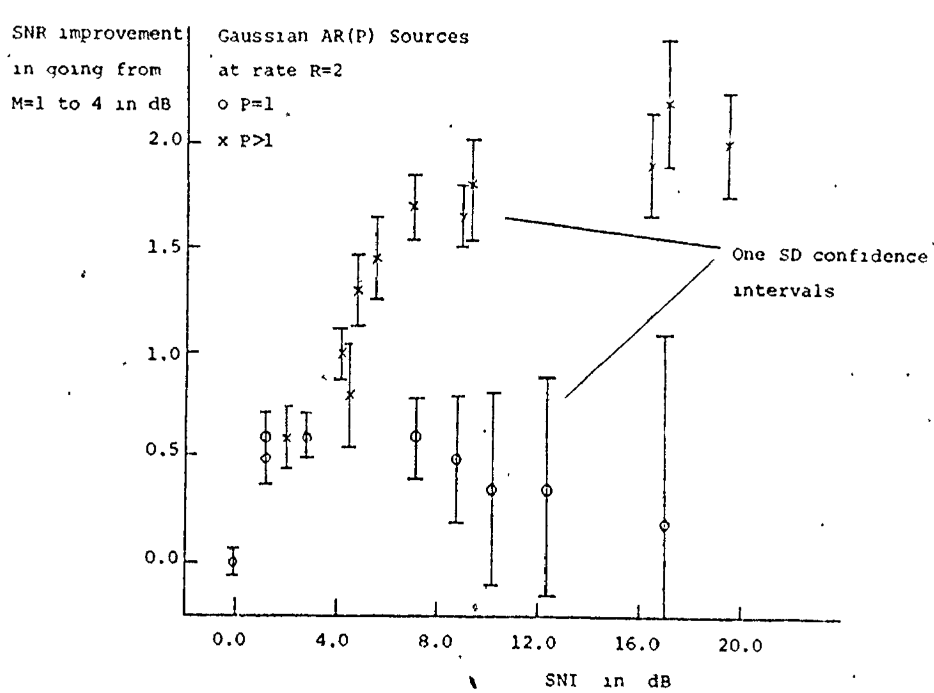

Fig 4.2. SNR imrovement as a function of SNI

and source letter correlation it was not difficult to see a pattern in the results. The SNR performance improvements due to tree searching seems to increase with source letter correlation. Now it remains to see if an appropriate source statistic can be found that plays a crucial part in determining the amount of gain due to tree searching.

Again, the simplest and most obvious choice appears to be the SNI. The resulting plotted points and associated one sigma confidence intervals are shown in fig. 4.1 and fig. 4.2 for Gaussian autoregressive sources at rates 1 and 2 bits per source letter. It is interesting to note that points corresponding to the first order sources split apart from the rest in both cases.

Another possibility is the eigenvalue spread of the autocorrelation matrix of the source:

$$R_x = \begin{bmatrix} 1 & r_1 & r_2 & & r_p \\ r_1 & 1 & r_1 & & r_{p-1} \\ \cdot & & & & \cdot \\ \cdot & & & & \cdot \\ \cdot & & & & \cdot \\ r_p & r_{p-1} & r_{p-2} & & 1 \end{bmatrix} \tag{4.1}$$

where P is the order of the source and $\{r_k\}$ are the autocorrelation coefficients defined in sec. 3.2. The eigenvalue spread plays a vital role in deciding the speed of convergence of adaptive algorithms such as LMS, Goddard's algorithm (Kalman Filter), etc.; the narrower the spread, the faster the convergence. It is an accepted measure of ill-

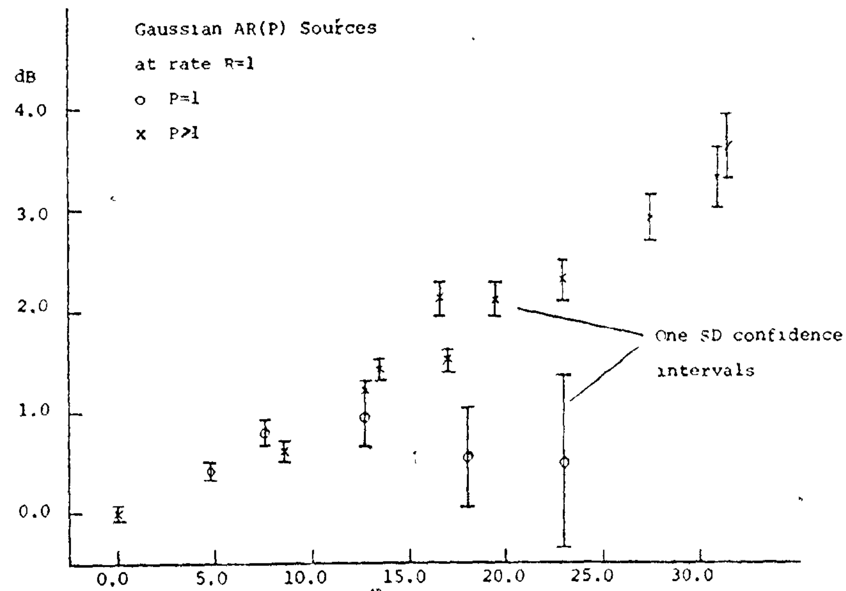

Fig 4.3 SNR improvement as a function of eigen-value spread $10\text{Log}_{10}(\lambda_{max}/\lambda_{min})$ of the auto-correlation matrix of autoregressive sources

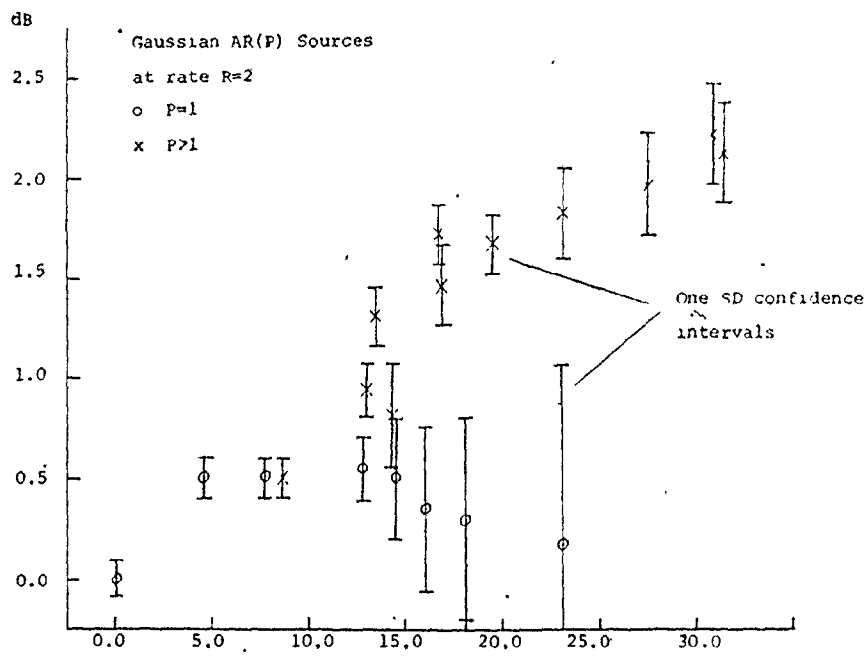

Fig 4.4 SNR improvement as a function of eigen-value spread $10\text{Log}_{10}(\lambda_{max}/\lambda_{min})$ of auto-correlation matrix of autoregressive sources

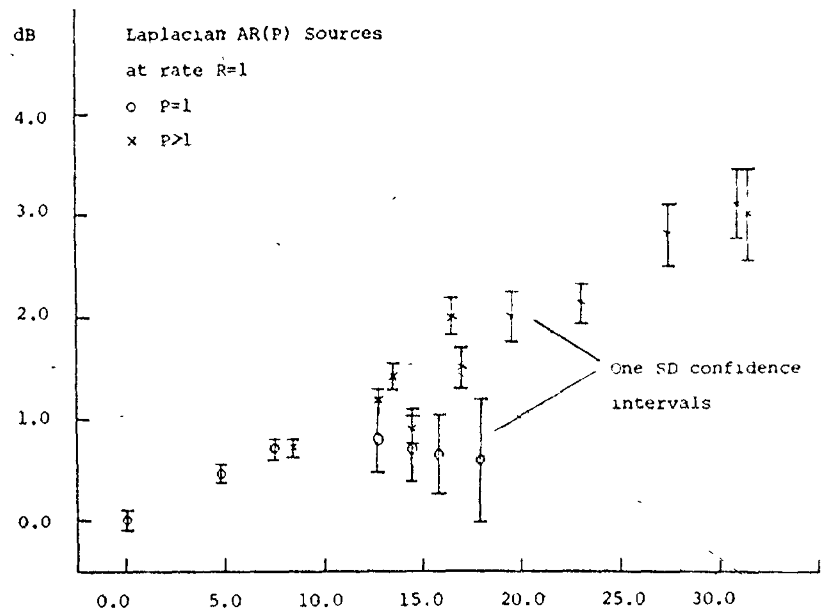

Fig 4.5 SNR improvement as a function of eigen-value spread $10\log_{10}(\lambda_{max}/\lambda_{min})$ of auto-correlation matrix of autoregressive sources

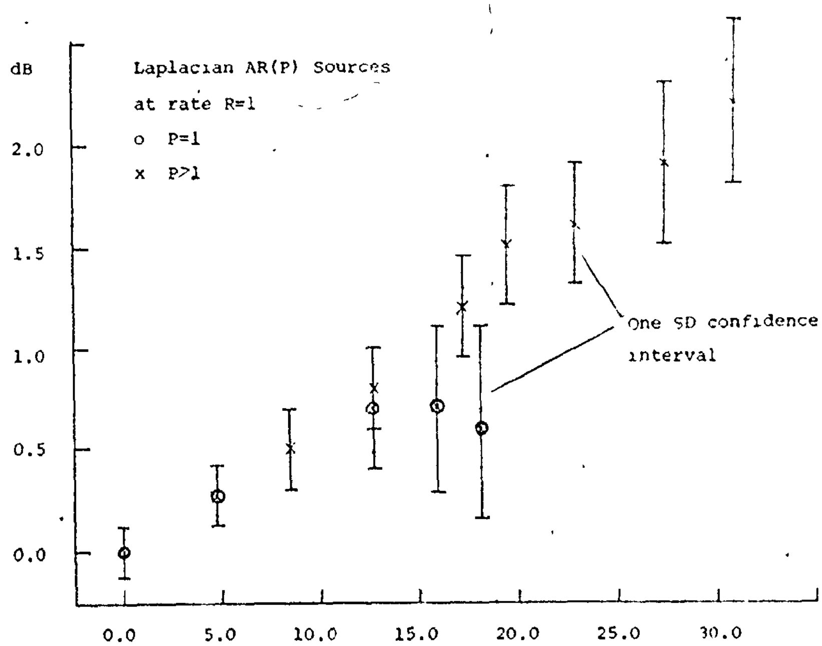

Fig 4.6 SNR improvement as a function of eigen-value spread $10\log_{10}(\lambda_{max}/\lambda_{min})$ of auto-correlation matrix of autoregressive sources

conditioning of the autocorrelation matrix of a source [27].

A word of caution here is essential since the eigenvalues that are being considered here are not exactly the same as those that come across in rate distortion theory. In this later situation, the correlation matrix is of infinite order or at least of very high order and the spread is the ratio of supremum to infimum of the $(\text{spectrum})^{-1}$ g(w) considered in sec. 2.2. The eigenvalues of a pth order autocorrelation matrix under consideration approaches the later ones in the limit of large P [29].

Let λ_{max} and λ_{min} stand for the largest and smallest eigenvalues respectively of the pth order autocorrelation matrix (4.1), then the eigenvalue spread p is defined in dB as,

$$p = 10 \text{ Log}_{10} \; [\lambda_{max}/\lambda_{min}] \qquad (4.2)$$

Fig. 4.3 and fig. 4.4 shows corresponding plotted points for Gaussian autoregressive sources at rate 1 and 2 bits per source letter. The respective plotted points for Laplacian AR(P) sources are shown in fig. 4.5 and fig. 4.6. Each plotted point has associated with it the appropriate one standard deviation confidence interval. Again, the SNR improvements for single tap sources split apart from the rest which represent multi-tap sources.

Another promising source statistic that ought to be considered is the critical distortion in Gray's theorem discussed in sec. 2.2. Fig. 4.7 and fig. 4.8 shows the respective plotted points and the fitted curve for Gaussian and Laplacian sources and the associated

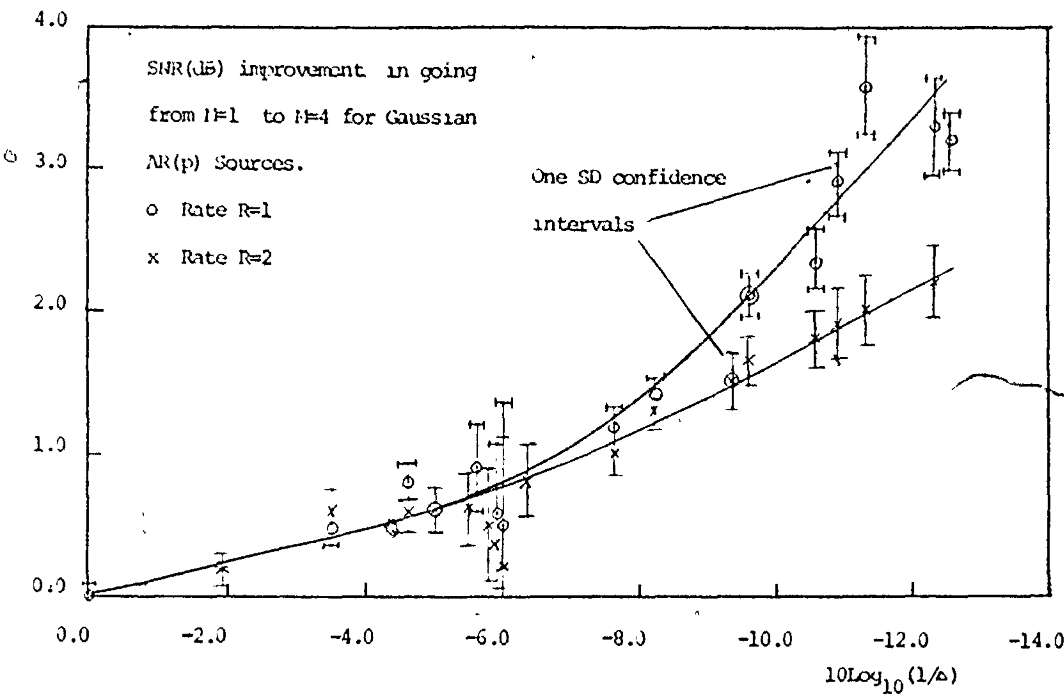

Fig. 4.7 SNR improvement achieved by tree searching with the DPCM system

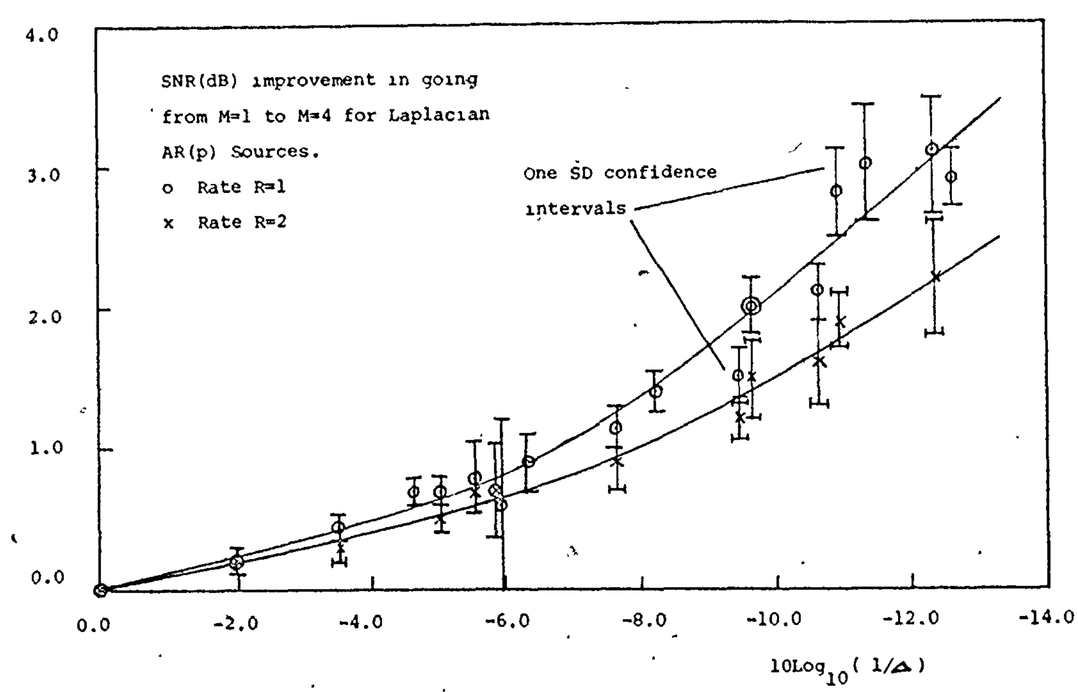

Fig.4.8 SNR improvement achieved by tree searching with the DPCM system

confidence intervals. Fig. 4.7 also shows a few points corresponding to the rate 7. Agreement is now good for all sources.

4.6 Mismatched Source-Predictor

There are situations in real life when we don't know the source exactly, such as in speech encoding, which naturally gives rise to mismatched source-predictor pairs. In order to investigate the effect of tree searching in such situations, we also encoded sources with a mismatched predictor. The predictor is fixed to be the "Butterworth" filter with AR taps 1.231, -0.625, 0.119 in each case; and we varied the source. Fig. 4.9 shows the case when the source is tightly correlated and is of order 3 at rate 7. Fig. 4.10 and 4.11 shows the performance of Butterworth filter predictor when used to encode two single tap sources, one lightly correlated and the other tightly correlated. In either case, tree searching does not give much more than a token improvement. The corresponding cases for rate 2 are shown in fig. 5.3 to 5.8 of the following chapter. In this case, the improvement is more pronounced but nothing of the nature of removing or fixing the mismatch.

One major drawback of the source statistics considered thus far, apart from SNI, is that they are meaningful only in the situations when the predictor is matched to the source. The SNI, which does take account of source as well as the predictor structure, has not been particularly useful in predicting searching gain. Therefore, it becomes highly desirable to seek a source statistic through which we

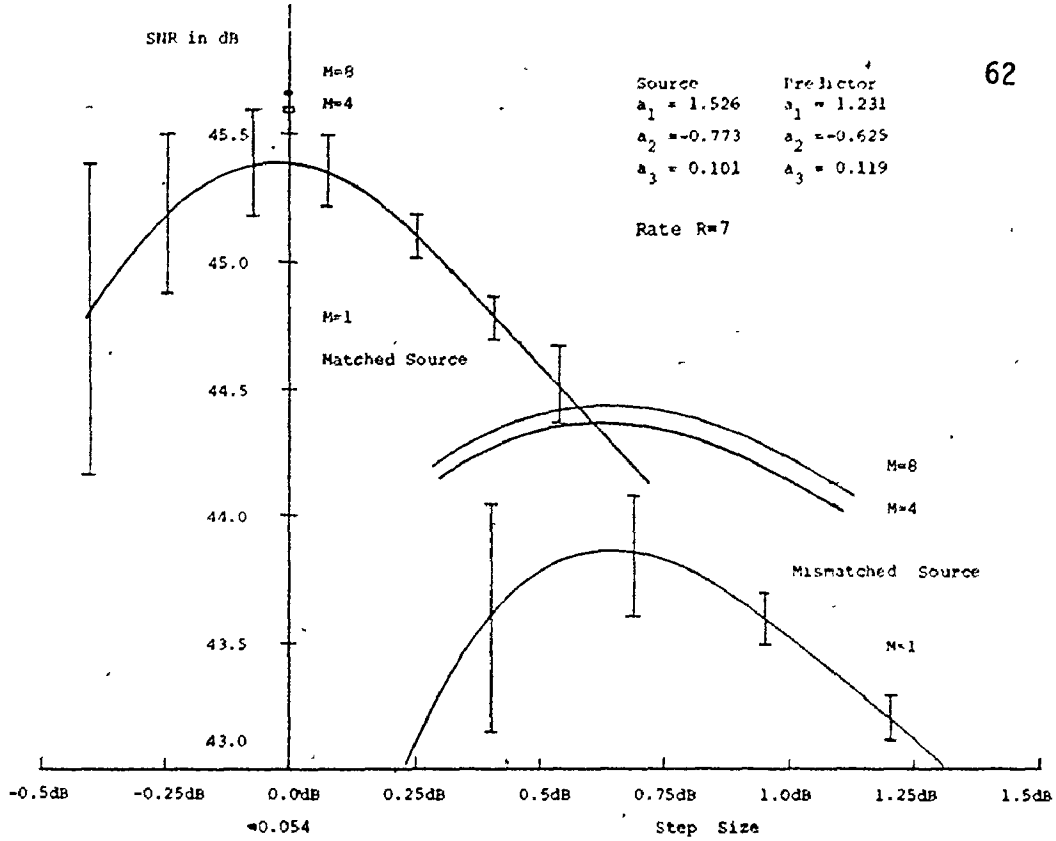

Fig.4.9 SNR performance of DPCM encoder with a matched and mismatched predictor with and without tree searching at high rates.

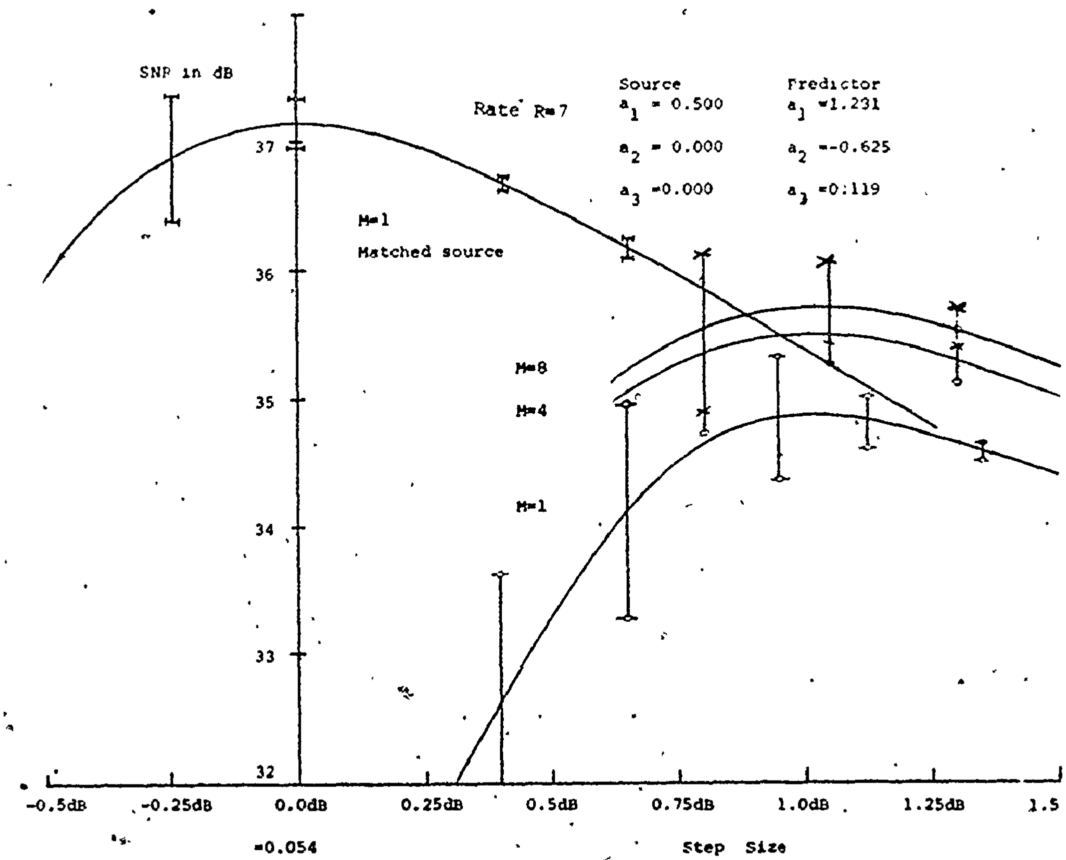

Fig. 4.10 SNR performance of DPCM encoder with a matched and mismatched predictor with and without tree searching at high rates.

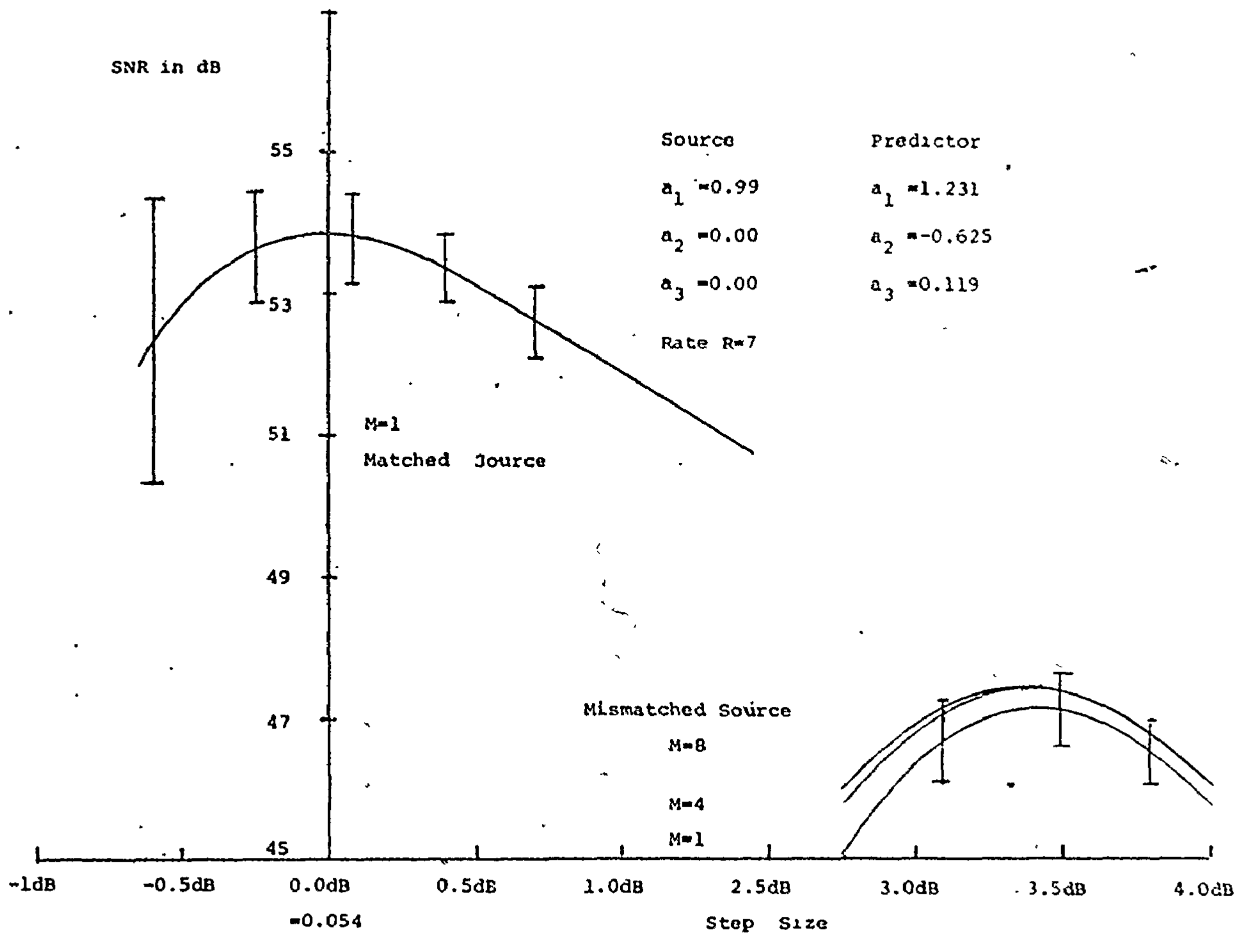

Fig.4.11 SNR performance of DPCM encoder with a matched and mismatched predictor with and without tree searching at high rates.

can unify the SNR improvement that can be achieved when the predictor is matched to the source as well as when it is mismatched.

One such statistic is our SNR loss as defined in sec. 3.5 at low rates. The SNR improvement as a function of SNR loss is shown in fig. 4.12 and 4.13 for rate 1 and 2. It is interesting to note that both curves hold irrespective of the distribution. It is also worth pointing out that fit of the curve for rate 2 is rather excellent; i.e., almost every point that we tried fell close to the fitted curve, regardless of distribution. Furthermore, as far as mismatched source-predictor pairs are concerned, the statistic has failed to deliver. The SNR improvements in such situations do not follow the curves even approximately.

4.7 General Discussion

Examination of plots clearly indicates that we gained most through tree searching where the traditional DPCM predictive quantiser did the most damage, something that we anticipated when discussing the results of Chapter 3. Clearly, tree searching of moderate intensity has had a considerable success in fixing a substantial part of the damage done by quantiser non-linearity in the predictive loop of DPCM. The SNR improvement is quite substantial both at rate 1 and 2. Furthermore, considering the statistical nature of results, the SNR improvement appears to be a very well-behaved function of the critical distortion and the DPCM SNR loss.

The results obtained through these controlled experimentations

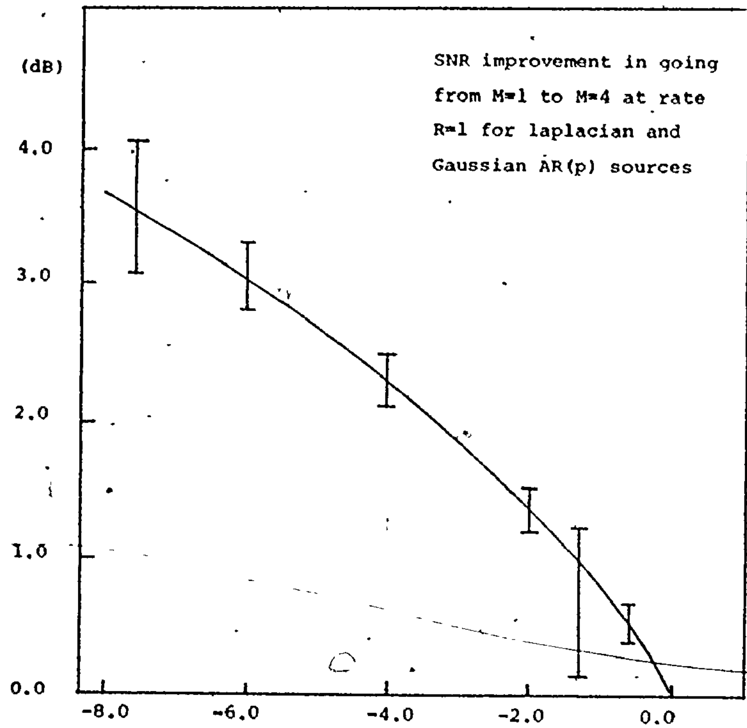

Fig.4.12 DPCM SNR gain with tree searching as a function of DPCM SNR Loss.

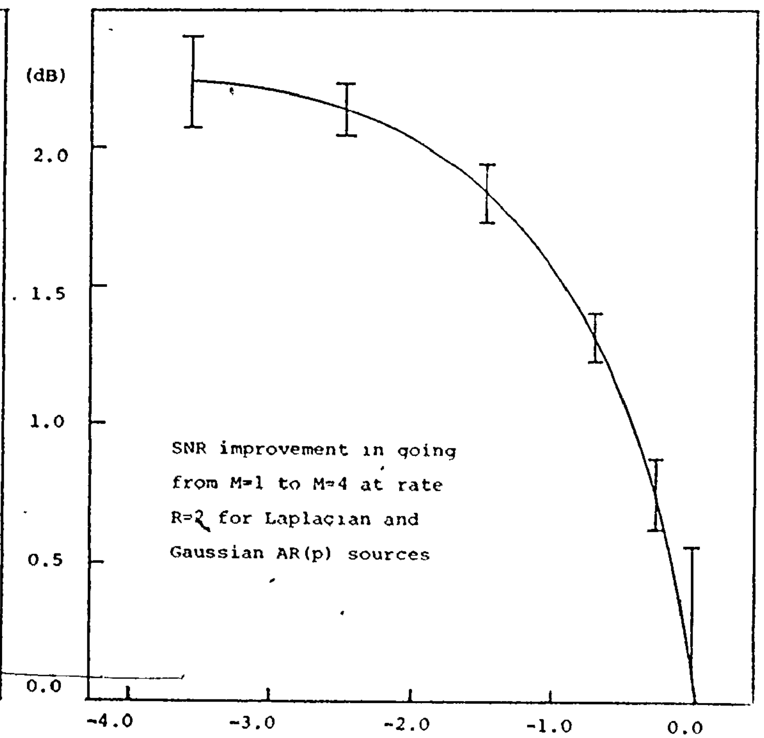

Fig.4.13 DPCM SNR gain with tree searching as a function of DPCM SNR Loss

confirm the SNR improvements obtained by other authors at comparable search intensity with speech. The results also establish the indispensibility of tree searching when working at low rates with autoregressive sources even when the predictor is perfectly matched to the source. Thus, the opinion expressed by Chan and Anderson is completely justified: That the SNR improvements available from adapting and tree searching are independent, at least for speech modelled as an autoregressive source. Also, the role of distribution in determining the gain due to tree searching seems to be quite limited at these low rates.

Furthermore, we see no sign of widely held belief that tree searching can recover to any substantial extent the loss caused by a mismatched predictor at either high rates or at low rates. However, the critical distortion $1/\Delta$ seems to play a vital role in determining the gain due to, searching.

CHAPTER 5

SMOOTHED CODES

The theorem by Berger [14] discussed in sec. 2.5 clearly establishes that at rates for which the distortion D is less or equal to the critical distortion, even if a decoder filter is perfectly matched to the source, such a source decoder is not optimal. The theorem advocates a mismatched decoder filter on purpose by introducing zeroes in the matched decoder filter transfer function in a certain fashion described in sec. 2.5. The net affect of these zeroes is that we require a smoothing filter cascaded to the matched decoder filter to achieve optimality. A counter-part of the above result at low rates, rates that are of interest to us in this work, has not been reported since. However the significance of b-coeffients specifying the smoothing filter at high rates increases as the rate goes down, i.e., as the distortion D approaches the critical value D_0, so it is very likey that we need a smoothing filter even for $D > D_0$.

In what follows, we shall discuss a rather ad hoc scheme brought about only by intuitive considerations. We combine it with the traditional DPCM decoder filter and investigate the resulting performance. The scheme was initially proposed by Anderson and Bodie [23] to encode speech but it has not been studied in a careful manner with probabilistic sources.

5.1 Smoothed LMS Predictor Codes

The principal effect of the quantiser in the predictive loop of the DPCM encoder is to superpose on the source output $\{x_t\}$ a "zitter", or an $f_s/2$ note (f_s is the sampling frequency). One approach in the design of a smoothing filter will be to remove or at least lessen as much as possible this high frequency note.

The approach suggested by Anderson and Bodie [23] achieves this through a small additional increase in complexity. When the DPCM predictor output is smoothed by a filter with the following specifications, we denote the set of outputs as a smoothed LMS predictor code.

1. Zeroes at $f_s/2$
2. Some attenuation near $f_s/2$
3. Short impulse response (to avoid lengthening the total predictor filter order, and keep down its impulse response)
4. Minimal delay

These requirements can be met by a first order or a second order transversal filter. However, a second order filter seems to do a distinctly better job compared to a first order filter. The transfer functions of first and second order filters can be expressed as:

$$F(z) = f_0 + f_0 z^{-1} \qquad (5.1)$$

$$F(z) = F_0 + pf_0 z^{-1} + (p-1) f_0 z^{-2} \qquad (5.2)$$

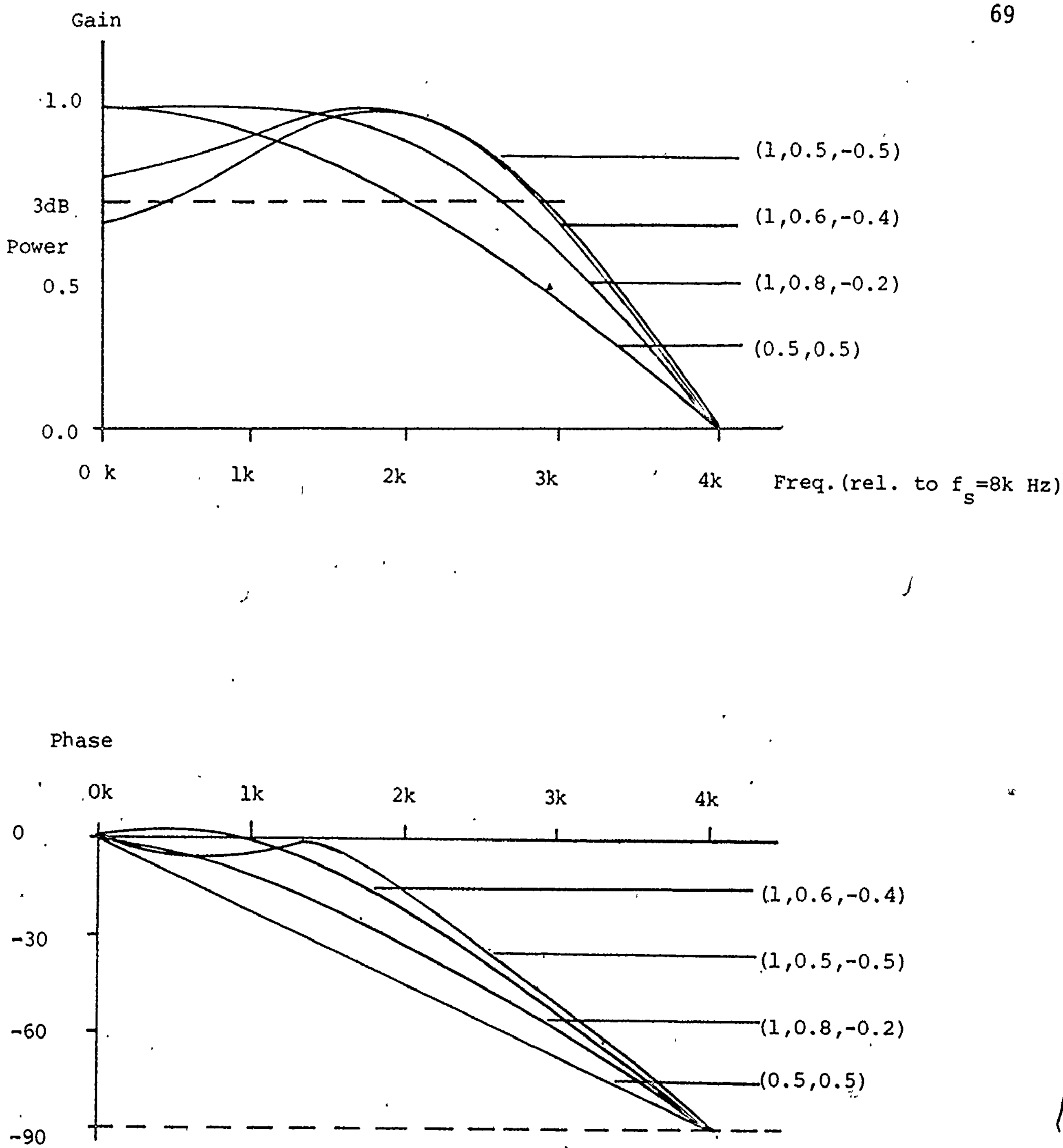

Fig. 5.1 Gain-Phase responce of the smoothing filters

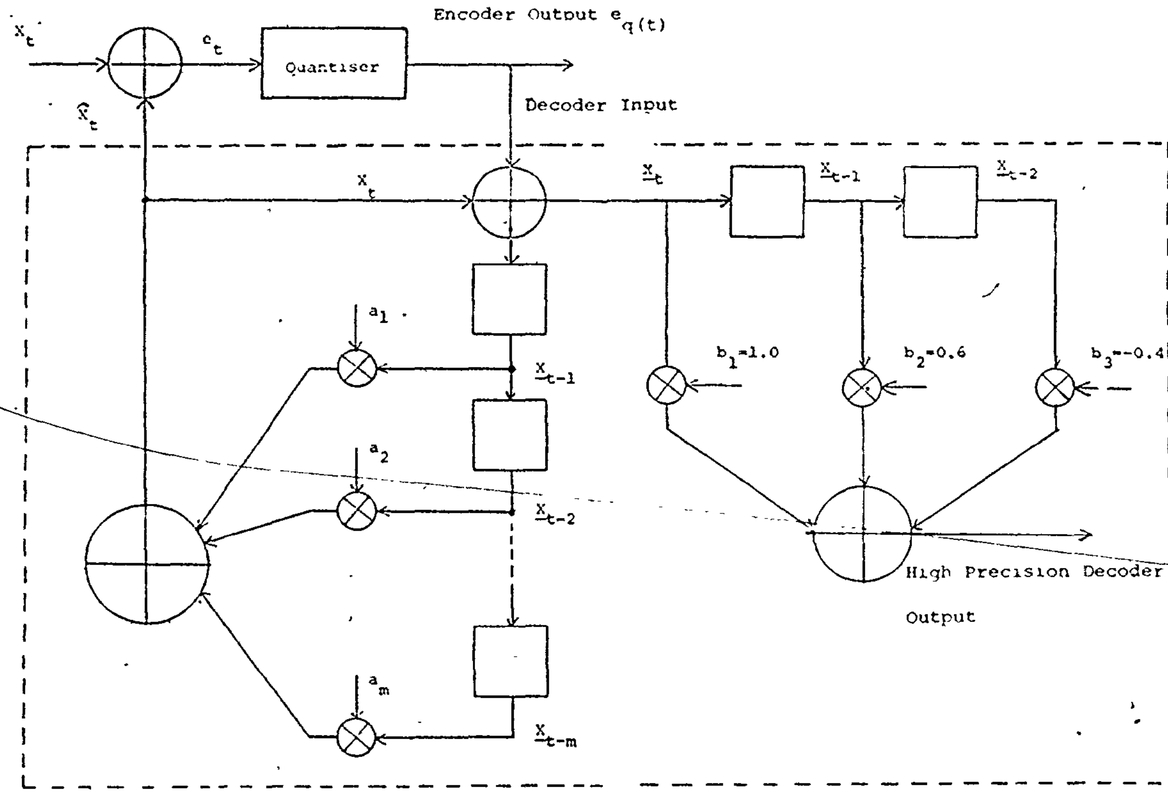

Fig. 5.2 DPCM Encoder (Decoder in the dotted box)

with Smoothing Filter Cascaded

Here p ($1/2 \leq p < 1$) is free parameter. According to Anderson and Bodie [23] a p = 0.6 has been found particularly useful with speech. For a filter of given order p determines the shape of gain-phase response of the filter. With the substitution $z^{-1} = e^{-jw}$, fig. 5.1 gives plots of phase and gain vs. frequency.

5.2 Experimental Setup

In the ensuing experimentations, we shall use a second order filter with p = 0.6. The filter will be cascaded with the traditional DPCM encoder of fig. 1.3 as shown in fig. 5.2. The rest of experimental setup remains exactly the same as in Chapter 4. Autoregressive sources with innovations having Laplacian and Gaussian distributions will be encoded using the encoder of fig. 5.2 and the M-algorithm with M = 4 and L = 32. Matched as well as mismatched source-predictor pairs are examined at rate 1 and 2.

A uniform step-size quantiser is utilized with Gaussian autoregressive sources. From Paez and Glisson [21] the optimum quantiser step size at rate 2 is (0.395, 1.410) for independent letter source with Laplacian distribution. When encoding coloured Laplacian sources, in this case, we shall use a multiplier b on these steps and optimize the size b. Therefore, the SNR plots in this case will be against b.

5.3 Results

The results at rate 2 are shown in fig. 5.3 to fig. 5.5 for Gaussian autoregressive sources. We have used the Butterworth filter

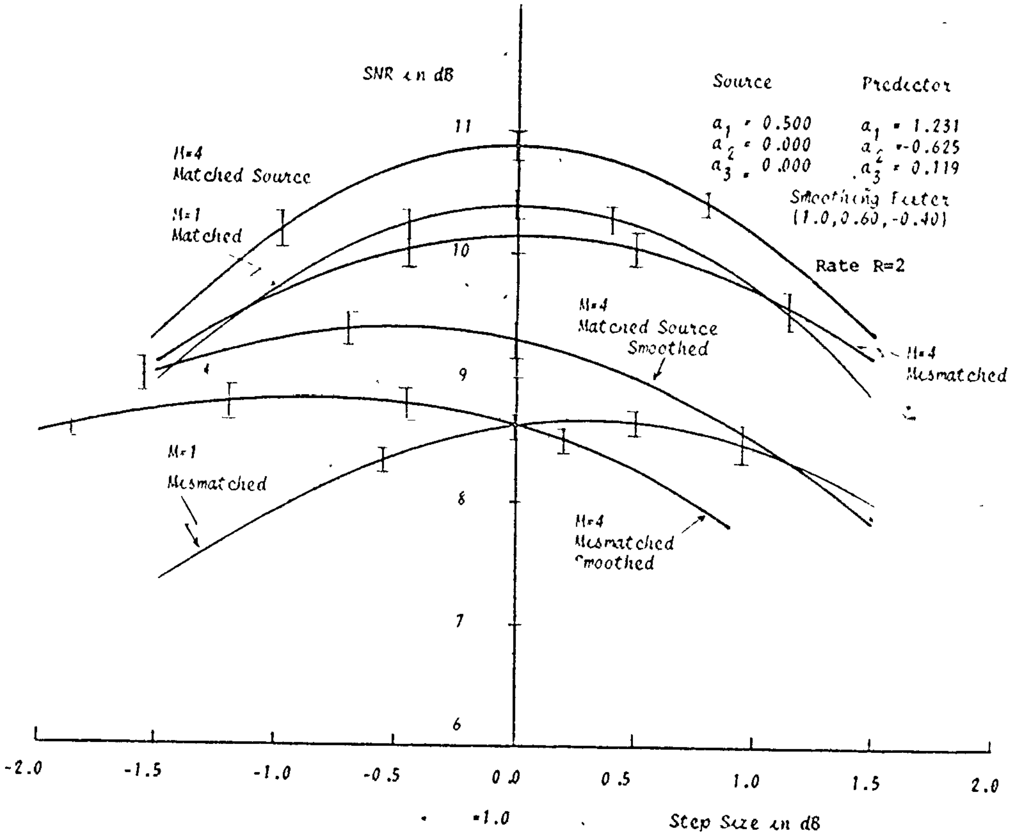

Fig.5.3 SNR performance of DPCM encoder

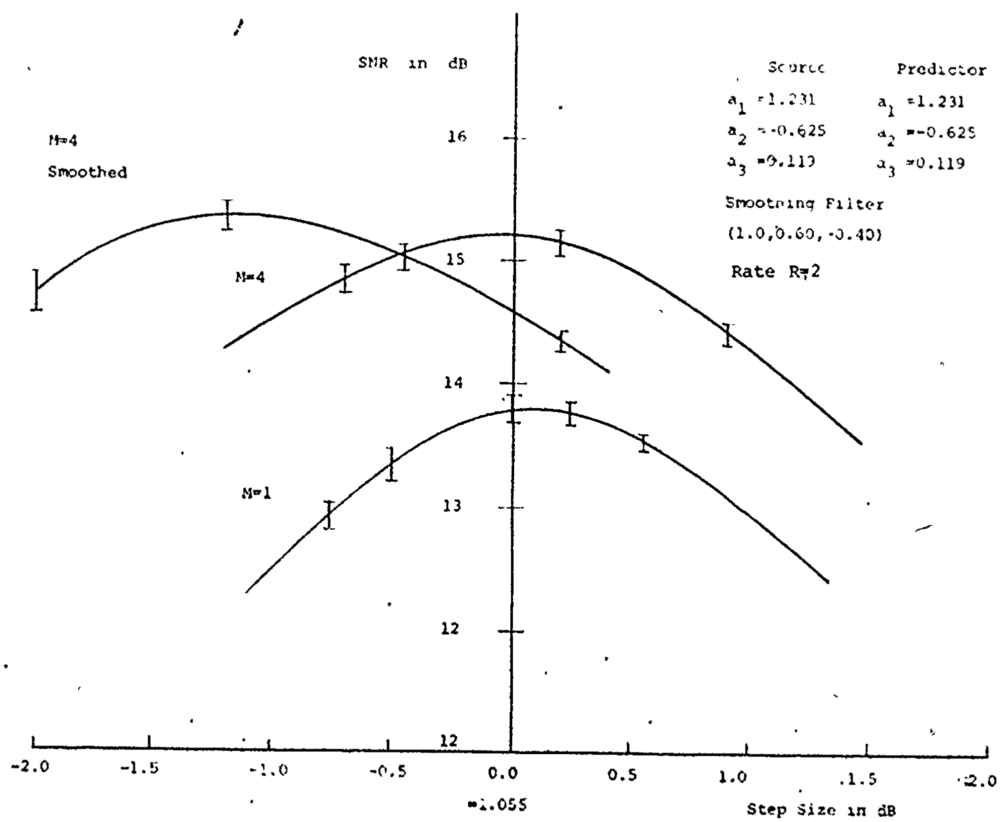

Fig.5.4 SNR performance of DPCM encoder

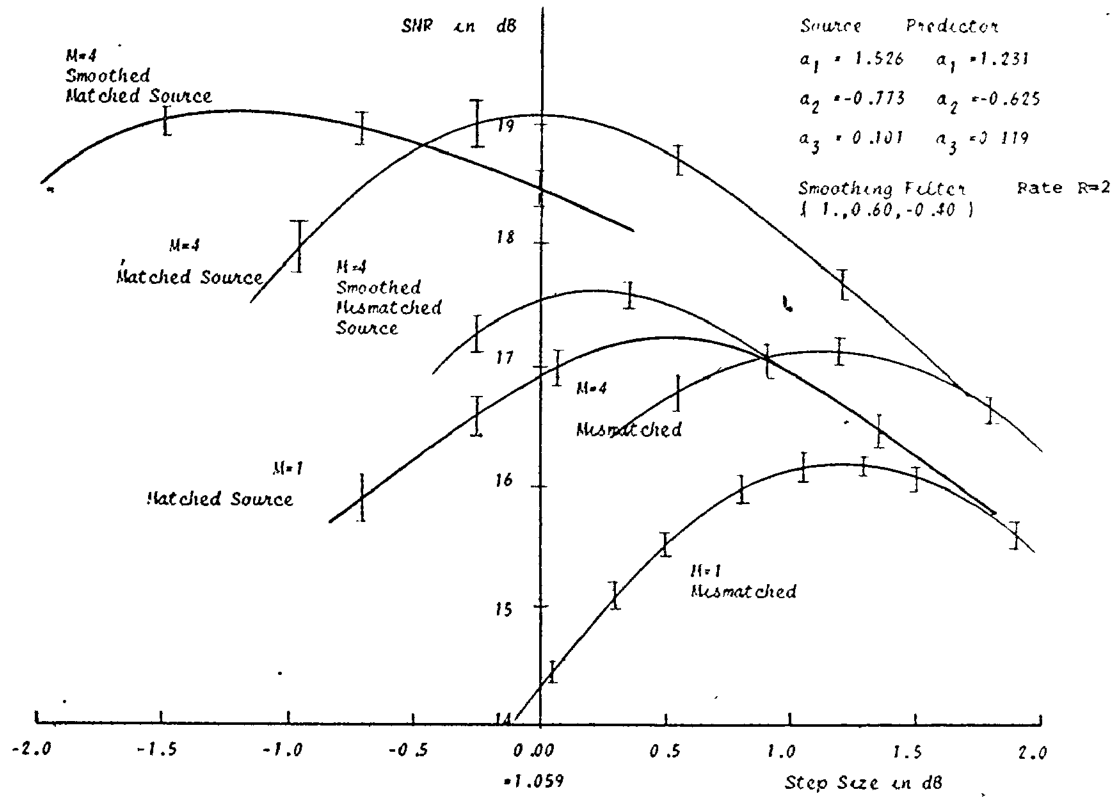

Fig. 5.5 SNR performance of DPCM encoder

taps as predictor in all cases. Butterworth filter (a_1 = 1.231, a_2 = -0.625, a_3 = 0.119) is quite popular in speech encoding. The source is varied from a very lightly correlated single tap source with a_1 = 0.5 in fig. 5.3 to a tightly correlated 3 tap source in fig. 5.5. Smoothing is used both when the predictor is matched to the source and when it is mismatched. The corresponding results for Laplacian autoregressive sources are shown in the fig. 5.6 to fig. 5.8.

In the case of lightly correlated sources (a_1 = 0.5) in fig. 5.3 and fig. 5.6, we see that the smoothing filter has made matters significantly worse compared to non-smoothed case, regardless of if the source is matched to the predictor or mismatched. However, when the source is tightly correlated as in the case of fig. 5.5, we see that the smoothed code performance is comparable to non-smoothed code at M = 4, with the source matched to the predictor. A very slight gain even appears.

When the source and predictor are mismatched, one can see in fig. 5.5 that there is a significant gain of $\sim$ 0.5 dB. However, this is far from making up the loss due to mismatch. Fig. 5.9 and fig. 5.10 show the rate 1 results. Here again, there seems nothing promising.

On the whole, the smoothing filter appears to have made matters worse when the source is lightly correlated. It has given an insignificant gain when the source is tightly correlated and the predictor is matched to the source. There is some improvement when the source is tightly correlated and the predictor is mismatched to the source.

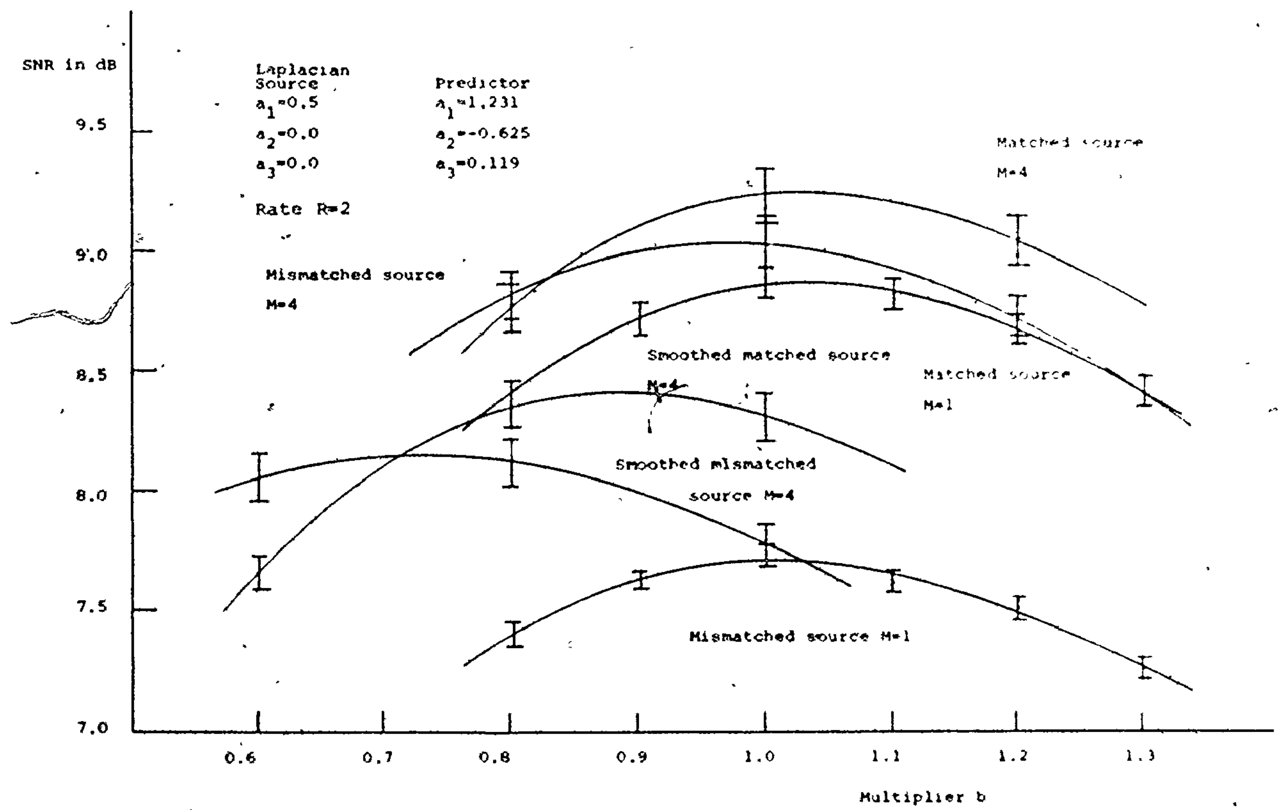

Fig. 5.6 SNR performance of DPCM encoder

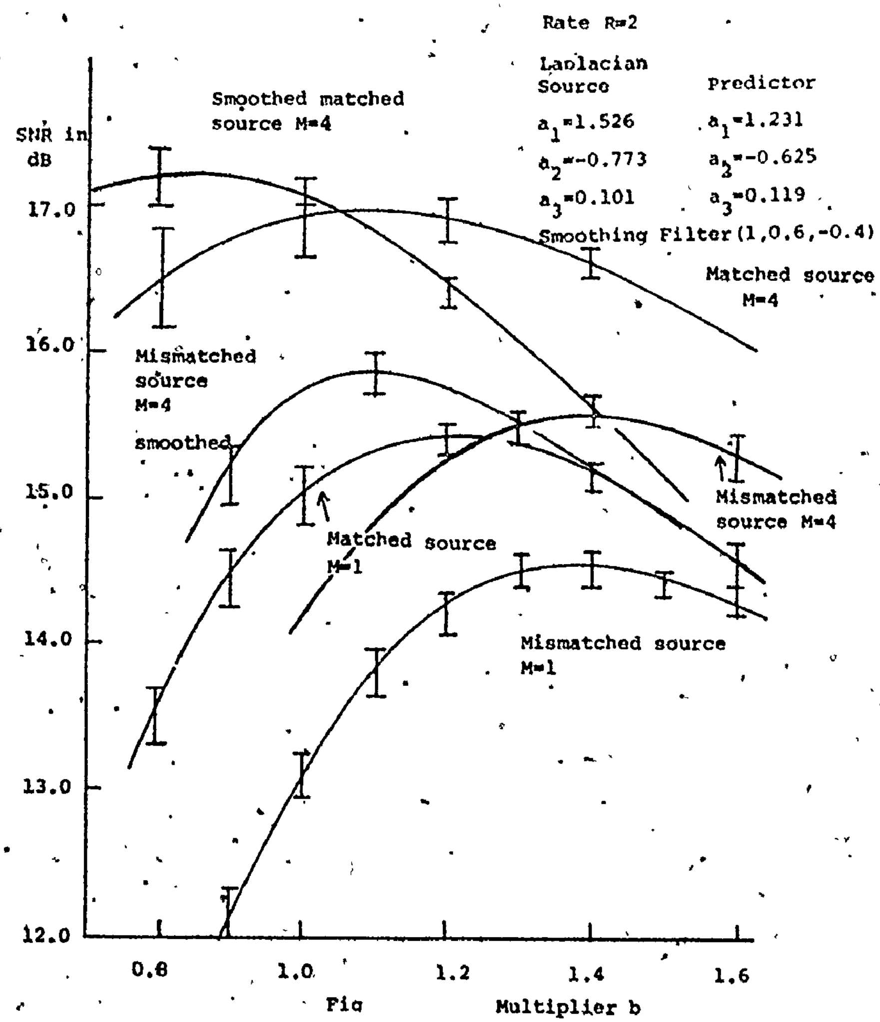

Fig. 5.7 SNR performance of DPCM encoder

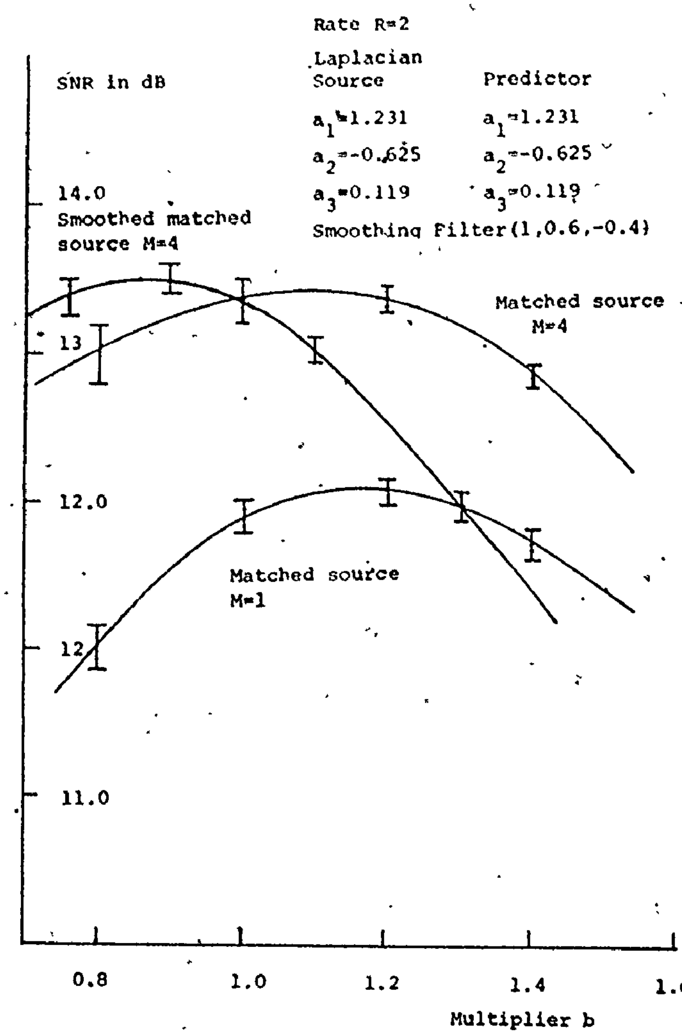

Fig. 5.8 SNR performance of DPCM encoder

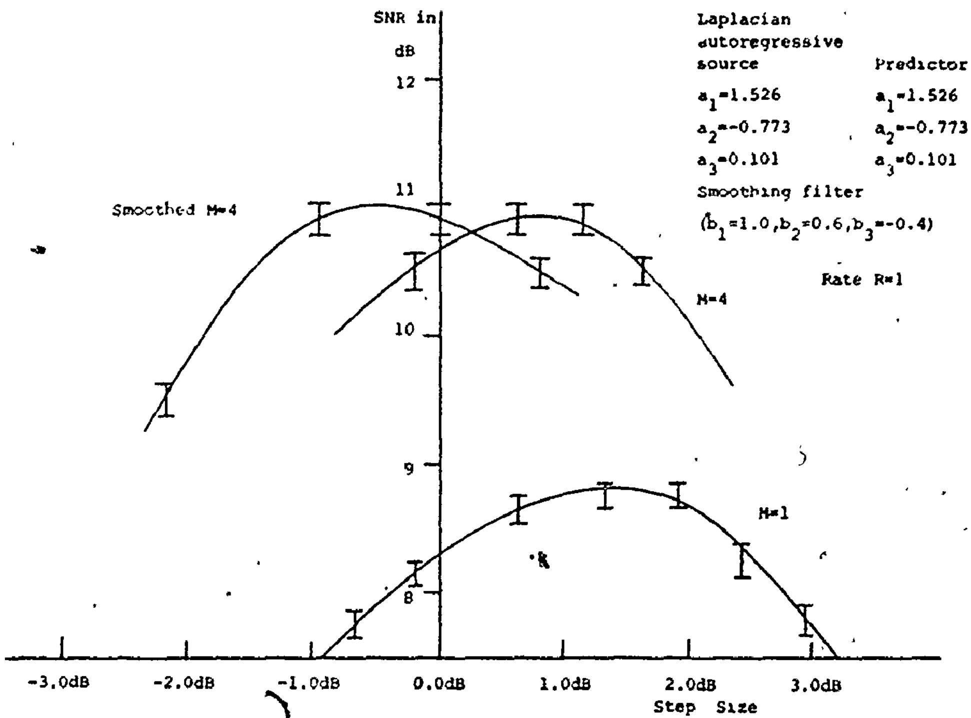

Fig.5.9 SNR performance of DPCM encoder with and without smoothing.

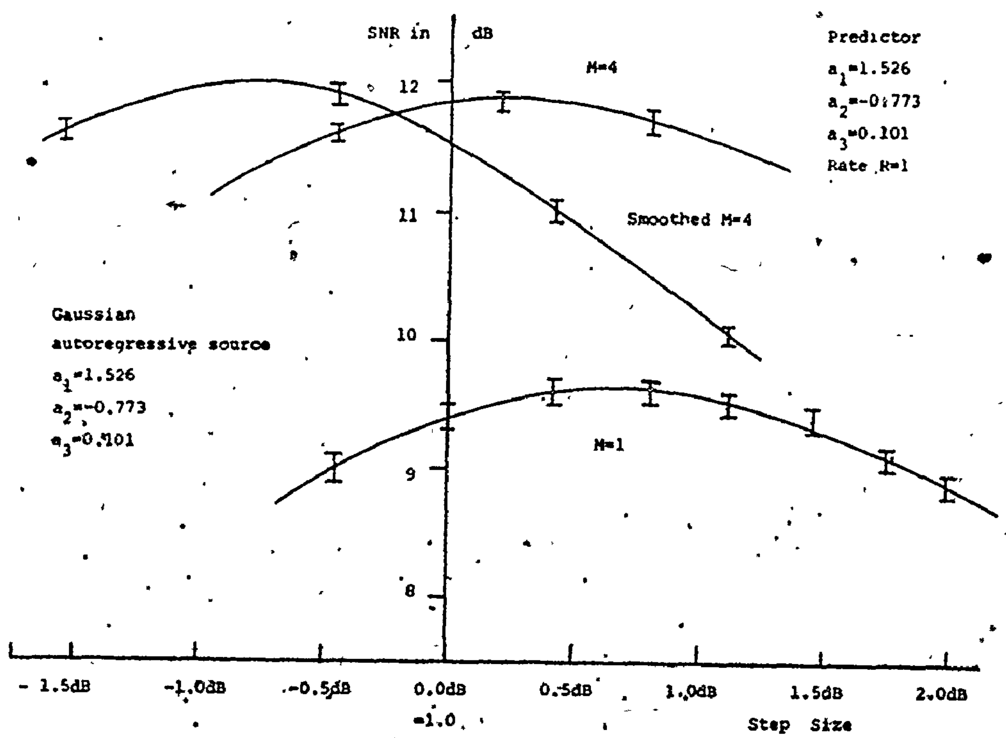

Fig.5.10 SNR performance of DPCM encoder with and without smoothing.

These results agree with the observations of Chan and Anderson [26], who used the same smoothing filter in cascade with the traditional DPCM code generator to encode speech. Furthermore, the role of the distribution as in all previous cases seems to be quite limited.

CHAPTER 6

RANDOM CODES

Minimum Mean Square Error quantisation is not very efficient if a signal having probability density function (pdf) with long tails is to be encoded at low rates. That is, the SNR performance of MMSE (optimal) quantiser is best when encoding uniformly distributed sources, it is not so good with normally distributed sources, it is bad with Laplacian distributed source, and it is even worse with Gamma distributed sources. On the contrary, the rate distortion theory says that greater data compression is possible with Laplacian and Gamma distributed sources whose pdf has long tails.

We noticed in Chapter 3 that the overall SNR obtained with Laplacian sources was significantly lower than that obtained with their Gaussian counterparts. This follows from the fact that we ideally expect the MMSE quantiser SNR to be $SNR_q + SNI$ (where SNR_q and SNI are as defined in Chapter 3) and at rate 1 the SNR_q (ideally) for Laplacian sources is 3 dB while for Gaussian sources it is about 4.4 dB. In addition, the SNR loss at low rates for the two cases was of more or less similar magnitude, thus leaving us with a significantly lower net SNR in case of both memoryless as well as correlated sources.

Tree searching did improve the performance of optimal predictive quantiser codes (DPCM codes) but it gave more or less the same improvement with Gaussian and Laplacian sources. Thus the large potential gains with Laplacian sources remained untapped. In other words, tree searching could not fix the inherent shortcomings of the quantiser codes when encoding sources with a pdf with long tails; all it did was to make up for the damaging interaction of the quantiser nonlinearity and the feedback present in a DPCM circuit irrespective of the pdf of the source.

Although at rates higher than 1 entropy coding has had considerable success in reducing the margin of performance between rate distortion theoretic lower bounds and conventional DPCM (or PCM with memoryless sources), our knowledge of coding strategies that can tap the large potential gains at rate 1 is very limited. The rate distortion theory which establishes the existence of these possible but yet untapped gains used random coding arguments in arriving at these results. Accordingly, our approach in the following work will be to use certain random tree codes to encode Gaussian and Laplacian distributed autoregressive sources at rate 1. We expect that even though suboptimal, these random codes may do well, considering the poor performance of traditional schemes such as PCM and DPCM at these low rates.

6.2 Random Tree Coding

To generate a random ensemble of tree codes for a p-th order, Gaussian autoregressive source when $D \leq 1/\Delta$, one chooses a tree with α branches per node and β symbols on each branch so that $\beta^{-1} \log \alpha$ is the rate of encoding. Each branch of this tree is next populated with independent identically distributed N(0, 1) random variables. The resulting code is called a white tree code and each path through this tree is called a white code word. According to rate distortion theory the ensemble performance of such white tree codes (with a certain scaling operation on the variance of the random variables) should achieve the rate distortion theoretic lower bounds with memoryless Gaussian sources in the limit of large tree depth and an exhaustive search. Next let each white code word be passed through the filter of fig. 2.2 with the shift registers set to their initial zero state between successive input words. The a's in fig. 2.2 are the autoregression coefficients of the sourse and b's are calculated using the relationship (2.1). The resulting set of output words constitute a typical code in the ensemble. Thus, we generate a tree code that is matched to the memory of the autoregressive source by appropriately colouring the white tree codes that one uses for encoding i.i.d. Gaussian source.

If the source encoder were ideal, it would find that word in the white code which results in a colouring filter output word that approximates the source word best. It, then, would forward the path map of this white code word to the channel encoder. Assuming no channel impairments, the source decoder only needs to put the white code word specified by the path map into the colouring filter and present the resulting output word to the user.

Clearly, this scheme requires an exponentially increasing memory and computations as the tree depth grows. In practice, therefore, one needs to modify the above scheme in various ways to incorporate memory and computational constraints. We shall accomplish this with the M-algorithm and by constraining the depth of the tree.

6.2 Experimental Design

At rate 1 for many of the sources that are to be encoded, $D > 1/\Delta$; this implies that we do not have a valid method for computing the b-coefficients in fig. 2.2. Furthermore, the coding strategy of section 2.5 has been proven only for Gaussian autoregressive sources. Taking all these facts into account, our approach in this work will be to set all the b's to zero except for b_0 which is set to 1, in all the experiments, irrespective of the distribution. This will reduce the colouring filter of fig. 2.2 to that of fig. 1.1.

Autoregressive sources are generated exactly in the same fashion as described in the experiments of Chapter 3. In addition, a segment length of 1000 source letters, a sample size of 20, one standard deviation (one sigma) confidence intervals and the M-algorithm with M = 1, 4 and 8, and L = 32 are utilized.

A segment length of 1000 implies 2^{1000} different code words and will require a generation of $\sum_{k=1}^{1000} 2^k = 2^{1001}-1$ random variables (r.v.) to fill a tree of depth 1000. Even though we overcome most of the computational problem by using the M-algorithm, since the algorithm will explore only a limited number of good paths, storing 20 trees of depth 1000 each is not a very pleasant task. For our purpose, we overcome this difficulty by sacrificing the idea of using the same white code for all the sources to be encoded. We shall use the following modified scheme:

A source letter is fed to the encoder, the encoder will call the random number generating subroutine to generate $M2^R$ random variables. (In case of Gaussian AR(P) sources we use the subroutine GGNML [32] and for Laplacian sources, we use the same subroutine GGUBS which is used to generate the Laplacian AR(P) sources but with a different SEED. In our work the rate R is 1, and in the colouring filter of fig. 1.1, the a's are chosen the same as the source autoregression coefficients). In the steady state operation of encoder we have M retained paths from the previous encoding. The contents of the shift register in fig. 1.1 are set to contain the P most recent branch lebels of one of these M retained paths and a r.v. is fed to the circuit. (At the start of the encoding it is assumed that the M paths have their branch labels that are all zero). The shift register is set to its original condition and another r.v. is fed to the filter. Once 2^R r.v. have been fed, the contents of the shift register are changed to contain the P most recent branch lebels from another of the M retained paths. 2^R new r.v. are presented to the filter as before. The process is repeated till all the M paths are extended to generate $M2^R$ new paths. As usual, the encoder retains the M best of these $M2^R$ extended paths. The same procedure is repeated till the whole string of 1000 source letters is encoded. The best of these M paths is taken as the encoder output.

The path metric in this case is the signal-to-noise ratio,

$$SNR = \frac{\sum_{k=1}^{k} x_k^2}{\sum_{k=1}^{k} (x_k - x_k')^2} \qquad (6.1)$$

Here x_k' is the branch lebel of the coloured random tree.

A careful examination of the above scheme suggests that we will not only have a different white tree code from one source to another but a different code even for the same source but a different M. This follows from the fact that we only generate $M2^R$ random numbers at each level of the tree and these $2^R M$ random numbers will fill only certain branches of the tree at that level depending on where we were, the level beforehand.

The main difficulty with this scheme is that information theoretic lower bounds are derived, based on ensemble arguments, i.e., the average performance of an ensemble (collection) of codes achieves the specified lower bounds. Therefore, it is not necessary that all the codes belonging to an ensemble are good. The theory only says that the collective performance of the ensemble will be as good as the lower bounds. Thus, we do not have a unique white code to encode

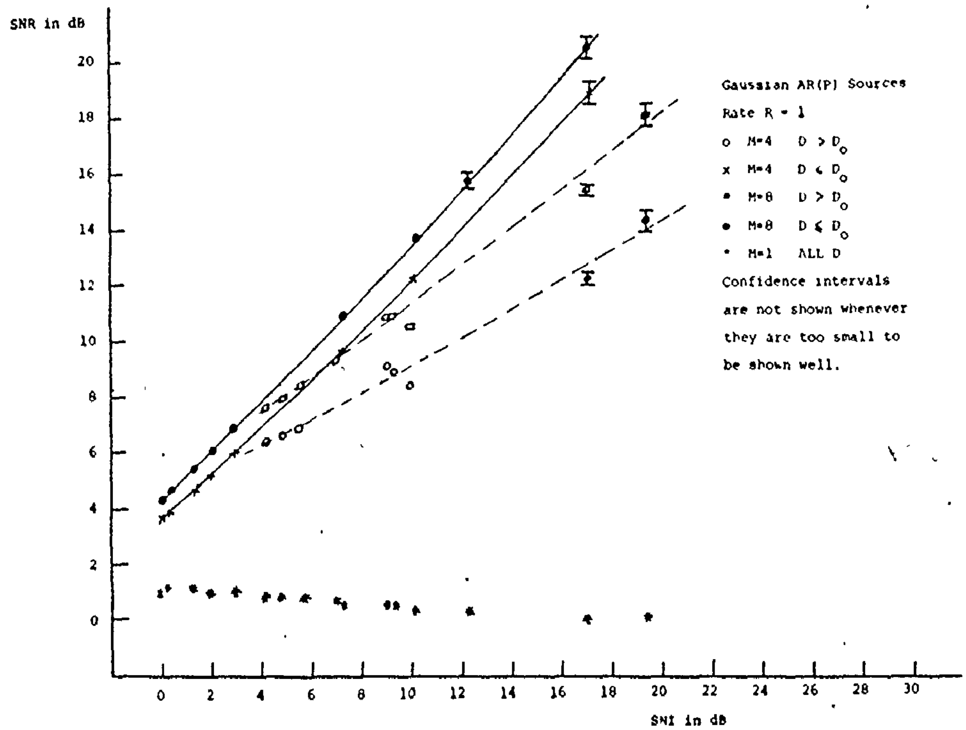

Fig. 6.1 SNR obtained with random codes vs. SNI.

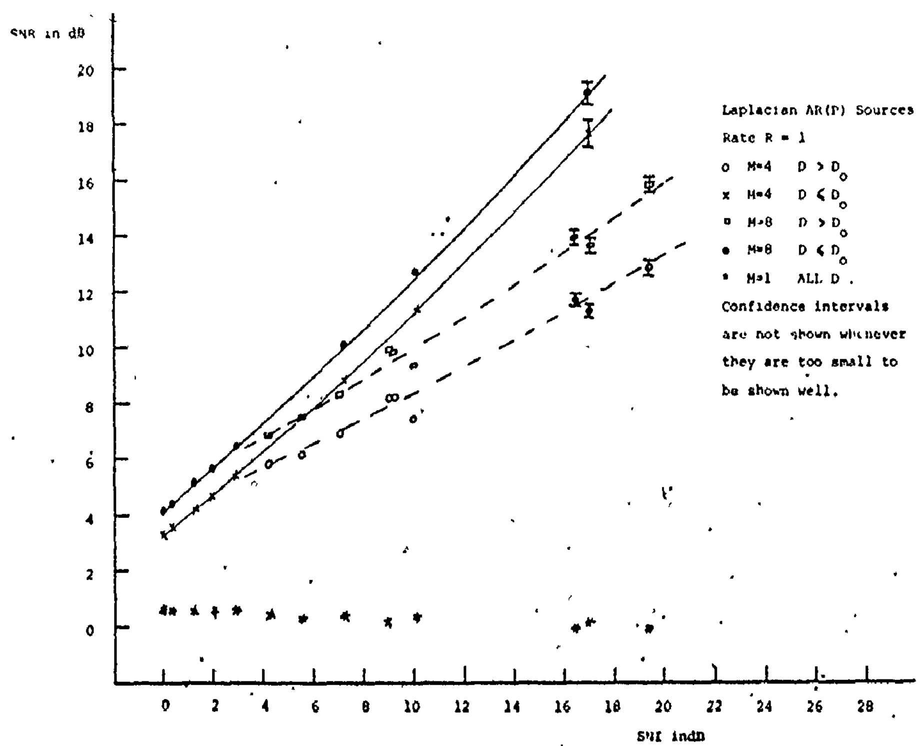

Fig. 6.2 SNR obtained with random codes vs. SNI.

all the sources. But with a fairly large sample size and segment length, the meaningfulness of the results should not be seriously disturbed.

6.3 Results and Conclusions

Fig. 6.1 and 6.2 show the actual SNR obtained with various Gaussian and Laplacian sources plotted against the SNI. At M = 1, the random coding scheme has given more or less zero SNR in either case. However, at M = 4 we get a substantial improvement in both cases and this performance is further improved at M = 8. Here again, like DPCM codes, we see SNR as a function of SNI splits into two trends. Those autoregressive sources for which $D \leq D_0$ obtain a larger SNR than those for which $D > D_0$. Actually, we have no hesitation in fitting curves for points for which $D \leq D_0$: The SNR obtained is proportional to the SNI. But this does not appear to be the case for sources for which $D > D_0$, since the confidence intervals are very small and points are very widely scattered. But we have drawn a dotted curve to ease the comparison between the two cases.

It is also worth pointing out that many of the sources for which $D \leq D_0$ are single tap sources while all the others are multi-tap sources. Also, the SNR improvement in going from M = 4 to 8 is more for sources for which $D > D_0$. Another important observation about the plots, is that at a moderate search like the one we have used, SNR's obtained with Laplacian and Gaussian sources are more or less of the same magnitude at low SNI values. With DPCM, this

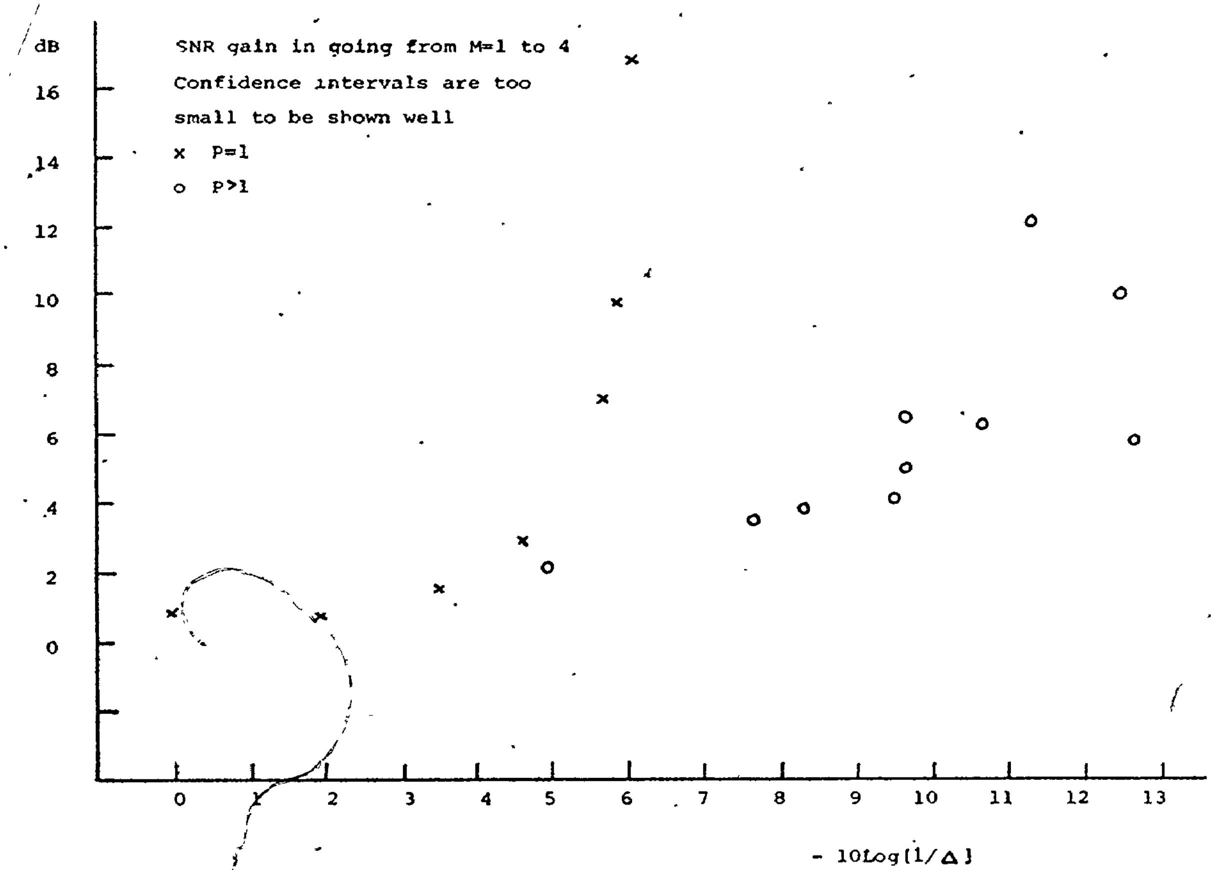

Fig. 6.3 SNR gain vs. the critical distortion D_o for gaussian AR(P) Sources.

is not the case as we have already pointed out.

Fig. 6.3 shows the SNR improvement in going from M = 1 to 4 for Gaussian AR(P) sources as a function of the critical distortion D_0. Clearly, it is not possible to detect any underlying law in the plotted points like we did in the case of DPCM (see Fig. 4.7 and 4.8). Since at M = 1, the random coding scheme has not given a significant SNR while DPCM codes gave most of their SNR performance at M = 1, this comparison won't be very realistic. A more meaningful comparison will be the one in which we choose an M which gives more or less the same SNR with random codes as did M = 1 for DPCM. Then we multiply the M in both cases by 4 and compare the SNR gain as a function of $1/\Delta$.

Fig. 6.4 and fig. 6.5 show the SNR improvement in going from M = 4 to 8 as a function of $1/\Delta$ for both Gaussian and Laplacian AR(P) sources, respectively. The SNR improvement now is a fairly well-behaved function of the critical distortion. Furthermore, unlike DPCM, we have obtained SNR improvement with memoryless Gaussian and Laplacian sources in this scheme in going for a higher M.

Fig. 6.6 shows the gain in SNR obtained by the random codes at M = 4 and 8 over the DPCM codes with M = 4 for Gaussian AR(P) sources. It is clear that at M = 4 random codes performed poorly compared to DPCM codes at M = 4. At M = 8 there is significant improvement with the random codes over M = 4 but still DPCM codes at M = 4 outperform them. It has been pointed out in previous chapters that there is little to be gained by going from M = 4 to

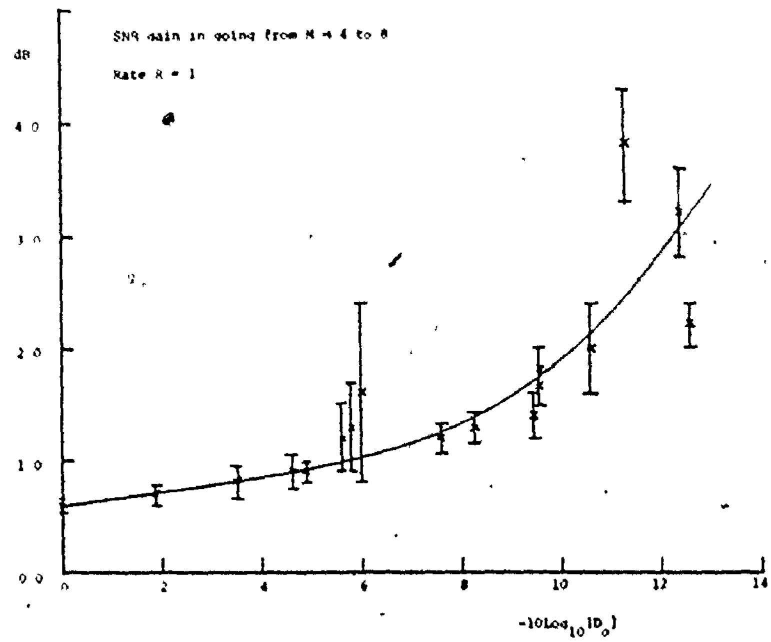

Fig. 6.4 SNR gain vs. the critical distortion for Gaussian AR(P) Sources

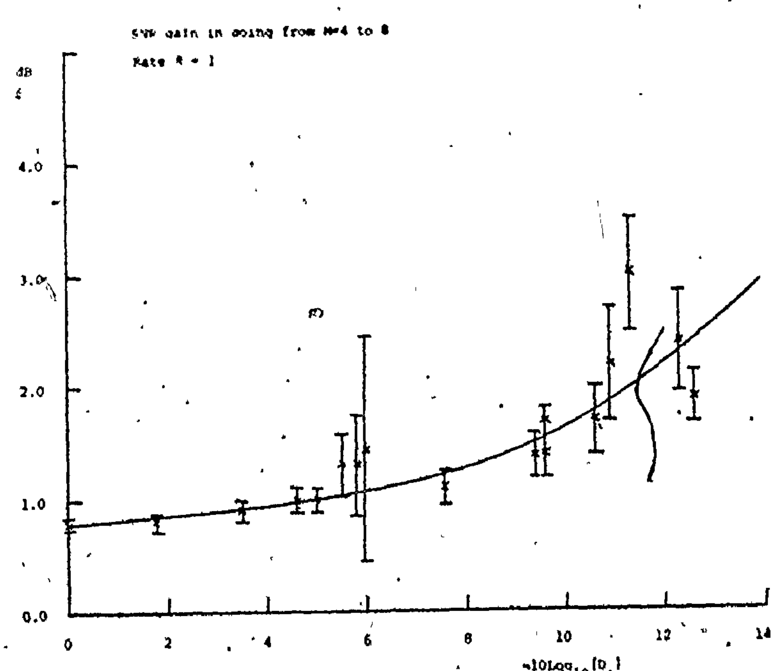

Fig. 6.5 SNR gain vs. the critical distortion for Laplacian AR(P) sources.

M = 8 with DPCM codes. But this is not the case with random codes and if we search harder, the performance of these codes will improve and eventually should overtake DPCM. However, it is also clear that at least with Gaussian AR(P) sources, random codes will require an intense search for a significant improvement over the traditional schemes such as DPCM with tree searching. This is not unjustified considering the fact that performance of DPCM with tree searching is not very far from rate distortion theoretic lower bounds with Gaussian sources.

The SNR performance improvement obtained with Laplacian AR(P) sources at rate 1 with M = 4 and 8 over DPCM at M = 4 is shown in fig. 6.7. As expected, we see a different situation in this case. Random codes at M = 4 outperform the DPCM codes at M = 4 when the sources are uncorrelated or very lightly correlated. At M = 8 we see an even greater improvement in the gap between the performances of a random code and the DPCM. In the region of light correlation, the random codes outperform DPCM codes at M = 4 by as much as 1 dB. This figure is significant since M = 8 is by all means a moderate search effort. However, in the absolute SNR terms, we have not achieved a higher SNR than the Gaussian case. The gain over DPCM resulted only because DPCM performed poorly with long tailed Laplacian distributed sources.

Another important observation is that with tightly correlated sources (large Δ) the random codes even at M = 8 did not perform well, compared to DPCM codes with tree searching. The reason for this may

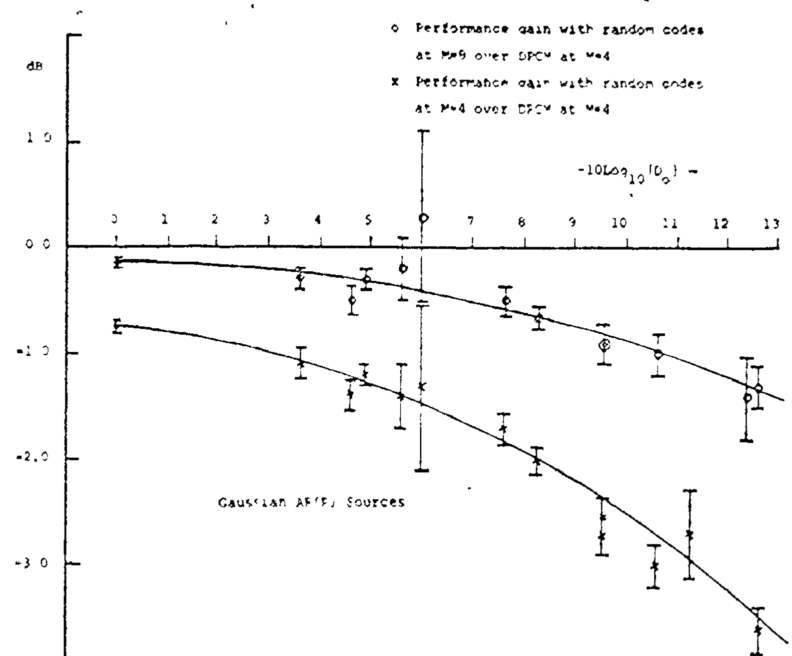

Fig 6.6 A comparison between the performance of a random coding scheme and DPCM codes

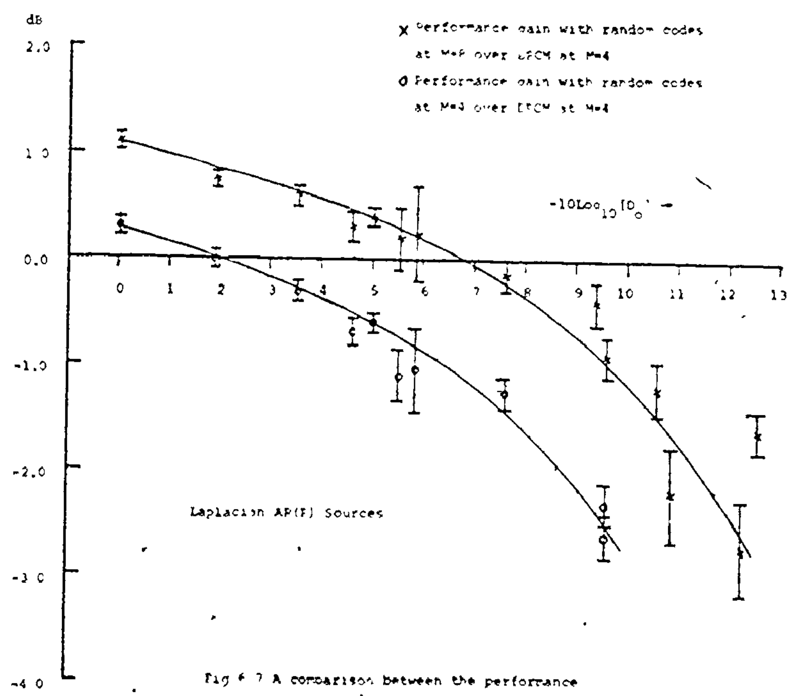

Fig 6.7 A comparison between the performance of a random coding scheme and DPCM codes

be twofold. With tightly correlated sources, random codes appear to require a more intense search. Secondly, as Δ grows, our sub-optimal coding scheme becomes more and more suboptimal and this may have a more pronounced effect on random codes than DPCM codes. But in any case, the amount of SNR improvement that materialised at low Δ-values is very encouraging and makes one wonder if random coding could be a viable scheme at low rates for sources whose pdf has a long tail.

CHAPTER 7

SUMMARY AND CONCLUSION

One important result of this work is that the traditional DPCM encoder is ill-equipped to perform an efficient encoding of autoregressive sources at low rates. The widespread myth that better techniques of adaptation are the answer to low rate waveform encoding of speech appear quite ill-founded. The results in Chapter 3 establish that even a perfectly matched predictor causes severe performance degradation at low rates and is clearly far from an optimal scheme. In addition, we have been able to devise certain experimental laws which describe the performance of the DPCM encoder at low rates for general autoregressive sources. The quantiser non-linearity combined with the feedback present in a DPCM circuit has frustrated all attempts so far for an analytical evaluation of the DPCM encoder performance at low rates for general AR(P) sources. Thus, these experimental laws will be useful whenever it is not convenient to evaluate the encoder performance through actual simulation.

A moderate amount of tree searching has been very successful in improving the performance of the DPCM encoder and this improvement resulted because the tree searching checked with a considerable success the feedback effect of the large quantisation errors at these low rates.

The results clearly establish that tree searching is not a replacement for better adaptation: The SNR improvement due to adaptation is nearly independent of the gain due to searching. Our results also establish that tree searching with DPCM codes will not help at high rates but at low rates a moderate degree of searching is indispensible, particularly with tightly correlated sources.

Here again we have been able to detect experimentally the underlying laws governing the gain due to tree searching. The gain is a very well-behaved function of the critical distortion D_0 (= $1/\Delta$) in Grays theorem (see sec. 2.2) and the SNR Loss defined in sec. 3.5.

The smoothing filter that we utilized in Chapter 5 made the performance worse in many cases and gave only a slight improvement when the source was very tightly correlated. Therefore, it is possible that the smoothing filter design needs to take into account the source statistics. In fact, the R(D) theory smoothing filters discussed in sec. 2.5 did take into account the source statistics. The smoothing filter scheme in this work which did not account for the source statistics, however, is an unproductive additional complexity.

The above results holds by and large, irrespective of the distribution of the source, be it Gaussian or Laplacian. In short, to achieve best results with a DPCM encoder at low rates the predictor must be well matched to the source, with or without tree searching. An important, though rather unpleasant conclusion of this work is that even tree searched DPCM with a perfectly matched predictor may not be adequate to achieve the goal of rate 1 waveform encoding of speech with a telephone quality reproduction. (Something that has so far eluded us and the many reasons e.g. faster and better adaptation attributed to this by various authors in the past appear very inconvincing now). This leaves us with no choice but to look for a better code generator different from DPCM. DPCM has been particularly ineffective in performing efficient data compression with Laplacian distributed sources. According to rate distortion theory there are large untapped gains here. Though tree searched DPCM did improve the performance, a lot of ground is yet to be covered.

Random coding scheme of Chapter 6 performs poorly with Gaussian AR(P) sources compared to tree searched DPCM at moderate search efforts. However, this was not the case with Laplacian AR(P) sources: We obtained a significant amount of gain over tree searched DPCM with lightly correlated sources at a moderate search. This is very interesting since speech can be considered a Laplacian AR(P) source. An even better approximation to the distribution of speech is the Gamma distribution and the random coding scheme may perform even better with these sources, compared to DPCM with tree searching. The theoretical gap between the performance of quantiser codes and rate distortion theoretic lower bounds in this case is almost double that for Laplacian sources.

However, the performance of random codes dropped significantly at tight correlations compared to DPCM with tree-searching, perhaps due to the fact that our scheme becomes very suboptimal at these correlations. Thus, it is essential that we find the optimal (or nearly optimal) code generator to perform an efficient encoding of these sources at low rates.

TABLE

Autoregressive sources used in the experiments and their relevent statistics:

AR(P) coefficients	Autocorrelation coefficients	Eigen-value Spread (dB)	SNI (dB)	$10 \log_{10}(1/\Delta)$
0.25	0.25		0.28	2.6
0.5	0.5	4.8	1.25	-3.5
-0.5	-0.5	4.8	1.25	-3.5
0.7	0.7	7.5	2.92	-4.6
0.9	0.9	12.8	7.21	-5.6
0.93	0.93	14.5	8.69	-5.7
0.95	0.95	15.9	10.11	-5.8
0.97	0.97	18.0	12.28	-5.9
0.99	0.99	23.0	17.01	-6.0
0.656 -0.094	0.6 0.3	8.4	1.97	-4.9
1.029 -0.371	0.75 0.4	12.8	4.2	-7.6
1.116 -0.468	0.76 0.38	13.6	4.82	-8.25
0.95 -0.117	0.85 0.69	14.4	5.64	-6.31
1.421 -0.579	0.9 0.7	19.5	8.98	-9.55
1.333 -0.667	0.8 0.4	16.6	6.98	-9.55
1.725 -0.781	0.968 0.889	27.5	16.45	-10.92
1.807 -0.843	0.98 0.927	31.4	19.44	-11.25

TABLE (cont.)

AR(P) Coefficients	Autocorrelation Coefficients	Eigen-value Spread(dB)	SNI (dB)	$10Log_{10}(1/\Delta)$
1.231	0.790	17.1	5.5	-9.45
-0.625	0.441			
0.119	0.17			
1.526	0.9	23.0	9.25	-10.62
-0.773	0.691			
0.101	0.46			
1.869	0.968	31.0	17.0	-12.37
-1.10	0.889			
0.185	0.781			
1.748	0.864		9.9	-12.6
-1.222	0.557			
0.301	0.227			

REFERENCES

[1] N. S. Jayant, Waveform Quantisation and Coding, IEEE Press, 1976.

[2] C. E. Shannon, "A Mathematical Theory of Communication", Bell Systems Tech. J., Vol. 27, (pt. I), pp. 379-423, (pt. II), pp. 623-656, 1948.

[3] J. D. Gibson, "Adaptive Prediction in Speech Differential Encoding Systems", Telecommun. and Control Systems Lab., Texas A and M University, Oct. 31, 1979.

[4] R. M. Gray, "Information Rates of Autoregressive Processes", IEEE Trans. Info. Theory, Vol. IT-16, July, 1970.

[5] R. A. McDonald, "Signal-to-Noise and Idle Channel Performance of Differential Pulse Code Modulation Systems - Particular Applications to Voice Signals", Bell Syst. Tech. J., Vol. 45, pp. 1123-1151, Sept. 1966.

[6] J. B. O'Neal and R. W. Stroh, "Differential PCM for Speech and Data Signals", IEEE Trans. Commun., Vol. COM-20, pp. 900-912, Oct. 1972.

[7] J. B. O'Neal, "A Bound on Signal-to-Quantising Noise Ratio for Digital Encoding Systems", Proc. IEEE, Vol. 55, pp. 287-292, Mar. 1967.

[8] H. Gish and J. H. Pierce, "Asymptotically Efficient Quantising", IEEE Trans. Info. Theory, IT-15 (2), pp. 248-252, 1969.

[9] T. J. Goblick, Jr., and J. L. Holsinger, "Analog Source Digitisation: A Comparison of Theory and Practice", IEEE Trans. Info. Theory, (correspondence), Vol. IT-13, pp. 323-326, April 1967.

[10] F. Jelinek, "Tree Encoding of Memoryless Time-discrete Sources with a Fidelity Criterion", IEEE Trans. Info. Theory, Vol. IT-15, pp. 584-590, Sept. 1969.

[11] P. Noll and R. Zelinski, "Bounds on Quantiser Performance in the Low Bit Rate Region", IEEE Trans. Commun., Vol. COM-26, Feb. 1978.

[12] J. B. O'Neal, "Entropy Coding in Speech and Television Differential PCM Systems", IEEE Trans. Info. Theory, Vol. IT-17, pp. 758-761, Nov. 1971

[13] C. R. Davis and M. E. Hellman, "On Tree Coding with a Fidelity Criteria", IEEE Trans. Info. Theory, Vol. IT-21, pp. 373-378, July 1975.

[14] T. Berger, Rate Distortion Theory - A Mathematical Basis for Data Compression, Prentice Hall Inc., New Jersey, 1971.

[15] W. Feller, An Introduction to Probability Theory and Its Applications, New York: Wiley, 1967, Vol. 2.

[16] Denis J. Aigner, Principle of Statistical Decision Making, MacMillan, 1968.

[17] A Papoulis, Probability, Random Variables and Stochastic Processes, McGraw-Hill, New York, 1965.

[18] R. G. Gallager, "Tree Encoding of Sources with a Distortion Measure", IEEE Trans. Info. Theory, Vol. IT-20, pp. 65-76, Jan. 1974.

[19] J. B. Bodie, "Multi-Path Tree Encoding for Analog Data Sources", Internal Report CRL-20, Comm. Research Lab., McMaster University, Hamilton, Ontario, June, 1974.

[20] J. Max, "Quantising for Minimum Distortion", IRE Trans. Info. Theory, Vol. IT-6, pp. 7-12, Mar. 1960.

[21] M. D. Paez and T. H. Glisson, "Minimum Mean-Squared-Error-Quantisation in Speech PCM and DPCM Systems", IEEE Trans. Commun. Vol. COM-20, pp. 225-230, April 1972.

[22] D. S. Arnstein, "Quantisation Error in Predictive Coders", IEEE Trans. Commun. Vol. COM-23, April 1975.

[23] J. B. Anderson and J. B. Bodie, "Tree Encoding of Speech", IEEE Trans. Info. Theory, Vol. IT-21, pp. 379-381, July 1975.

[24] N. S. Jayant and S. A. Christensen, "Tree Encoding of Speech Using the (M, L) Algorithm and Adaptive Quantisation", IEEE Trans. Commun., Vol. COM-26, pp. 1376-1379, Sept. 1978

[25] S. G. Wilson and S. Husain, "Adaptive Tree Encoding of Speech at 8000 bps with a Frequency-Weighted Error Criterion", IEEE Trans. Commun., Vol. COM-27, pp. 165-170, Jan. 1979.

[26] H.-C. Chan and J. B. Anderson, "Speech Digitizing by Adaptive DPCM with Tree Searching", Internal Report, CRL-76, Commun. Research Lab., McMaster University, Hamilton, Ontario, Aug., 1980.

[27] J. Makhoul, "Linear Prediction: A Tutorial Review", Proc. IEEE, Vol. 63, pp. 561-580, April 1975.

[28] R. M. Gray, "Time-Invariant Trellis Encoding of Ergodic Discrete-Time Sources with a Fidelity Criterion", IEEE Trans. Info. Theory, Vol. IT-23, pp. 71-83, Jan. 1977.

[29] Ulf Grenander and G. P. Szeto, Toeplitz Forms and Their Applications, Berkeley: University of California Press, 1958.

[30] J. B. Anderson and F. Jelinek, "A 2-Cycle Algorithm for Source Coding with a Fidelity Criterion", IEEE Trans. Info. Theory, Vol. IT-19, pp. 77-92, Jan. 1973.

[31] A. J. Viterbi and J. K. Omura, "Trellis Encoding of Memoryless-Discrete-Time Sources with a Fidelity Criteria", IEEE Trans. Info. Theory, Vol. IT-20, (3), pp. 325-332, 1974.

[32] IMSL Library, IMSL, Customer Relations, Sixth Floor, NBC Building, 7500 Bellaire Boulevard, Houston, Texas, 77036, U.S.A.

[33] N. K. Sinha and Z. Abu-El-Magd, "Time Series Models for Adaptive Prediction in Speech Differential Encoding Systems", Dept. Elec. Eng., McMaster University, Hamilton, Ontario, 1981.

[34] Box and Jenkins, Time Series Analysis: Forecasting and Control, San Francisco: Holden-Day, 1970.