A Variable Rate Adaptive Transform Coder for the Digital Storage of Audio Signals

R. W. Tansony, B.A.Sc., P.Eng.

Department of Electrical Engineering
McGill University
Montréal, Québec

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

April 1987

ABSTRACT

This thesis describes the development of a transform coding algorithm for the digital archiving of audio signals. The storage of both speech and music is considered. Silence deletion and signal bandwidth estimation are employed to provide a continuously variable bitrate, adaptively matched to the characteristics of the input signal. The economic and operational feasibility of replacing a traditional analog archive with a coder-based digital system is examined. A floating point Fortran simulation shows that the proposed archiving algorithm offers average storage savings of 75% over 16 bit linear-PCM systems.

SOMMAIRE

Cette thèse décrit le developpement d'un algorithme adaptif pour l'archivage numérique par transformée des signaux auditifs. L'archivage de la parole et de la musique y est traité. L'élimination des intervalles silencieux, et l'estimation de la largeur de bande des signaux d'entrée sont utilisés afin de fournir un débit binaire variable et continue, en s'adaptant, aux caractéristiques du signal d'entrée. Le potentiel économique et opérationnel du remplacement des archives analogiques traditionnelles par un tel archivage numérique avec codeur, est examiné. Une simulation en Fortran à point flottant a montrée que l'algorithme d'archivage proposé, offre une économie de stockage de 75% sur les systèmes PCM (linéaire á 16 bits).

ACKNOWLEDGEMENTS

I would like to thank Dr. P. Kabal for his many suggestions and criticisms. Thanks also to Dr. P. Mermelstein for reviewing the algorithm in its formative stages.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

Chapter 1 Introduction

1.1 The Mass Storage of Audio Signals

This thesis presents an adaptive transform coder designed for the mass storage of digital audio signals.

Advances in optical storage technology are now producing the low cost media and devices needed to make such storage attractive relative to archiving via traditional analog magnetic media. The recent success of compact audio disks demonstrates the current demand for economical, high quality, compact, and durable digital audio storage. Such digital storage offers many advantages over previously available analog methods. Compact disk technology however, is designed to capture high fidelity music, and is not ideal for all audio sources. For instance, compact disk's 705.6 kb/s bitrate is much higher than that required to store speech signals. In fact, recent advances in speech coding have produced sophisticated algorithms that remove redundancies from speech and allow for the storage or transmission of speech at reduced bitrates, with no apparent reduction in quality.

The objective of this work is to develop an adaptive coding algorithm that responds efficiently to wide variations in source material *type* and *fidelity* — deleting silent intervals, and varying the storage bitrate as the bandwidth and characteristics of the source material change.

Specifically, the archiving algorithm will be designed to meet the needs of a large museum audio archive collection. Such collections are found in the museums of major cities around the world, and are typically 50% speech and 50% music — comprised of interviews with artists and dancers, political dissertations, native songs and dances, and other ethnic selections of interest provided by ethnomusicologists.

The collections are held on a wide variety of storage media, including wax cylinders, shellac and vinyl disks, wire spools, cassettes, and various vintages of reeled tape. Thus, the source material varies in quality and fidelity over a wide range. A typical museum collection could easily contain 10,000 to 20,000 hours of material.

Each selection is copied onto modern $\frac{1}{4}$ inch analog magnetic reel-to-reel tape before presentation to the public. These $\frac{1}{4}$ inch tape copies must currently be re-recorded every 5 to 8 years due to the aging (print through, binder degradation, etc.) of the analog storage medium. This is a time consuming and expensive task, and great interest is being shown throughout the audio archiving field in the potential economies to be realized through digital optical storage.

The archiving algorithm presented in this work will be derived from an existing speech coding algorithm, developed for the efficient fixed rate transmission of narrowband speech: The Adaptive Transform Coder. An economic analysis will show the feasibility of replacing existing analog archives with a digital system using the proposed variable rate archiving algorithm. Simulation results will show that the algorithm provides very high quality representations of both speech and music signals, at storage savings of 68–87% over 16 bit linear PCM systems.

1.2 Scope and Organization of the Thesis

In Section 2, the need for an adaptive audio coding algorithm is outlined, with specific reference to the requirements of a museum audio archive collection. It is shown that the algorithm must respond efficiently to both speech and music, over a wide range of signal fidelities. Design criteria for the coder are developed, mass storage media are explored, and an economic analysis is performed on the proposed digital archiving system.

Section 3 examines the state of the art in speech coding research. Background information is given on basic coder types, and the development of the final coding algorithm is traced from waveform coder to frequency domain coder to adaptive transform coder. Background is also given on the components of the adaptive transform coder. At each stage of development, choices are made from the background options presented. The rationale for each choice, based on the results of published research, and on the design criteria of Section 2, is explained.

Using the background information of Section 3, the final variable rate archiving algorithm is described in Section 4. Emphasis is placed on the changes made to the standard ATC algorithm to achieve a variable bitrate, and to achieve silence deletion.

In Section 5, the results of a floating point Fortran simulation of the algorithm are given. These results show reductions in storage space ranging from 68–87%, over 16 bit linear pulse code modulation (PCM) coding, depending on the source material used. The results of performance comparisons between the variable rate archiving algorithm and both the CCITT standard wideband coder [1], and a basic log-PCM coder [2], are also presented in this section.

Conclusions, and suggestions for modifications to, and real time implementation of, the algorithm are given in Section 6.

1.3 Survey of Related Work

Very little information on coding for the purpose of archiving audio signals has been published. The present work has thus been based on the wealth of information available in the speech coding literature. While most of this work is tailored towards narrowband (0–4 kHz) speech coding for toll quality transmission over telephone lines, interest has recently been shown in wideband (0–8 kHz) speech coding, variable rate speech coding, and in coding for speech storage. These latter topics are strongly related to the present work, and have provided much of the basic background research upon which the high quality audio archive coder was built.

Several general surveys of speech coders exist [3,4,5]. These proved useful in choosing an appropriate algorithm for audio archiving, even though such was obviously not their original purpose. More specific works exist on frequency domain coding [6], and adaptive transform coding [7] — both of which provided information useful in the development of the final archiving algorithm.

Past research on the *variable rate* coding of speech, although potentially of great use in the present work, is concentrated on applications in telephone transmission networks. Such applications are based mainly on achieving variable bitrate through bit allocation amongst multiple users on a network [8,9,10]. This is of little use in an archiving application.

More useful information on achieving variable rate coding is available in the speech coding literature on dynamic bit allocation [11,12,13]. Such procedures, permit the allocation of quantization noise throughout the frequency spectrum in a way that takes full advantage of any masking effects the signal being coded may offer, and can be easily modified to allocate bits in a way that meets the spectral requirements of the input signal, rather than meeting an arbitrary fixed bitrate criterion.

A very limited amount of work also exists in the speech coding literature on the variable rate coding of speech *for storage* [14]. Such research is the closest in spirit to the present work, and is directly applicable — requiring only modifications to account for non-speech waveforms, and for variable bandwidth material.

Thus, the coding of audio for archiving is a relatively new field, lacking existing literature. The introduction of economical mass storage media such as the optical disk, and the continuing drop in cost of computing power has however spurred interest in this area, particularly in the coding of speech for storage. The variable rate audio archiving algorithm presented in this work has thus been based on related work in the speech coding literature. Relevant results have been drawn from general coding, wideband coding, and variable rate coding literature to assemble an adaptive archiving algorithm based on previous research, but meeting the unique needs of a general purpose audio archive. Further details are given in the coding background information of Section 3, and in the specific algorithm description of Section 4.

Chapter 2 Design Principles

2.1 Design Motivation

As outlined in Section 1, audio archive collections are currently held on $\frac{1}{4}$ inch reel-to-reel analog magnetic tape media, which are subject to degradations due to aging, and must be regularly re-copied. Technological evolution has caused, and is continuing to cause, drastic reductions in cost, and great improvements in the capabilities of computing, optical storage, and digital signal processing devices. This three part evolution has sparked great interest in the audio archiving field in digital audio storage. Such digital storage, via longer–lived storage media, shows great potential for reducing the cost of re-copying collection material.

The following sections (2.2, 2.3, 2.4) will show that an optical disk based digital archive, in conjunction with the variable rate archiving algorithm proposed in Section 4, form an attractive and economically feasible alternative to analog audio archiving.

2.2 Algorithm Design Criteria

Given the special needs of the audio archive, and the greater flexibility for compression that audio *storage* offers over audio *transmission*, this section will outline the coder design criteria to be followed as the variable rate archiving algorithm is developed (sections 3 and 4). Emphasis is placed on those design elements which differ from standard practice in speech coding.

The traditional approach in the coding (or digitization) of any analog waveform, is to low-pass (anti-alias) filter the input signal at the highest frequency component of interest, f_C, in the signal, and then to sample the filtered signal at frequency f_S; where f_S is generally slightly larger than $2f_C$. This approach has been used with great success in the coding of fixed bandwidth speech and music. In an archive application however, variable fidelity material from mixed sources must be coded (eg. material from a compact disk [20–20,000 Hz], versus material from a wax cylinder [200–2,000 Hz]). This would require changes in f_C and f_S before, and perhaps during, the coding of each selection. These changes would be made by non-technical library personnel who would not necessarily fully understand the parameters being set.

It was thus decided that an automated approach was preferred. In this work, f_C and f_S will be fixed at:

$$f_C = f_C';$$

$$f_S = f_S';$$

where, f_C' and f_S' are the settings which would traditionally be used to code the highest bandwidth signal in the collection[1] . To reduce the inefficiencies that would otherwise result from over-sampling narrow bandwidth signals, the coding algorithm must therefore automatically adapt to changes in signal bandwidth, and not code non-existent high frequency signal components. The coding algorithm must effectively exploit the fact that a given signal of bandwidth f_b, has the same intrinsic information content, whether it is sampled at $f_S = 2f_b$, or $f_S \gg 2f_b$.

Criteria important to the design of such an audio archiving algorithm include:

1) **Variable rate coding**
   - the coder must adjust to the bandwidth of the source — storing only the information required to accurately reproduce the signal. This implies a variable rate coding scheme.

[1] Thus, to cover the entire range of human hearing, one would set, for example:

$$f_c' = 20,000\text{Hz};$$
$$f_s' = 44,000\text{Hz}.$$

In the coding algorithm simulations of Section 5, the values $f_c' = 7,500$ Hz, and $f_s' = 16,000$ Hz were used, due to practical limitations.

2) **High quality**

- to meet the needs of an audio archive, the decoded signal must be indistinguishable from the original. The quality of a modern $\frac{1}{4}$ inch reel-to-reel analog tape deck, running at 19 cm/s ($7\frac{1}{2}$ ips), must be realized.

3) **Realistic coding complexity**

- to be of practical use, the final coding and decoding algorithms must be implementable, in real time, on existing digital signal processing devices.

4) **Uneven coder/decoder workload**

- the decoding of the signal (i.e. the retrieval of the signal from storage for playback) will occur more frequently[1], and at more work stations than the encoding (i.e. the storage) of the signal. In fact, it is likely that only one coding/storage–capable station will be required. An algorithm that reduces complexity in the decoder at the expense of increased complexity in the encoder is therefore desirable to reduce archiving costs.

5) **No limits on coding delay**

- since all coded material will simply be stored for later retrieval, any reasonable delays introduced by the coding algorithm (eg. for buffering, windowing, or filtering) will not be noticed.

6) **Economic feasibility**

- the cost of implementing the coder must not exceed the cost of the extra storage space that would be required if the coder were not used.

2.3 System Design

In this section the potential hardware design of a digital archiving system, to be used in conjunction with the variable rate coding algorithm, is examined. Specifically, this section will discuss potential media to be used for the storage of audio selections after coding, and the equipment to be used in conjunction with the coder to locate, retrieve, play, and record specific selections from the collection. This equipment will of course be required, in addition to the coding algorithm, to form a working digital audio archive. It will be shown that an optical storage system in conjunction with the proposed variable rate archiving algorithm, offers an attractive and economically feasible alternative to traditional analog archiving.

[1] Except for the initial digitization of an existing analog collection.

2.3.1 Mass Storage Media

At the heart of the digital archiving system is a mass storage device (Figure 2.1). It is this device which holds audio selections after processing by the variable rate coding algorithm, and is responsible for the secure and efficient storage of all selections in the collection.

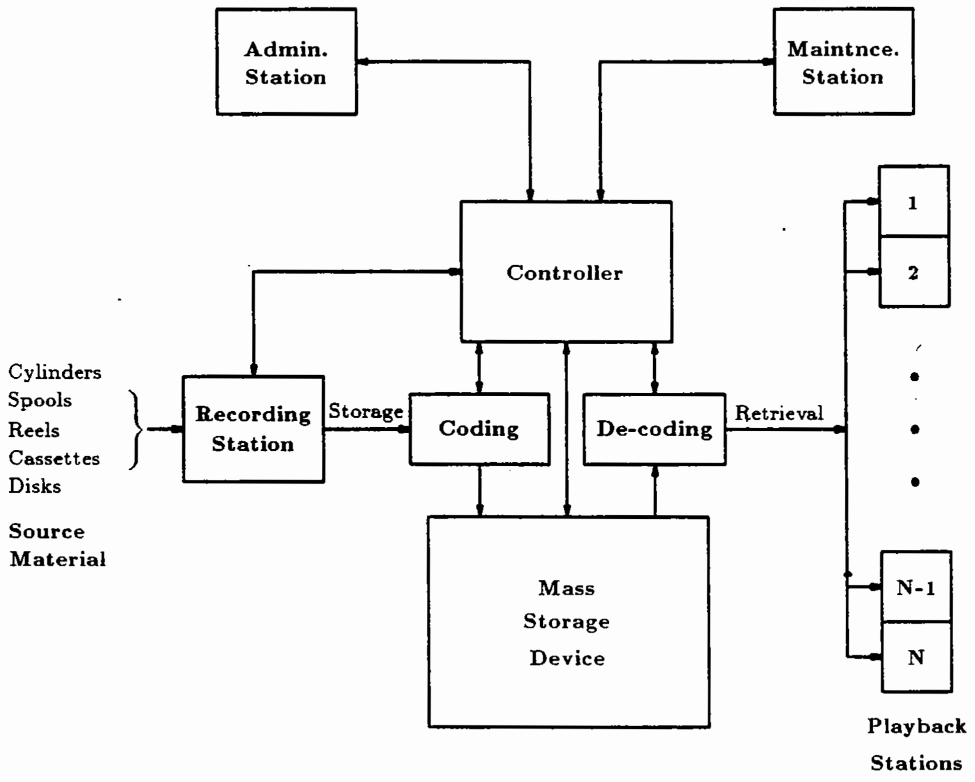


Figure 2.1 A Digital Archiving System

As previously stated, the traditional archive storage device is a high quality $\frac{1}{4}$ inch reel-to-reel tape deck, recording collection material on $7\frac{1}{2}$ inch analog magnetic reel-to-reel tapes.

In Table 2.1, the $\frac{1}{4}$ inch reel tape media is compared to four potential digital replacements: $5\frac{1}{4}$ inch floppy diskette, hard disk, 12 or 13 cm WORM (Write Once

Medium	Storage Type	Storage Format	Access	Capacity		Cost of Medium	Cost of Storage/Mb	Cost of Drive
				Mb	Hrs			
$\frac{1}{4}$ Inch Reel Tape	Analog	Magnetic	Sequential	115.2	0.5	\$5	\$0.043	\$3500
Floppy Diskette	Digital	Magnetic	Random	1.2	0.0052	\$2	\$1.67	\$250
Hard Disk	Digital	Magnetic	Random	40	0.174	—	\$50.00	\$2000
12/13 cm WORM	Digital	Optical	Random	600	2.6	\$75	\$0.17	\$3500
30 cm WORM	Digital	Optical	Random	2000	8.7	\$200	\$0.10	\$7500

Table 2.1 Archive Storage Media

Read Mostly) optical disk, and 30 cm WORM optical disk[1].

One potential digital medium noticeably absent from Table 2.1, and now becoming quite popular, is the CD-ROM (Compact Disk Read Only Memory). Although the CD-ROM appears cost competitive at \$500 per drive, and \$5–\$50 for a 600 Mbyte optical disk, its high mastering cost (\$10,000 per disk) makes it unsuitable for audio archive applications, where very few copies would be required of each disk.

In order to choose the optimum archival storage medium from those presented in Table 2.1, one must consider the following factors:

- Media Cost/Mbyte
- Physical Storage Cost
- Maintenance Costs
- Drive Cost/On-line Mbyte
- Durability
- Security
- On-line Capacity
- Longevity
- Speed of Access

Much interest is currently being shown throughout the audio archiving field in optical disk storage. Optical disk is in many ways an ideal medium for archiving. As can be seen from Table 2.1, the cost of storage per megabyte (Mb) of the optical

[1] The figures in Table 2.1 are representative of state of the art equipment commercially available at the time of writing. An average 64 kb/s coder bitrate was used to perform the megabyte(Mb)/hours capacity conversions.

media is not significantly higher than that of traditional analog magnetic tape; and this price differential will likely decrease as WORM disks begin to be mass produced. In addition, optical media do not suffer degradations due to aging to the same extent as analog tape — thus eliminating the expense of periodically re-copying collection material.

The optical disk also outperforms analog tape in the areas of durability, security, physical storage cost, and drive cost per on-line megabyte. Optical disks are much less susceptible to damage than analog tapes, providing resistance to breakage, jamming, and stray magnetic fields. The WORM disk, being a write-once medium, can not be changed or erased once written — providing high security against unauthorized, or authorized but incorrect, changes to archive material. WORM disk is a much more physically compact form of storage than analog magnetic tape, providing approximately 35.5 times the physical storage capacity ($\mathrm{Mb/m^3}$) of $\frac{1}{4}$ inch reel tape. This reduces stack shelving requirements, and other related physical storage costs. The high storage density of optical disk also leads to a higher on-line data capacity than analog tape, for an equivalent number of drives. Related to on-line capacity, is an optical disk's drive-cost/on-line-megabyte advantage: \$5.8/Mb for optical disk (13 cm), versus \$34.7/Mb for analog magnetic tape ($\frac{1}{4}$ inch reel). Finally, optical disk allows random access to archive selections, versus analog tape's slower sequential access.

Also included in Table 2.1 for comparison purposes are the traditional digital (computer) data storage devices: Hard disk, and Floppy diskette. Although ideal for general computer data storage, these devices are unsuitable for archive use due to their lower capacities, and high costs.

Thus, using the factors above as decision criteria, it is clear that optical disk is a logical choice as a digital replacement medium for $\frac{1}{4}$ inch reel-to-reel analog magnetic tape.

Although the 12 or 13 cm and 30 cm optical disks are more or less equally attractive for use in archiving, manufacturers are presently showing much greater interest in the 12 or 13 cm format. Thus, the economic feasibility analyses of Section 2.4, will be based on the use of a 13 cm WORM optical disk storage device.

2.3.2 System Configuration

In this section, the components required in addition to the mass storage device, and the variable rate archiving algorithm, to form a digital archiving system, will be introduced. The goal of this section is not to provide a detailed system design, but to give an overview of the state of the art equipment and devices currently available commercially, and demonstrate the potential for assembling a digital archiving system. The economic feasibility of the proposed system will be examined in Section 2.4.

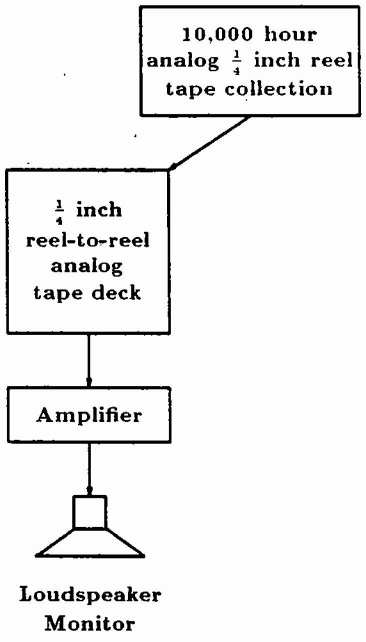

10,000 hour analog $\frac{1}{4}$ inch reel tape collection
$\frac{1}{4}$ inch reel-to-reel analog tape deck
Amplifier
Loudspeaker Monitor

Figure 2.2 Analog Archive

The existing analog archive configuration is given in Figure 2.2. To duplicate the functions of this system digitally, three modules are envisioned:

1) A control module to interface with the user, performing cataloging and indexing functions (i.e. providing the mechanism through which the user

locates and chooses the selection of interest). The control module would also oversee the retrieval and storage of selections from/to the optical disk for playback/storage.

2) A numerically intensive calculation module, to implement the variable rate archiving algorithm.

3) An audio module to perform analog to digital (A/D), and digital to analog (D/A) conversions, and to condition the signal before recording, or playback. The audio module would also include microphones, speakers, headphones, and their associated amplification circuitry.

Many specialized digital signal processing (DSP) devices exist to perform the functions of module 2. These DSP devices are essentially specialized microprocessors designed to perform very fast additions, multiplications, and shifts, through the use of parallel architectures. One line of such devices, the TMS 320 series by Texas Instruments, has been assembled by several manufacturers onto IBM-PC compatible plug-in circuit boards, in conjunction with A/D and D/A converters, memory, and clock signals (a list of such manufacturers is provided in Appendix B). The addition of anti-aliasing filters, and appropriate audio amplifiers to such a plug-in circuit board would meet the needs of module 3. Since IBM compatible WORM optical disk drives also exist, one convenient choice to perform the functions of module 1, is an IBM-PC compatible computer.

It is therefore possible to base the digital archiving system on an IBM-PC compatible controller equipped with a plug-in TMS320 DSP card, and connected to an IBM-PC compatible 13 cm WORM optical disk drive. Such a system is shown in Figure 2.3.

The system of Figure 2.3 could be expanded to include multiple user stations, and to provide increased storage capacity through additional WORM disk units by the addition of a high speed Local Area Network (LAN). The LAN would link user stations to the appropriate WORM disk unit, and could also provide access to a

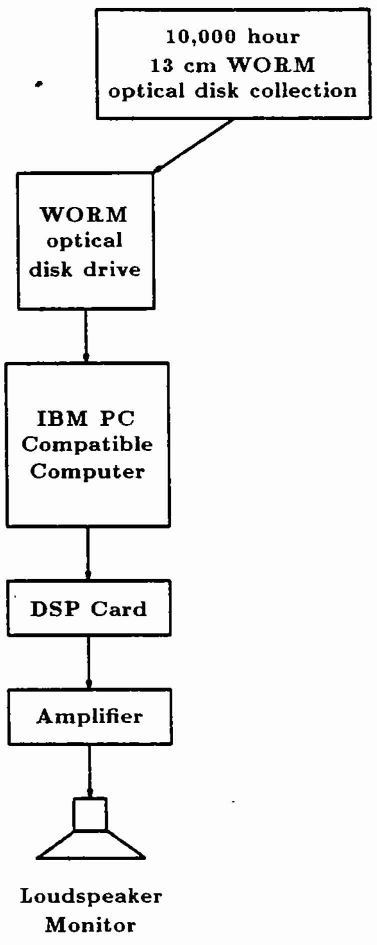


Figure 2.3 Digital Archive

centralized collection-index data base. Maintenance/Diagnostic, and Administration units could also be connected to the LAN as required by system growth.

In practice, the user would use the PC keyboard and screen display, to select the item of interest from the central cross-referenced collection index. A link would then be established over the LAN to transmit the required audio data from optical disk storage, to the appropriate user station. Throughout the playback process, a two way communications link between the user station and the system controller would remain open to enable the user to stop and start the playback, to fast forward or rewind, or to perform filtering, enhancement, or other real time processing functions.

2.4 Economic Feasibility

In this section, a brief economic feasibility analysis is performed on replacing the analog archive system of Figure 2.2, with the digital archive system of Figure 2.3. The feasibility of coding the audio data before storage is then examined. Two statements are proven:

1) Digital archiving using optical disk storage is an economically viable alternative to traditional analog archiving;

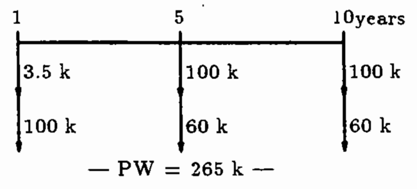
2) For the forseeable future, optical storage costs and densities will remain at the point where coding of the signal before storage is highly desirable in an archiving application.

2.4.1 Analog vs. Digital Archiving

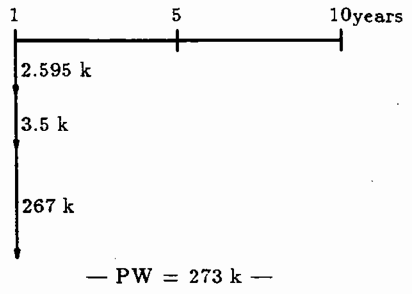
The following economic analysis is based on the small, single-user-station archive systems of Figure 2.2 (analog), and Figure 2.3 (digital).

A static 10,000 hour collection is assumed. The study period is 10 years. During this time it is assumed that the analog reel tapes must be replaced and re-recorded at the 5 and 10 year marks. The digital and analog systems are compared on a present worth basis, assuming an average interest rate of 10% per annum. The components presented in figures 2.2 and 2.3 are all commercially available at the present time. State of the art, cost effective equipment was chosen in each case. Details are provided in Appendix A.

Cash flow diagrams for the analog and digital systems are presented in Figure 2.4a and Figure 2.4b respectively. From these diagrams, the present worths are:



(a) Analog System



(b) Digital System

Figure 2.4 System Cashflows

Analog System

$$\begin{aligned}\text{PW} &= (\$100\text{k} + \$3.5\text{k}) + (P/F, 10, 5)(\$100\text{k} + \$60\text{k}) + (P/F, 10, 10)(\$100\text{k} + \$60\text{k}) \\ &= \$104\text{k} + (0.62092)(\$160\text{k}) + (0.38555)(\$160\text{k}) \\ &= \$265\text{k}\end{aligned}$$

Digital System

$$\begin{aligned}\text{PW} &= \$2.595\text{k} + \$3.5\text{k} + \$267\text{k} \\ &= \$273\text{k}\end{aligned}$$

where, the notation $(P/F, i, N)$ denotes that P is the present worth of a future amount F, at a point N compounding periods into the future, given an interest rate of $i\%$ per compounding period [15].

The present worth figures show that the digital system is approximately 3% more costly than the analog system over the 10 year study period. This simple feasibility analysis however, does not take into account the following factors:

- The digital system provides a storage medium that is 35 times more compact than the analog system, thus greatly reducing shelf space and other physical storage costs. These savings were not included in the analysis.

- Over the long term, due to the costs of re-copying analog tape, the digital system becomes increasingly attractive. For instance, over a 20 year period, the digital system is 16% *less* costly than the analog system.

- The cost of WORM optical disk drives and media will likely drop as these devices leave their introductory phase, and begin to be commonly mass produced.

- The digital system offers benefits such as random access to stored material (versus sequential access for the analog system), greater on-line storage capacity (2.6 times larger) than the analog system, and other benefits such as the potential for variable playback speed or other signal processing or enhancement functions, that are not easily assigned a monetary value.

- The collection is assumed static in the study above, but would typically be growing at about 10% per year, in a true archive. At the end of the 10 year study period, the collection would thus be roughly 2.5 times larger than it was at year 1. The costs of re-copying this extra material would quickly outweigh the lower storage costs per megabyte in the analog system, thus making it less economically desirable.

Given the small cost differential between the analog and digital systems, and the other factors above, it is clear that digital archiving is a feasible alternative to traditional analog archiving, both from an economic and operational point of view.

2.4.2 Coding vs. Straight Storage

An important goal in the development of the variable rate coding algorithm was to ensure that the cost of implementing the coder did not exceed the cost of the extra storage space that would be required if the coder were not used. This goal has been achieved. The test file data in Section 5.5 show that the proposed variable rate archiving algorithm reduces a linear-PCM bitrate of 256 kb/s (16 kHz sampling rate at 16 bits per sample) to an average rate of 59.2 kb/s.

The only additional cost associated with coding, is the DSP board required at each user station to perform the required coding/decoding calculations.

Using the cost data of Appendix A, and the bitrates above, and considering the single user digital archive system presented in Figure 2.3, it can be shown that coding is advantageous, and will continue to be of benefit until the cost of a 13 cm optical disk falls well under $1. The variable rate archiving algorithm reduces the number of optical disks required by 76.8%, which in the case of a 10,000 hour archive provides savings of 11,808 disks, or $885.6 k. This more than offsets the $2.6 k cost of the required DSP board.

At the current cost of a 13 cm optical WORM disk ($75), an archive would have to contain less than 30 hours of material for coding to be un-economical. For large archives there is little doubt that the cost of optical WORM disks will never fall to the point where coding is not warranted.

In addition to directly reducing the amount and cost of the storage media required, coding will also reduce the physical space needed to store a given amount of archival material. In addition, coding will allow each drive to provide more data on-line.

It is quite clear that any digital archive of practical size will benefit from coding.

2.5 Design Summary

It has thus been shown that digital archiving is an economically feasible alternative to traditional analog archiving, and is one which can be implemented using commercially available components. It has also been shown that the coding of audio signals before storage is highly desirable from an economic standpoint.

Design criteria for an appropriate archival coding algorithm have been given. This algorithm will be developed in sections 3 and 4.

Chapter 3 Coding Algorithm Background

Using the design criteria of Section 2, this section will provide general audio coding background, and will outline the choices made from the vast array of existing speech coding research, to form the variable rate archiving algorithm of Section 4.

3.1 Audio Coding

Coders are used, in general, to remove redundancies from an audio waveform, and thus to allow efficient transmission or storage of its coded bitstream. Efficiency in this context, denotes the goal of the highest possible signal quality, at the lowest possible bitrate.

The extent to which these redundancies exist is clearly illustrated by considering that the English language contains only 40 phonemes. At an average length of 80 ms per phoneme, the information content of speech can thus theoretically be transmitted at less than 100 bits/second. However, a basic coding algorithm in wide use throughout the North American telephone system, log-PCM, requires 64,000 bits/second to transmit long distance ("toll") quality speech. This 640:1 difference in bitrate clearly shows the potentially great gains to be made through efficient coding in the case of speech. This comparison is not quite fair, since the 64,000 bit/s signal carries more than just the pure "information content" of speech; it does show however, that speech is a highly redundant signal.

Increasing the bitrate of a coded signal improves the accuracy of the the signal representation, and thus improves the fidelity of the decoded signal. To increase coding accuracy *without* an increase in bitrate, the complexity of the coder must be increased so it can more efficiently exploit signal redundancies. This increase in complexity implies increased hardware needs and larger computational capacity, resulting in increases in the cost of implementing the coding algorithm.

Thus, in choosing the appropriate coding algorithm for a particular application, tradeoffs must be made amongst three interrelated factors: bitrate, complexity, and cost. These tradeoffs will be discussed in this section in the context of a coding algorithm suitable for audio archiving.

As outlined in Section 2, the audio archives of concern in this work are composed of both speech and music — typically equal quantities of each. Although there is a vast amount of ongoing research into speech coding [3,16], virtually no work at all has been specifically tailored towards music.

Fortunately, speech and music have similar properties, and intelligent choices from the available array of speech coding options lead easily to an algorithm suitable for both sources.

In understanding these properties, the spectogram is an invaluable tool. In Figure 3.1, a wideband spectogram of the spoken phrase "She burns toast" is given. The spectogram is a three dimensional plot, presenting frequency on the vertical axis, time on the horizontal axis, and intensity as the darkness of the display. Thus, a large amount of energy persisting in a certain frequency band for a long period of time will appear as a dark horizontal bar.

Speech waveforms can be broken down into two broad categories, both of which are visible in Figure 3.1: sibilants (voiced speech) and fricatives (un-voiced speech). Sibilants (phonemes such as /e/, /i/, /m/, /w/) use the periodic vibrations of the vocal cords as their sound source. The line spectrum produced by the vocal cords is shaped by the vocal tract (tongue, teeth, lips, palate, etc.) to produce a spectrum

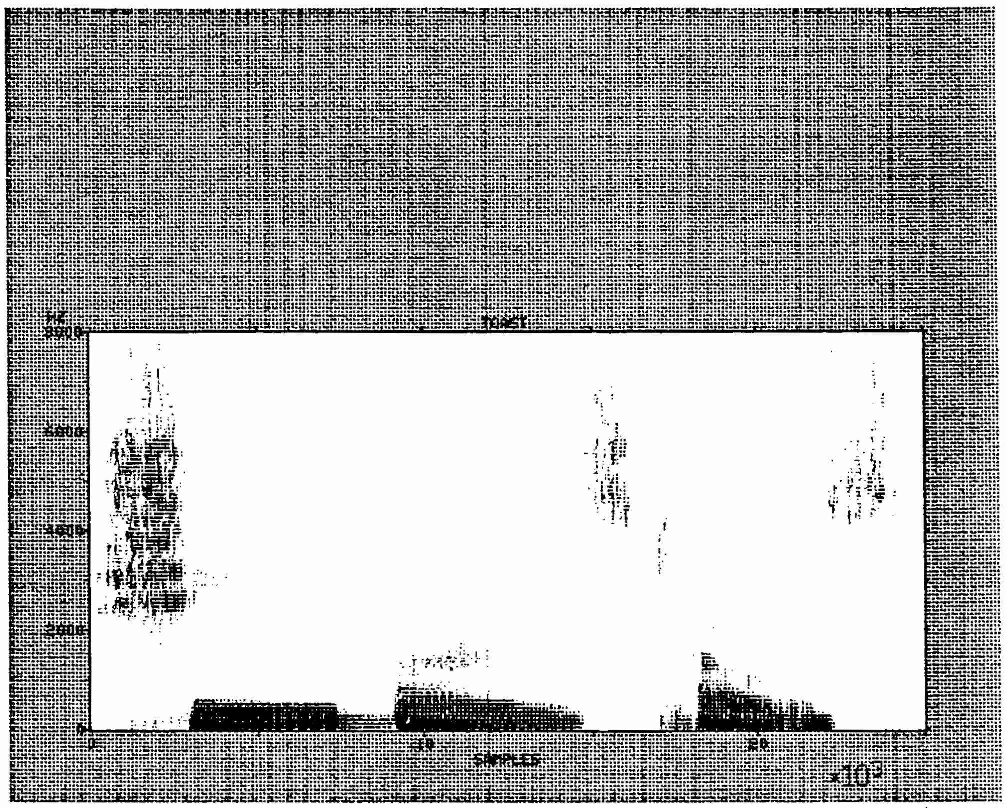


Figure 3.1 Audio Spectogram: "She Burns Toast"

containing resonances. Such resonances are known as formants, and appear in Figure 3.1 as a vertical series of roughly horizontal bars, such as those seen between the 18,000–22,000 sample mark of Figure 3.1, corresponding to the "oa" in "toast".

In Figure 3.2, the spectogram of a series of piano notes shows the same formant structure. In this case, the spectrum is formed by a vibrating steel string source, and is shaped by the piano cabinet; however, the piano spectrum is remarkably similar to that of the voice sibilant, the major differences being:

- greater energy in the high frequency formants than seen in voice spectra;
- purer and more clearly defined formants than seen in voice spectra;
- very sudden spectrum changes; the piano notes have a much faster attack than voice phonemes, and often a much longer sustain.

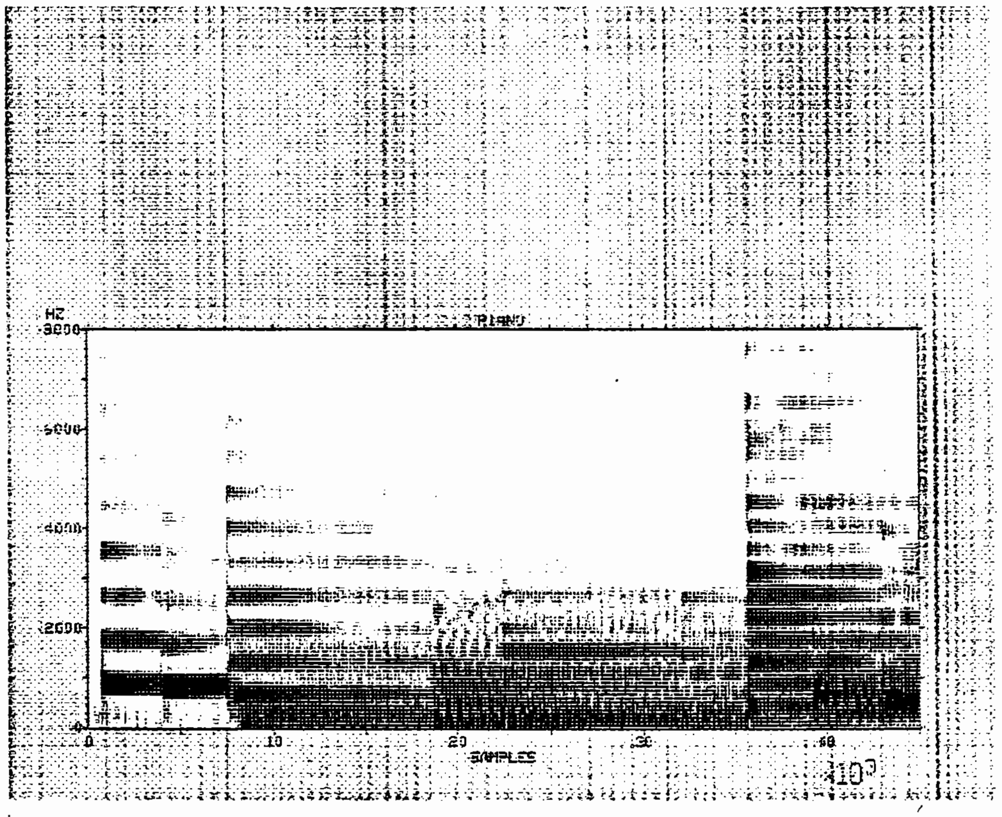


Figure 3.2 Audio Spectogram: Piano

The majority of non-percussive (and some percussive) instruments exhibit similar strong formant spectra (eg. the trumpet of [17]).

The second speech category, fricatives (phonemes such as /f/, /s/, /v/, /h/), use turbulent air flow as a sound source, which is again spectrally shaped by, in this case, the upper vocal tract. This results in a broadband high pass spectrum such as that seen at the 0 – 2800 sample mark of Figure 3.1, corresponding to the "sh" of "she". Again, a musical equivalent to the fricative exists as seen in Figure 3.3 — the spectogram of a cymbal crash.

Thus, speech and music are quite similar, in that they are to a large extent both derived from, and relatively well-modeled by, line spectrum and white noise sources spectrally shaped by a resonant cavity.

At this point, a decision can be made between two broad classes of coders: *Wave-*

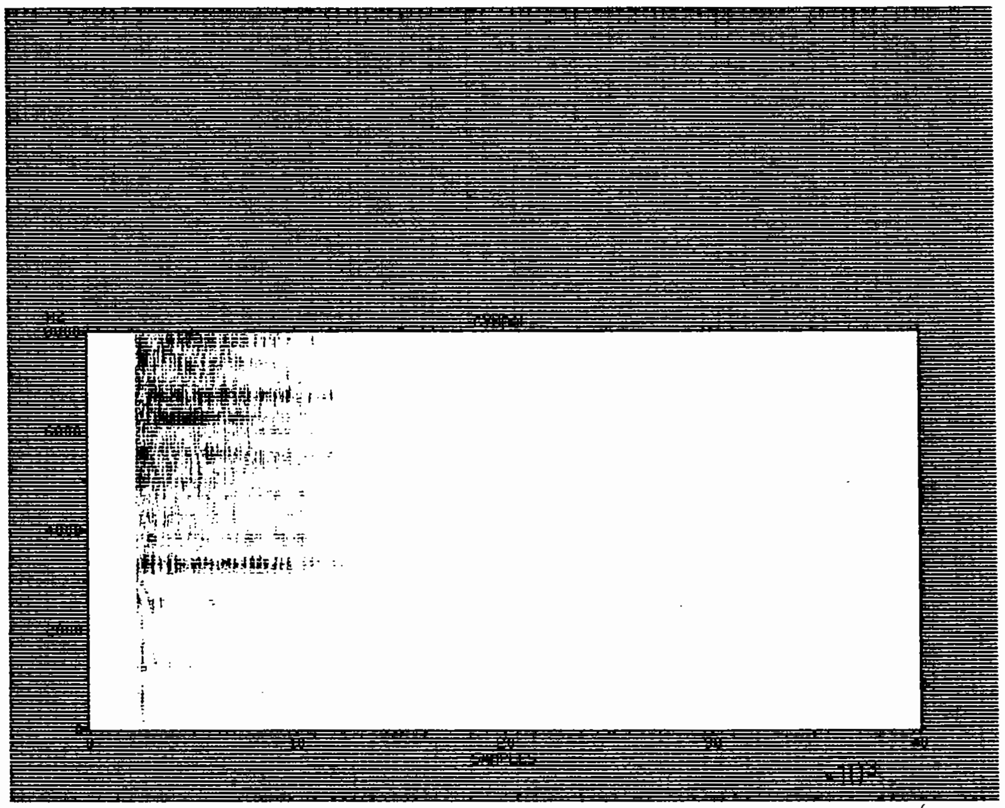


Figure 3.3 Audio Spectogram: Cymbal

form coders, and *Source* coders (also known as Vocoders).

Waveform coders are designed to reproduce, as precisely as possible, a given audio waveform. They reconstruct the signal on a sample by sample basis, and after estimating, transforming, or predicting a signal value, will generally transmit or store the difference between the estimate and the original signal (the residual).

Source coders, on the other hand, assume an audio source model, and code the signal in terms of the parameters of that model. If the model does not accurately match the source involved, coding will be poor. If it does match well, great economies in storage and transmission can be realized. Residuals are not transmitted or stored in this case. Coding is also not performed on a sample by sample basis. Large blocks of code may be processed before determining model parameters.

In speech applications, source coders can achieve much lower bitrates than waveform coders while maintaining speech intelligibility, but source coders tend to perform poorly under non-ideal conditions (i.e. with background noise), are often speaker dependent, and tend to sound artificial.

Given the design criteria of Section 2.2, and the needs of the audio archive, it is clear that a waveform coder is required. No single model could accurately describe the wide variety of source material to be coded. Archiving also requires high quality and robustness. Thus, a waveform coder is the natural choice.

3.2 Time Versus Frequency Domain Coders

Given that a waveform coder is required, a second broad choice may now be made between *time* and *frequency* domain coders.As shown in Table 3.1, many algorithms of each type exist.

Time Domain	Frequency Domain
Pulse Code Modulation (PCM)	Sub-band Coder (SBC)
Log-PCM (Log-PCM)	Adaptive Transform Coder (ATC)
Adaptive PCM (APCM)	Time Domain Harmonic Compression (TDHC)
Differential PCM (DPCM)	
Adaptive Differential PCM (ADPCM)	
Delta Modulation (DM)	
Adaptive DM (ADM)	
Adaptive Predictive Coder (APC)	

Table 3.1 Time vs. Frequency Domain Coding Algorithms

Time domain coders act directly on the digitized speech samples to either directly quantize the source (PCM, Log-PCM, APCM), or to quantize the source via a prediction process (DPCM, DM, ADM, ADPCM, APC). More efficient coders will adapt parameters to match the current statistics of the signal being coded (APCM, ADM, ADPCM, APC).

Frequency domain coders operate in the frequency domain via digital filtering (SBC), or through a mathematical transform (ATC, DHC). All of the frequency domain coders in Table 3.1 adapt parameters to match source statistics.

Many other coders have been described in the literature (an up-to-date summary is provided in [4]); only the most common are presented in Table 3.1.

Given the design criteria of Section 2.2, and the wide variations in fidelity of typical archive material, a frequency domain coder was clearly indicated. Such a coder could efficiently sense the bandwidth of the material being recorded, and adjust its bitrate accordingly. Only the information required to accurately reproduce the original signal would thus be stored, leading to great economies in archive storage space.

Two frequency domain algorithms from Table 3.1 were considered to be adequately researched, and particularly well suited for use in an audio archive application: the adaptive transform coder, and the sub-band coder.

It was decided that the adaptive transform coder could provide the high frequency resolution required for variable fidelity archiving, at a lower complexity, and with less numerical calculation, than the sub-band coder. Both of the latter two factors would be significant in achieving criterion 3 of Section 2.2 — that of physical implementability on existing processors. The adaptive transform coder (ATC) algorithm was therefore chosen as the basis for the variable rate archiving algorithm presented in Section 4.

It is interesting to note that the Comité Consultatif Internationale de Télégraphique et Téléphonique (CCITT) is currently standardizing a wideband (64 kb/s) sub-band coding algorithm with the same bandwidth as that used during the simulation testing of Section 5 [1]. This algorithm was designed with different goals in mind, however, having a fixed rather than a variable bitrate, and being designed specifically for fixed bandwidth speech[1], with music as only a minor consideration.

A comparison made in Section 5.7 between the CCITT sub-band coder, and the variable rate archiving algorithm of Section 4, shows that the silence deletion and variable bitrate properties of the proposed algorithm allow it to outperform the

[1] Provision is also made for the transmission of up to 16 kb/s of data.

CCITT coder for narrowband signals, and to provide the higher quality required in an archiving application (albeit at an increased bitrate) for wideband signals.

3.3 The Adaptive Transform Coder

In this section, the operation of the traditional adaptive transform coder (ATC) will be described, and an archive-optimized collection of its variations assembled, based on research published in the speech coding literature.

The modifications made to the traditional ATC coder to provide variable rate, are given in the detailed algorithm description of Section 4.

3.3.1 General

Early work on transform coding was performed by Campanella and Robinson [18], and by Wintz [19], in the early 1970's. The adaptive transform coder, in the form to be used in this work, was introduced by Zelinski and Noll in 1977 [7], and has been modified and improved since then by several others [6, 20, 21].

An overview of the ATC coding scheme is given in Figure 3.4.

In ATC coding, it is the transform coefficients which are quantized and stored, rather than the time domain signal samples themselves. The ATC coder also adapts its quantization parameters to the short term statistics of the input signal. This results in a much higher quality than could be achieved by a fixed parameter coder, but requires that the parameters be stored as side information along with the main transform coefficient bitstream.

As shown in Figure 3.4, audio signal samples are collected into blocks by the input buffer before processing. A smooth spectrum estimate of the input signal is then made based on each block of input samples, as described in Section 3.3.4, and stored via the side information bitstream. The buffered data block is also transformed (typically using a discrete cosine transform), and quantized to form the main information bitstream. Each transform coefficient has a separate quantizer, individually adapted

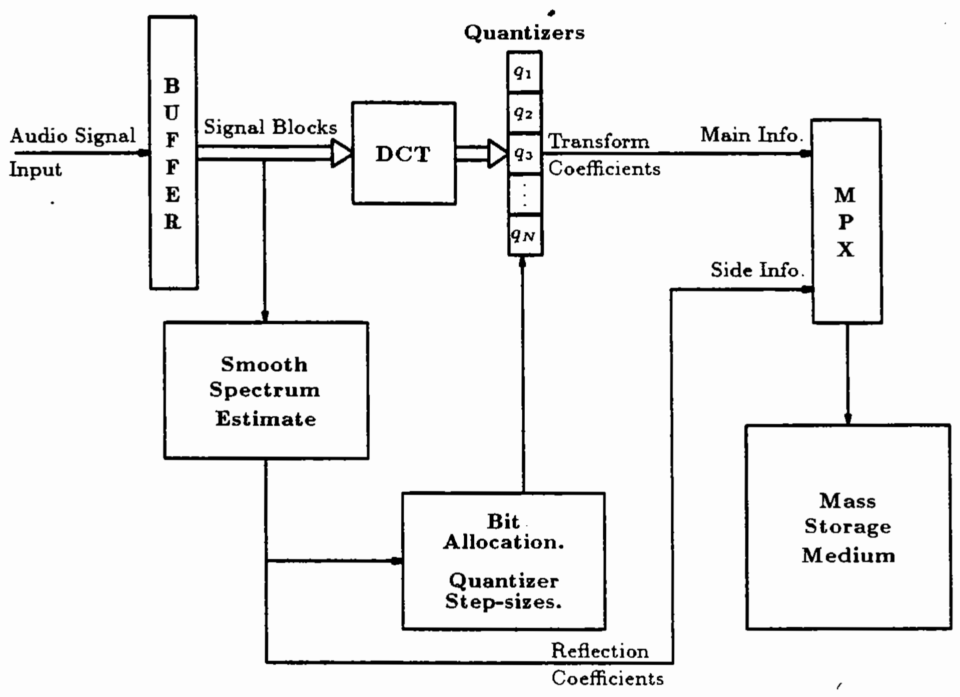


Figure 3.4 ATC Coder Overview

to the statistics of that coefficient. The individual quantizer step-sizes and bit allocations are derived from the smooth spectrum estimate. The main information and side information bitstreams are then multiplexed (MPX), and written to the mass storage medium.

When decoding the signal, a similar but reversed process is used. The side and main information bitstreams are de-multiplexed, and the smooth spectrum estimate is recovered from the side information. This allows the quantizer step-sizes and bit allocations to be re-generated, and thus the transform coefficients to be recovered from the main information bitstream. The decoded output samples are then available after inverse transforming the recovered transform coefficients.

3.3.2 The Transform Algorithm

The first step in assembling an efficient transform coder is the choice of an appropriate transform algorithm.

The efficiencies of the transform coding approach rely on transforming the input signal into a set of uncorrelated coefficients in the transform domain. These coefficients may then be quantized individually with independent quantizers optimally suited to the statistics of each particular coefficient, with no loss in overall coder performance. Independent quantization also allows quantization error (noise) to be spread throughout the frequency spectrum in the manner least offensive to the human ear. This can reduce the perceived effects of quantization noise. Thus, the performance of a transform coding scheme is dependant on:

- efficient quantization of the transform coefficients (requiring intelligent bit allocation and quantizer step-size calculation strategies);
- an efficient de-correlating transform algorithm.

Quantization of the transform coefficients will be discussed in Sections 3.3.5 to 3.3.7.

De-correlation of the input signal before quantization is a way of exploiting signal correlations (redundancies) to achieve a higher signal to noise ratio after quantization than would have been possible with straight quantization of the time domain signal. Thus, using the same average number of bits per sample in each case, a signal quantized after transformation will be of higher quality than the same signal quantized without transformation.

For a perfect de-correlating transform, the theoretical gain in signal to noise ratio to be made by quantizing after transformation, G_{TC}, has been shown to be [7, 22]:

$$G_{TC} = \frac{\frac{1}{N}\sum_{k=0}^{N-1} X^2(k)}{\left[\prod_{k=0}^{N-1} X^2(k)\right]^{1/N}};$$

where, $X(k)$, for $0 \leq k \leq N-1$ are the transform coefficients; and, N, is the length of the input signal block.

The gain is thus the ratio of the arithmetic and geometric means of the transform coefficients. Signals with "peaky" spectra (i.e. signals possessing strong resonances/formants) thus will benefit more from transform coding than signals with flat spectra (i.e. white noise, where $G_{TC}=1$, and no gain is achieved).

Previous work has shown that the transform algorithm required to provide total de-correlation of the transform coefficients is the Karhunen-Loeve Transform (KLT), [7]. The KLT algorithm, however, is dependent on signal statistics, and the derivation of its transform matrix elements is a complex process. In addition, no fast algorithms exist for the computation of KLT transform coefficients. The KLT thus fails to meet Criterion 3, of Section 2.2 (i.e. the requirement that the coding algorithm be implementable in real time, on existing digital signal processing devices), and a less numerically intensive transform of roughly equivalent performance must be chosen.

Such an algorithm is realized in the Discrete Cosine Transform (DCT). Studies of several algorithms (the Karhunen-Loeve, Walsh-Hadamard, discrete slant, discrete Fourier, and discrete cosine transforms) have shown, that the DCT approaches the performance of the KLT for large block sizes (on the order of 128 samples), [7]. The Discrete Fourier Transform (DFT) also approaches the performance of the KLT, though not as rapidly (the signal to noise ratio (SNR) performance of the DCT is 4 to 5 dB higher than the DFT at 128 samples/block [7]).

Other work has shown that the DCT produces fewer boundary effects than the DFT, leading to reduced "click" and "burbling" distortion at the block rate [6]. Finally, the DCT can be shown to have the same spectral envelope as the DFT spectrum [6]. The DCT spectrum is thus directly related to the true frequency properties of the input signal, and facilitates spectral noise shaping via efficient bit allocation.

The DCT is, for these reasons, the transform of choice in the traditional adaptive transform coder, and has been chosen for use in the variable rate archiving algorithm of Section 4.

The N-point discrete cosine transform of an input signal, $x(n)$, is given by [23]:

$$X(k) = c(k) \sum_{n=0}^{N-1} x(n) \cos\Big[\frac{(2n+1)\pi k}{2N}\Big] \qquad \text{for} \quad 0 \leq k \leq N-1; \tag{3.1}$$

and the inverse transform given by:

$$x(n) = \frac{1}{N} \sum_{k=0}^{N-1} X(k) c(k) \cos\Big[\frac{(2n+1)\pi k}{2N}\Big] \qquad \text{for} \quad 0 \leq n \leq N-1; \tag{3.2}$$

where in both cases,

$$c(k) = \begin{cases} 1, & \text{for } k = 0; \\ \sqrt{2}, & \text{for } 1 \leq k \leq N-1. \end{cases}$$

3.3.3 Windowing

An important consideration in the spectral processing of an input signal on a block-by-block basis, is how best to window each input block. An adaptive transform coder, in processing a signal block is in effect performing a short-time spectral analysis of the input signal at the time of the block. In taking a short-time data block, the "infinite length" input signal, $x(n)$, has been multiplied by a rectangular analysis window, $w(n)$, of the form:

$$w(n) = \begin{cases} 0, & \text{for } -\infty < n < 0; \\ 1 & \text{for } 0 \leq n \leq N-1; \\ 0 & \text{for } N \leq n < \infty; \end{cases} \tag{3.3}$$

where, N, is the length of the input block.

Letting the discrete Fourier transform (DFT) of $x(n)$ be $X(k)$, and the DFT of $w(n)$ be $W(k)$, where the DFT, as usual is defined as:

$$X(k) = \sum_{n=0}^{N-1} x(n) e^{\frac{-j2\pi nk}{N}} \qquad \text{for} \quad 0 \leq k \leq N-1; \tag{3.4}$$

the DFT of the input block, $w(n)x(n)$, will be $X(k) * W(k)$ [24], where $*$ denotes circular convolution. Thus, the magnitude of the DFT of the input signal block provides not the spectrum of the "true" input signal, but the spectrum of the input signal convolved with the spectrum of the window, $w(n)$. For optimal frequency resolution, the window $w(n)$ should thus ideally be chosen such that its spectrum is a single impulse at a radian frequency of $\omega = 0$. This would however, result in a window of infinite length in the time domain, and the window would no longer provide the desired "sample" of local signal characteristics. Window design thus involves tradeoffs between time and frequency domain performance. Many compromise window designs exist in the digital signal processing literature [25, 26].

All windows except the rectangular window of Equation (3.3) require overlapping of adjacent input blocks, and a corresponding increase in bitrate, during signal synthesis. At low bitrates, it has been shown in the speech coding literature [26], that windowing reduces distortion due to block end effects. Given the high quality required for archiving however, no significant end effects are expected, and windowing will not provide benefits to the same extent.

Windowing however, also reduces spectral smearing. This is an important consideration in archiving, where narrowband signals will often be over-sampled. If these signals were to spread beyond their original bandwidths, bits will be assigned needlessly to extraneous high frequency components.

An ideal window for speech archiving would therefore offer an appropriate compromise between the zero-overlap rectangular window, and the superior frequency resolution and resulting decrease in spectral smearing offered by smoother windows. Such a compromise is conveniently found in the adjustable form of a cosine rolloff window (See Figure 3.5).

This window has therefore been chosen for use in the variable rate archiving algorithm of Section 4. The cosine rolloff window, $w(n)$, is given by:

$$w(n) = \begin{cases} \dfrac{1}{2\left[1 - \cos\left(\dfrac{\pi n}{M+1}\right)\right]} & \text{for } 0 \le n \le M-1; \\[2ex] 1 & \text{for } M \le n \le N-M-1; \\[2ex] \dfrac{1}{2\left[1 - \cos\left(\dfrac{\pi(M+1-n)}{M+1}\right)\right]} & \text{for } N-M \le n \le N-1; \end{cases} \tag{3.5}$$

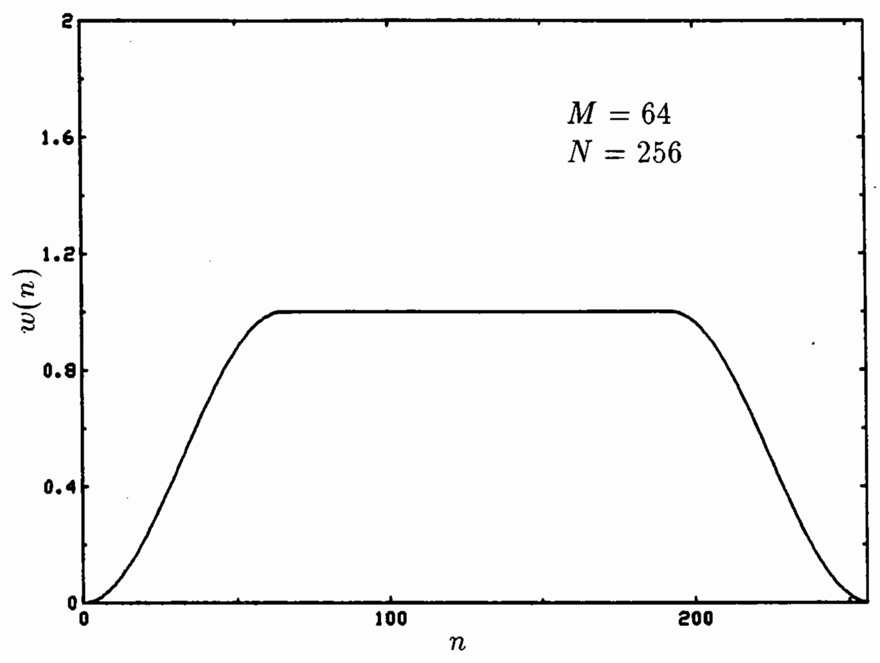


Figure 3.5 Cosine Rolloff Window

where the parameter M can be varied to control the number of samples which overlap between successive blocks.

3.3.4 The Smooth Spectrum Estimate

As outlined in Section 3.3.1, the adaptive transform coder requires that a compact description of the input signal spectrum be stored once per block as side information. To meet this requirement, the assumption is made that the input signal spectrum can be approximated by a smooth spectrum curve. This smooth spectrum estimate can be stored much more compactly than the original spectrum, and will be used by both the coder and the decoder to calculate adaptive bit allocations and quantizer step-sizes, and thus to quantize/reconstruct the transform coefficients.

To preserve the efficiency of the ATC coding scheme, the amount of information required to store the spectrum estimate must be kept as small as possible. Three methods for generating a space efficient smooth spectrum estimate have been proposed in the ATC literature:

1) The Log-Average Estimate [7];
2) The Linear Prediction Estimate [6];
3) The Homomorphic Estimate [20].

The Log-Average estimate was proposed by Zelinski and Noll in their introductory paper on ATC coding, [7], in 1977. As shown in Figure 3.6, this scheme averages a group of neighbouring spectral values to produce a single value representative of the entire group.

The spectral curve to be averaged is composed of the transform coefficients squared, σ_k^2, (Figure 3.6a):

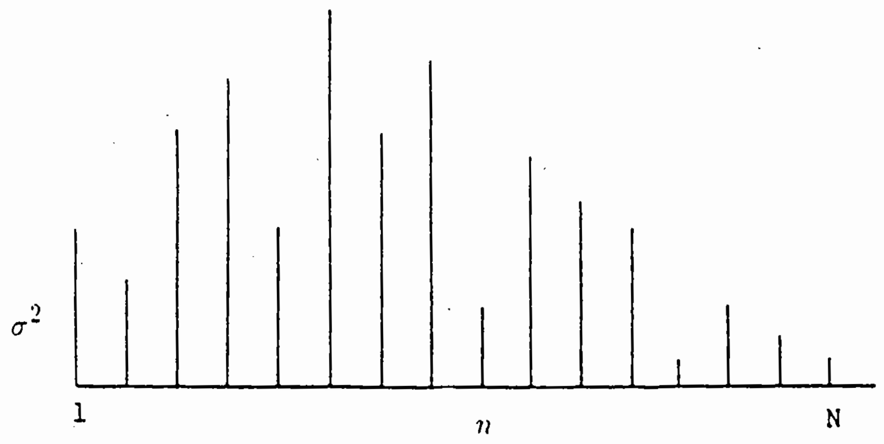
$$\sigma_k^2 = \log\left[X^2(k)\right] \qquad \text{for } 0 \leq k \leq N-1; \tag{3.6}$$

where, $X(k)$ is the set of transform coefficients (Equation (3.1)), and N is the transform block length. The curve is then split into small segments, and representative average basis values, $\hat{\sigma}^2$, (Figure 3.6b), are calculated, quantized, and stored as side information. In the ATC scheme of [7], a 128 point transform was represented by 16 basis values of $\hat{\sigma}^2$. Thus, groups of 8 coefficients were averaged, and through coarse quantization, the total side information bitrate was held to 2 kb/s (2 bits/basis value at an 8 kHz sampling rate). The final smooth spectrum estimate, $\hat{\hat{\sigma}}^2$, (Figure 3.6c) is then derived by linearly interpolating between the basis values $\hat{\sigma}^2$.

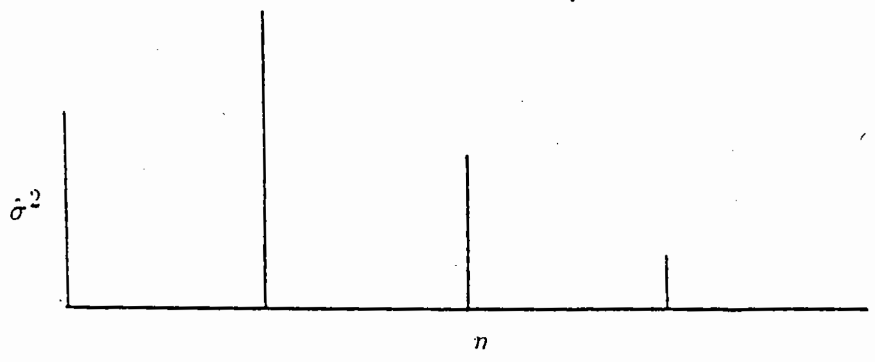
Although the Log-Average method is effective and simple, more efficient spectrum smoothing techniques have been developed since its introduction. One such technique is based on Linear Predictive Coding (LPC). This technique uses an autoregressive model to simulate the spectral behaviour of the input signal.

In the LPC approach, advantage is taken of the fact that speech, and as outlined in Section 3.1, music signals, are characterized primarily by resonances [27]. Thus, an all-pole signal model is used, with a transfer function of the form:

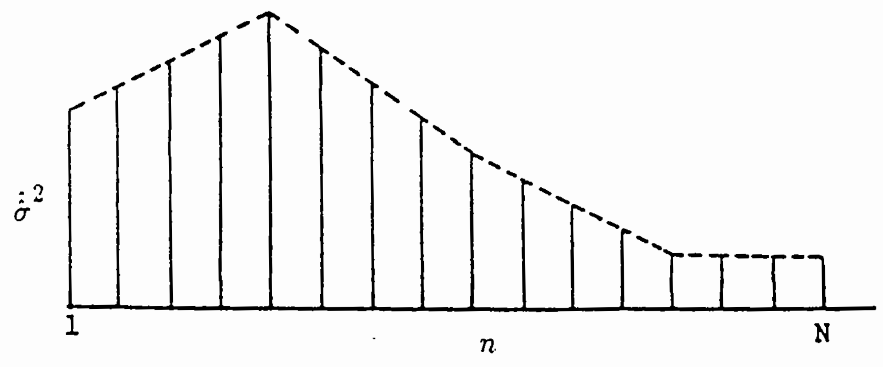
$$H(z) = \frac{G}{1 - \sum_{k=1}^{N_p} a(k) \cdot z^{-k}}; \tag{3.7}$$



(a)



(b)



(c)

Figure 3.6 The Log-Average Spectrum Estimate

where, the $a(k)$ are a set of linear prediction coefficients, G is a scaling factor, N_p is the number of predictors used, and z is the complex domain variable of the mathematical z-transform [28,29], which maps a discrete sequence $x(n)$ onto $X(z)$ such that:

$$X(z) = \sum_{n=-\infty}^{\infty} x(n) \cdot z^{-n}.$$

The response of $H(z)$, calculated as: $|H(z)|$ at $z = e^{j\omega}$, is the desired smooth spectrum estimate.

Pole-zero signal models, with transfer functions of the form:

$$H(z) = G \cdot \frac{1 + \sum_{l=1}^{N_q} b(l) \cdot z^{-l}}{1 - \sum_{k=1}^{N_p} a(k) \cdot z^{-k}}; \qquad (3.8)$$

provide superior performance relative to the all-pole model of Equation (3.7), but do so at the expense of greatly increased complexity. For the all-pole model, optimal values (in the sense of achieving a minimum mean square prediction residual error) of the coefficients $a(k)$ can be found through the solution of a set of linear equations. In addition, efficient recursive algorithms exist to find this solution[1]. The full pole-zero model however, generally requires the solution of a set of non-linear equations. All-pole models have been proven adequate for speech coding, since speech production is a fundamentally all-pole process. Although archive material other than speech may well contain significant zero content, the transfer function of Equation (3.7) will be used in this work, to minimize complexity.

The first step in producing the LPC smooth spectrum estimate is to find the values, $a(k)$, which satisfy as exactly as possible, the N_p-pole prediction equation:

$$\hat{x}(n) = \sum_{k=1}^{N_p} a(k) \cdot x(n-k) \qquad \text{for } 1 \leq n \leq N. \qquad (3.9)$$

The input signal $x(n)$, is thus to be predicted from a weighted linear combination of its past values.

[1] For example, the Levinson/Durbin recursion of equation set (3.12)

Since no real world signal will conform exactly to the impulse response of the N_p-pole filter, exact values for $a(k)$ will not exist, and compromise values, optimum in some sense, must be found.

One such compromise involves minimizing the mean square difference between the predicted value of the input signal $\hat{x}(n)$, and its true value $x(n)$. This requires that [30]:

for a minimum of $\mathrm{E}\left[\left\{x(n) - \hat{x}(n)\right\}^2\right]$,

$$\frac{\partial}{\partial a_k}\left[\mathrm{E}\left\{[x(n) - \hat{x}(n)]^2\right\}\right] = 0;$$

for all predictor coefficients a_k. Thus,

$$\mathrm{E}\left[2\left\{x(n) - \hat{x}(n)\right\} \cdot \frac{\partial}{\partial a_k}\left\{-\hat{x}(n)\right\}\right] = 0;$$

and,

$$\mathrm{E}\left[\left\{x(n) - \hat{x}(n)\right\} \cdot x(n-k)\right] = 0;$$

where $\mathrm{E}[y]$ denotes the expected value of y.

Thus the orthogonality principle[1] holds, and it can be shown [30] that optimal values for $a(k)$ are chosen if:

$$r(j) = \sum_{k=1}^{N_p} a(k) \cdot r(\,|\,j - k\,|\,), \qquad 0 \le j \le N-1; \qquad (3.10)$$

where $r(j)$ is the autocorrelation function of the input signal, calculated as:

$$r(j) = \frac{1}{N}\sum_{n=0}^{N-1} x(n) \cdot x(n+j) \qquad 0 \le n \le N-1. \qquad (3.11)$$

Although optimal values of $a(k)$ can be calculated from Equation (3.10) through straightforward matrix manipulations, or through a triangular matrix decomposition

[1] For a minimum mean square error, the prediction coefficients must be chosen so that the resulting prediction error, is orthogonal to the data used to calculate the coefficients.

[31]; as indicated above, an efficient recursive algorithm introduced by Levinson and modified by Durbin exists [27].

Solution by Levinson/Durbin recursion expresses predictor coefficient values for a system model of order i, in terms of predictor coefficient values for a model of order $i-1$. The recursion begins at $i = 1$ and continues step by step until $i = N_p$, using the following set of equations:

$$\begin{aligned}
k(i) &= \frac{\sum_{j=1}^{i-1} [a_{i-1}(j) \cdot r(i-j)] - r(i)}{\epsilon^2(i-1)} \\
a_i(i) &= -k(i) \\
a_i(j) &= a_{i-1}(j) + k(i) \cdot a_{i-1}(i-j) \qquad \text{for} \quad 1 \leq j \leq i-1; \\
\epsilon^2(i) &= \left[1 - k^2(i)\right] \cdot \epsilon^2(i-1)
\end{aligned} \tag{3.12}$$

The process is initialized by setting $\epsilon^2(0) = r(0)$, so that $k(1) = -r(1)/r(0)$. The final set of optimized predictor coefficients is given by:

$$a_{N_p}(j) \qquad\qquad 1 \leq j \leq N_p.$$

At each stage of the recursion, the current mean square prediction error $\epsilon^2(i)$, is produced as an intermediate result. Another intermediate result is the set:

$$k(i) \qquad\qquad 1 \leq i \leq N_p;$$

of reflection coefficients (also known in the statistical literature as partial correlation coefficients). The $k(i)$ are restricted numerically to the range $-1 \leq k(i) \leq +1$, and are thus convenient to quantize. For this reason, it is traditionally the set $k(i)$ that is quantized (usually after a non-linear transformation) and sent as side information, rather than the predictor coefficients $a(k)$.

Given a set of predictor coefficients $a(k)$, the smooth spectrum estimate (to be realized as the response $|H(z)|$ of Equation (3.7)) can now be found via the following three step process:

1) The response, $d(k)$, of the denominator of $H(z)$ is found through a $2N$-point DFT of the sequence $a'(k)$:

$$d(k) = \left| \mathrm{DFT}\left[a'(k)\right] \right| ;$$

where,

$$a'(k) = \begin{cases} 1, & \text{for } k = 0; \\ -a(k), & \text{for } 1 \leq k \leq N_p; \\ 0, & \text{for } N_p+1 \leq k \leq 2N-1; \end{cases}$$

the DFT is performed as given in Equation (3.4); and N represents the length of a block.

2) The response, $V(k)$, corresponding to $H(z)$, is found by inverting $d(k)$:

$$V(k) = \frac{1}{d(k)}, \qquad 0 \leq k \leq N-1.$$

3) $V(k)$ is scaled so that the total energy of the true spectrum $X^2(k)$ equals the total energy of the smoothed estimate $v(k)$, (i.e.) so that:

$$\sum_{k=0}^{N-1} X^2(k) = \sum_{k=0}^{N-1} v(k).$$

Then,

$$v(k) = G \cdot V(k), \qquad 0 \leq k \leq N-1; \tag{3.13}$$

where,

$$G = \frac{\sum_{k=0}^{N-1} X^2(k)}{\sum_{k=0}^{N-1} V(k)} . \tag{3.14}$$

and, $X(k)$, is the set of DCT transform coefficients (Equation (3.1)).

The final smooth spectrum estimate is thus $v(k)$.

A modification to the LPC estimate procedure as described to this point, was proposed by Tribolet and Crochiere in 1979 [6]. This enhancement adds a pitch model to represent pitch striations in the speech spectrum. For archiving purposes however, it was not clear that the pitch striations present in speech would also occur to the same extent in other audio signals. It has also been shown, [26], that improvements due to the pitch model are often speaker dependent. Thus, due to the complexity added by the pitch calculations, this enhanced linear prediction procedure was not considered in the present work.

The third smooth spectrum estimating procedure proposed in the ATC literature uses a homomorphic vocoder model to develop a cepstral spectral estimate. In this approach, published by Crochiere and Cox in 1981 [20], a pseudospectrum, $c(n)$, of the input signal is calculated once per block as the inverse SDFT of the sequence $X'(k)$, derived from the transform coefficients, $X(k)$, as:

$$\begin{aligned} X'(k) &= \log\Big[\,|X(k)|\,\Big] \qquad \text{for} \quad 0 \le k \le N-1; \\ c(n) &= \mathrm{SDFT}^{-1}\Big\{X'(k)\Big\}. \end{aligned}$$

The SDFT (Symmetric Discrete Fourier Transform) of an N–point sequence $x(n)$, is defined as the $2N$–point DFT of the sequence $x'(n)$, where:

$$x'(n) = \begin{cases} x(n), & \text{for } 0 \le n \le N; \\ x(2N-n), & \text{for } N+1 \le n \le 2N-1. \end{cases}$$

Thus, the SDFT is a real to real transform.

A cepstrum approximation, $c'(n)$, is then formed by retaining the first 10 to 14 values of $c(n)$, and discarding the rest. The $c'(n)$ values are quantized and stored as side information. To recover the smooth spectrum estimate, the sequence $c'(n)$ is padded with zeros and SDFT'd to produce a smoothed version of the original log magnitude spectrum $X'(k)$. In [20], $X'(k)$ is used directly to logarithmically quantize the transform coefficients, and to compute the quantizer step-sizes.

There are thus three spectrum smoothing methods potentially suitable for use in an audio archive transform coder. For coder bitrates in the range 9.6 – 16 kb/s, the homomorphic and LPC methods have been found to be clearly superior to the log-average scheme [6, 20]. Above this range very little work has been done, but reports of similar performance [6], and of slightly improved performance for the LPC over the log-average scheme [26] can be found.

Despite its increased complexity over the log-average method, the LPC scheme was chosen for use in the variable rate algorithm of Section 4, for the following reasons:

- The LPC physical model is based on resonances, which provides a storage framework closer to the physical mechanism of speech and music production than the other models;
- The LPC model provides the best fit of any of the options presented above, in areas of high spectral energy (i.e. near formants), where the signal must be encoded most accurately;
- The use of reflection coefficients provides a highly efficient means for storing the LPC spectrum estimate;
- The LPC method has been well researched in the speech coding literature and efficient methods have been developed for its implementation;
- The LPC method, using the final prediction error (FPE) criterion introduced in Section 4, is conveniently adjustable to the variable bitrate scheme required for archiving.

3.3.5 Bit Allocation

The allocation of bits amongst the transform coefficients is of major importance in ATC coding. This allocation controls the distribution of quantization noise in the frequency domain, and thus the perceptual quality of the coder.

3.3.5.1 Optimal Bit Assignment

In order to determine an optimal bit assignment, it is generally assumed that the transform coefficients can be approximated by an independent identically distributed Gaussian source. As outlined in Section 3.3.2, this assumption is reasonable if the coder transform (in this case the DCT) efficiently de-correlates the input signal samples. Hence, the transform coefficients are viewed as a column vector of N samples taken from a continuous source, which are to be individually quantized in a way that minimizes the overall mean square error incurred in quantizing the block. Given this assumption, results from rate distortion theory show that the optimal bit assignment for a Gaussian source is [22]:

$$b(i) = \frac{B}{N} + \frac{1}{2}\log_2\left[\frac{\sigma^2(i)}{D}\right]\ ; \tag{3.15}$$

where,

$b(i)$ is the optimal number of bits to be assigned to transform coefficient $X(i)$;

$\sigma^2(i)$ is the variance of transform coefficient $X(i)$, which will be approximated by: $\sigma^2(i) \approx X^2(i)$;

D is the average mean square distortion incurred in quantizing the block;

B is the total number of bits available for quantizing the block;

N is the length of the block.

Given that the mean square distortion incurred in quantizing transform coefficient i is $d(i)$, it can be shown, that for a minimum overall block mean distortion [32]:

$$D = d(i) \qquad\qquad \text{for} \quad 1 \le i \le N;$$

and,

$$D = \frac{1}{N}\sum_{i=1}^{N} d(i).$$

Thus the distortion incurred in quantizing each transform coefficient must be equal, to achieve a minimum distortion level for the block.

The optimal bit assignments thus take the form [7]:

$$b(i) = \frac{B}{N} + \frac{1}{2}\log_2\left[\frac{X^2(i)}{\left[\prod_{j=1}^{N} X^2(j)\right]^{1/N}}\right]\ ; \tag{3.16}$$

or equivalently,

$$b(i) = \frac{B}{N} + \frac{1}{2}\log_2\Big[X^2(i)\Big] - \frac{1}{2N}\sum_{j=1}^{N}\log_2\Big[X^2(j)\Big]. \tag{3.17}$$

Note that the optimal bit assignment thus sets the number of quantizer levels assigned to coefficient i, proportional to the variance of coefficient i. Equation (3.17) is thus attempting to assign equal step sizes to each quantizer, and allowing more quantizer levels for coefficients with larger variances.

The bit assignments of Equation (3.17) will generally be non-integer, and may even be negative. A procedure based on Equation (3.17) which provides optimal bit assignment when the $b(i)$ are constrained to be integers, has been proposed in [33]. This method is based on the mean square error distortion, $e(i)$, resulting from the quantization of a Gaussian distributed coefficient with an i bit optimum quantizer (see Equation (3.19)). The values $e(i)$ are tabulated in [34]. An array of differential distortion values, $e_d(i)$, is derived from $e(i)$ as:

$$e_d(i) = e(i) - e(i+1)$$

where i ranges from 1, to the maximum number of bits to be assigned to any one coefficient, $b_{\max}$. The $e_d(i)$ thus represent the decrease in quantization distortion that would occur if $i + 1$ bits were to be used rather than i bits to quantize a unit variance source. In practice, bit assignment is an iterative search procedure, where the marginal returns

$$R(i) = X^2(i) \cdot e_d\Big[b(i)\Big]$$

are calculated for each transform coefficient, $X(i)$ before assigning each bit; and, $b(i)$ is the number of bits currently assigned to coefficient $X(i)$ at the time of this calculation. The coefficient with the largest value of $R(i)$ is then assigned a bit, its $R(i)$ value updated, and the search procedure continued. The iterations continue until all bits have been assigned. This procedure assures that at each stage, bits are assigned to the transform coefficient that will benefit most (in the sense of showing the largest decrease in quantization distortion) from an additional bit: This ensures that a global minimum mean square error will be achieved for the block.

Although this integer–optimized procedure works well, it is also time consuming due to the search of N marginal return values that must be performed before the assignment of each bit.

A reduced–iteration, rounded form of bit assignment based on Equation (3.17) has thus been adopted for use in the variable rate algorithm of Section 4. Test results show that this compromise solution reliably provides bit assignments similar to that of the integer optimized method, with considerably less computation. Further details are provided in Section 4.3.3.

3.3.5.2 Spectral Noise Shaping

Although the bit allocations of Section 3.3.5.1 are optimal in the mean square error sense, they will not necessarily lead to the highest quality coded signal as perceived by human listeners. In fact, it is known that shaping the quantization noise spectrum to take advantage of the masking effects of the input signal will result in a higher perceived coded signal quality [6], without an increase in bitrate.

Such spectral noise shaping can be achieved through the use of a bit assignment weighting factor, generally resulting in a revised bit allocation equation of the form (from Equation (3.17)):

$$b(i) = \frac{B}{N} + \frac{1}{2}\log_2\left[X^{2u}(i)\right] - \frac{1}{2N}\sum_{j=1}^{N}\log_2\left[X^{2u}(j)\right] . \tag{3.18}$$

The weighting factor u, takes on values in the range:

$$0 \leq u \leq 1 .$$

For $u = 1$, the bit allocation is identical to that of Equation (3.17), and the quantization noise spectrum is flat. For $u = 0$, the bit assignment is constant, and the noise spectrum exactly follows that of the input signal. This revised bit allocation will be used in the variable rate algorithm of Section 4.

3.3.6 Quantizer Step Size Adaptation

As outlined in Section 3.3.1, the ATC coder quantizes each transform coefficient individually with a quantizer adapted on a block-by-block basis to the short-term statistics of each coefficient. In high quality ATC coding, it is reasonable to quantize the transform coefficients with a probability density function (PDF)–optimized uniform (linear) quantizer based on work published by Max, [34]. This work examined the tradeoffs to be made between granular and overload distortion when choosing an optimum quantizer step-size.

In [34], the optimum (in the sense of producing a minimum mean square quantization error) step-size, $\delta(i)$, is given for a unit variance source; where i is the number of bits to be used in the quantization. Values of $\delta(i)$ are given for various source PDF's. Measurements made of ATC transform coefficients show they tend to be Gaussian distributed in speech [7]. It is reasonable to assume, given the comparisons presented in Section 3.1, that the Gaussian approximation holds as well for music. Thus the optimized Gaussian Max step-size values listed in Table 3.2 were chosen for use in the variable rate algorithm of Section 4.

# Bits/coefficient i	Optimum Step-size $\delta(i)$
1	1.5956
2	0.9957
3	0.5860
4	0.3352
5	0.1881
6	0.1041
7	0.0569
8	0.0308

Table 3.2 Optimum Quantizer Step-Sizes

Given then that the number of bits to be used, $b(k)$, in quantizing transform coefficient k is derived from Equation (3.18), and the variance estimate, $v(k)$, is known from Equation (3.13), the quantizer step-size, $\Delta(k)$, to be used in quantizing coefficient k is therefore drawn from Table 3.2 as:

$$\Delta(k) = Q \cdot v(k) \cdot \delta\left[b(k)\right]. \tag{3.19}$$

The loading factor Q, was proposed by Tribolet and Crochiere in [6], as a way of directly controlling the granular/overload noise distribution. By altering Q around the Max–optimum value of 1.0, a greater proportion of overall quantization distortion can be allocated to the overload ($Q < 1$), or granular ($Q > 1$) modes. This permits compensation for the possibility that the minimum mean square error step-size is not necessarily the step-size which leads to minimum perceived distortion. Alterations to Q can also compensate to some extent for transform coefficients which are not well modeled by a Gaussian source.

3.3.7 Quantization of Main and Side Information

Both the main and side information bitstreams must be quantized before being multiplexed and written to the mass storage medium. The ATC coder, as realized in this work, has in total, four parameters to quantize:

1) the transform coefficient set, $X(k)$;
2) the reflection coefficient set, $k(i)$;
3) the predictor count, N_p;
4) the block scaling factor, V.

Quantization of the transform coefficients has been fully discussed in sections 3.3.5 and 3.3.6.

The block scaling factor is quantized as a base 2 value, consisting of a b_m bit mantissa, and a b_e bit exponent. The values chosen for the parameters b_e, and b_m, are given in Section 5.2.

The predictor count, being an integer value between 1 and $N_{\max}$, is directly quantized using b_p bits. The value chosen for b_p is given in Section 5.2.

Quantization of the reflection coefficients is a much more difficult problem which has received much attention in the speech coding literature. Two popular reflection coefficient transforms — log-area ratios, and inverse-sine functions — were compared to linear quantization in [35]. These transforms attempt to remove the sensitivity to quantization that reflection coefficients exhibit as they approach +1 or -1, by transforming the coefficients in a fashion that expands the region near ± 1. A linear quantization can then be performed on the transformed coefficients with reduced spectral sensitivity, and the resulting values inverse transformed [36]. In [35], it was shown that log-area ratios and inverse-sine functions perform similarly. Both improved upon linear quantization.

Many other methods have been proposed for representing the smooth spectrum estimate [36,37]. Only one method, recently receiving much attention, has been shown to outperform the log-area quantization of reflection coefficients.

This is the so called Line Spectrum Pair (LSP) method, which has shown a 30% reduction in side information bitrate in a 9.6 kb/s multi-pulse linear predictive coder [38]. LSP however, in order to determine the LSP parameters, requires that the roots of two equations, each of order $N_p/2$, be found[1]. For values of $N_p < 8$, non-iterative solutions exist. For higher order models, a trial and error search must be performed.

For use in an audio archive application where the complex musical spectra found in the test files of Section 5.3 often require large (> 12) numbers of predictors for an accurate spectral estimate, it is clear that log-area ratio (or inverse-sine function) quantization is preferable to the more complex LSP method. Log-area quantization has been chosen for use in the variable rate algorithm of Section 4.

The log-area transform function, $f(x)$, is given by:

$$f(x) = \log\left[\frac{1+x}{1-x}\right] \qquad \text{for} \quad a \le x \le b; \tag{3.20}$$

where,

$$a > -1;$$

$$b < +1.$$

[1] Where N_p is the number of predictors used to model the system.

Histograms are available in the speech coding literature which allow optimal choices of a and b for the quantization of each reflection coefficient [35]. The number of bits required to quantize each coefficient based on fixed spectral deviation limits, has also been published [35]. This data, in conjunction with histographic analyses of the music files of Section 5.3, was used as a guide in setting the log-area quantizer ranges and bit allocations given in Section 5.2 (Table 5.2).

3.4 Summary

A frequency domain waveform coder has therefore been shown to be appropriate for use in audio archiving. In particular, the adaptive transform coder has been chosen. Input signal blocks will be smoothed with a cosine rolloff window, and a compact estimate of the input signal spectrum will be formed through linear prediction, and stored using a set of log-area quantized reflection coefficients.

The coder will be based on a non-adaptive discrete cosine transform. Transform coefficients will be quantized using a linear Max quantizer, optimized for a Gaussian source. Optimal bit allocations will also be calculated assuming a Gaussian source, then rounded to the nearest integer. Spectral noise shaping will be performed through an exponential weighting of each transform coefficient's variance estimate, during the bit allocation process.

Chapter 4 The Variable Rate Archiving Algorithm

In this section, the traditional Adaptive Transform Coder as presented in Section 3, is modified to meet the needs of a general audio archive. In particular, methods are presented for achieving variable bitrate through:

1) Silence Deletion;
2) Signal Bandwidth Estimation;
3) Signal Order Estimation.

These modifications will enable the variable rate archiving algorithm to greatly reduce required audio storage space for the variable fidelity, and variable source material found in a typical archive.

4.1 Algorithm Overview

Figure 4.1 is a block diagram of the variable rate archiving algorithm. The source signal $x(n)$ is sampled at a rate f_s after suitable anti-alias filtering, and buffered into blocks of length N. The input signal block is then windowed (Equation (3.5)) by a cosine rolloff window, $w(n)$:

$$x_w(n) = x(n) \cdot w(n) \qquad \text{for} \qquad 0 \leq n \leq N-1. \tag{4.1}$$

After windowing, the input signal block is pre-emphasized to boost weaker high frequency components. This results in a reduced high frequency noise level for the coder, and a decrease in spectral dynamic range, which minimizes quality degradations due to spectral spillover effects. The pre-emphasized signal, $x'(n)$, is obtained from $x_w(n)$ via:

$$x'(n) = x_w(n) - \alpha x_w(n-1). \tag{4.2}$$

Each sample is then normalized by dividing by the mean energy of the block, V, where:

$$\begin{aligned} V^2 &= \sum_{n=0}^{N-1} x'(n)^2; \\ x''(n) &= \frac{x'(n)}{V} \quad \text{for} \quad 0 \le n \le N-1. \end{aligned} \tag{4.3}$$

The mean energy scaling factor V, is quantized and stored once per block as side information using a base 2 representation with a b_m bit mantissa and b_e bit exponent.

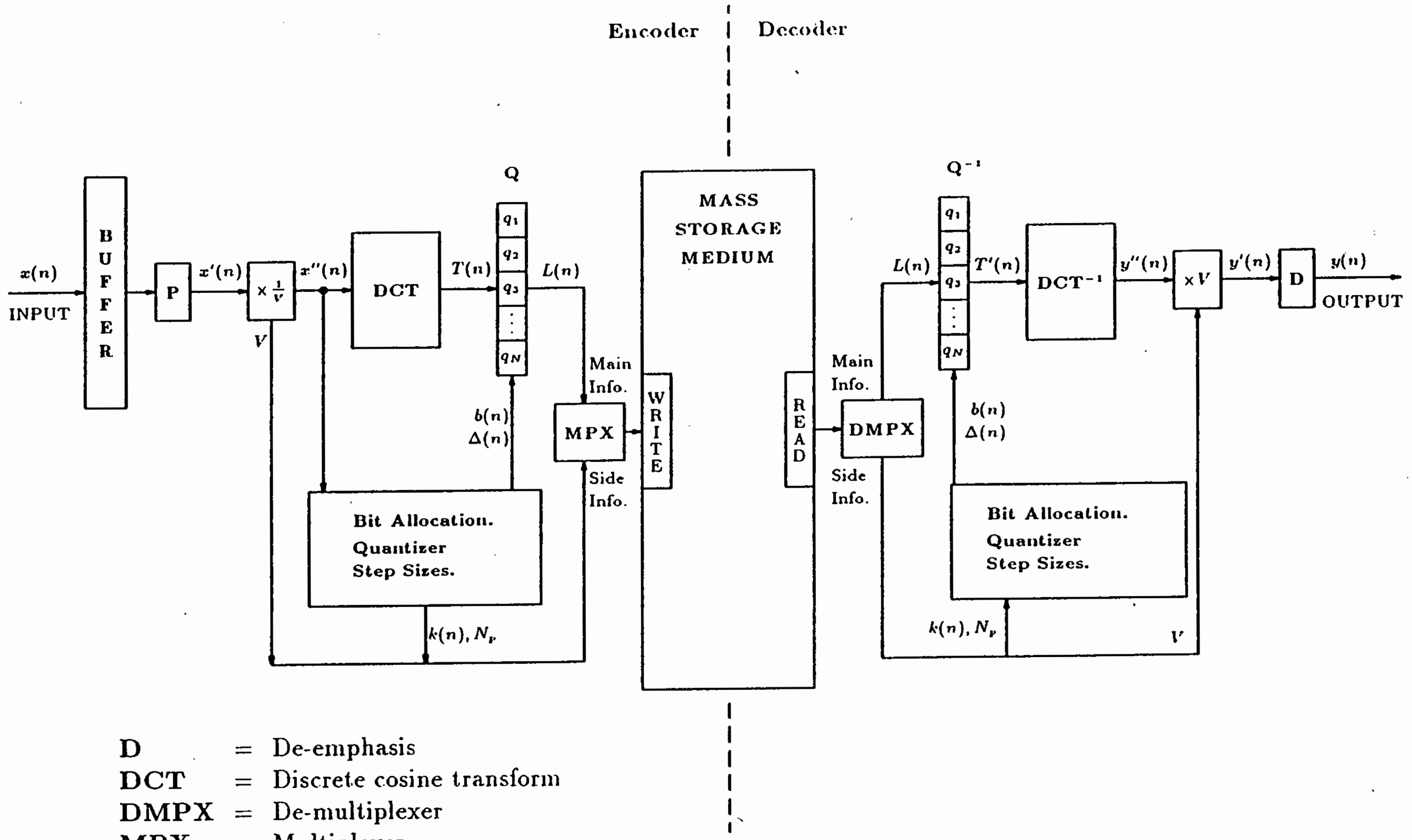
The normalized block of samples $x''(n)$ is then transformed using a DCT (Equation (3.1)) to provide a set, $T(n)$, of transform coefficients:

$$T(n) = \sum_{i=0}^{N-1} x''(n) \cdot c(i) \cdot \cos\left[\frac{(2i+1)\pi n}{2N}\right], \qquad \text{for} \quad 0 \le n \le N-1. \tag{4.4}$$

Side information in the form of a smooth spectrum estimate which is used by both the coder and the decoder to derive bit allocations, $b(n)$, and quantizer step-sizes, $\Delta(n)$, is calculated once per block and multiplexed with the main information bitstream before being written to the mass storage medium. The main information bitstream is composed of transform coefficient quantizer levels, $L(n)$, from which the decoder reconstructs the quantized transform coefficients, $T'(n)$, using the side information. The quantized transform coefficients are then inverse transformed by the decoder, and scaled to produce the decoded output signal $y'(n)$. This signal is then de-emphasized to give:

$$y(n) = y'(n) + \alpha y'(n-1). \tag{4.5}$$

The first M values of the sequence $y(n)$ are then added to (overlapped with) the last M values of the previous block, and the first $N-M$ samples of $y(n)$ written to the output buffer (the last M samples of $y(n)$ will be added to the first M samples of the next block, and written to the output buffer at that time).



D	=	De-emphasis
DCT	=	Discrete cosine transform
DMPX	=	De-multiplexer
MPX	=	Multiplexer
P	=	Pre-emphasis

Figure 4.1 Algorithm Overview

4.2 The Side Information Bitstream

The side information bitstream is used to store a smooth spectrum estimate which is used by both the coder and the decoder to calculate bit allocation and quantizer step-sizes.

4.2.1 The Smooth Spectrum Estimate

As outlined in Section 3.3.4, the smooth spectrum estimate is fully specified by a set of reflection coefficients:

$$k(i) \qquad\qquad 1 \leq i \leq N_p;$$

where N_p is the number of predictors used to generate the spectrum estimate. Equivalently, N_p is an estimate of the order of an appropriate autoregressive signal source model. The determination of N_p based on the latter interpretation will be discussed in Section 4.2.2. The reflection coefficients are found through a Levinson/Durbin recursion (equation set (3.12)). This requires an estimate of the autocorrelation function of the input signal. Such an estimate could be derived through the computation of Equation (3.11). This however, would give the autocorrelation function related to the "true" spectrum of the input signal. What is actually required for a smooth estimate of the DCT spectrum, is the autocorrelation function, $r(n)$, of the time sequence which has a power spectrum equal to that given by the block transform coefficient variance sequence $T^2(n)$. Since the inverse discrete Fourier transform (DFT^{-1}) of a power spectrum provides the required autocorrelation coefficients, $r(n)$ can be calculated as:

$$r(n) = Re\Big\{\mathrm{DFT}^{-1}[T_2^2(n)]\Big\};$$

where, $T_2^2(n)$ is a conjugate symmetric sequence of length $2N$ formed from $T^2(n)$ as:

$$Re\{T_2^2(n)\} = \begin{cases} Re\Big\{T^2(n)\Big\}, & \text{for } 0 \leq n \leq N-1; \\ Re\Big\{T^2(2N-1-n)\Big\}, & \text{for } N \leq n \leq 2N-1; \end{cases}$$

$$Im\left\{T_2^2(n)\right\} = Im\left\{T^2(n)\right\} = 0, \qquad \text{for} \qquad 0 \le n \le 2N-1.$$

(Re and Im denote the real and imaginary parts respectively of a complex sequence).

Although this real-to-real inverse DFT procedure provides the desired result, an equivalent and more elegant solution [39], is to calculate $r(n)$ directly as a circular autocorrelation of the input time domain sequence $x''(n)$:

$$r(n) = \frac{1}{2}\sum_{i=0}^{2N-1-n} x''(i)x''(i+n) + \frac{1}{4}\sum_{i=0}^{n-1}\left[x''(i)x''(n-1-i) + x''(N-n+i)x''(N-1-i)\right].$$

This direct approach was chosen since it avoids the use of the inverse DFT, and the squaring of the transform coefficients, which in an actual digital signal processor implementation, would lead to unnecessary loss of accuracy through truncation errors. In addition, only the first few values of $r(n)$ need be calculated, making the direct calculation approach more numerically efficient than an inverse DFT. Specifically, only the first $N_{\max}$ values of $r(n)$ are calculated, where $N_{\max}$ is the maximum number of poles to be used in the predictor calculations ($N_p \le N_{\max}$). The limit $N_{\max}$ is set using *a priori* knowledge of the signals to be coded, as outlined in Section 5.2.

4.2.2 Signal Order Estimation

In producing a smooth spectrum estimate through the linear prediction process, an autoregressive model of the input signal source is formed. An important consideration in linear predictive analysis is the determination of the number of predictors needed to adequately model the audio source. This is equivalent to determining the order of the audio source. Simply increasing the number of predictors without bound will obviously result in a constantly increasing estimate quality; but there is however, a point of diminishing return, where the improvement resulting from the addition of further predictors is not worth the extra bits required to store those predictors.

In speech coding, the human vocal tract is generally considered to be well modeled by a fixed number (8–12) of predictor stages [36, 4]. Thus, real-time variations in model order are not considered. In a variable rate archiving application however, where many different signal sources will be encountered (human voice, drums, piano, etc.), variation of model order to match the characteristics of the audio source, is a reasonable way to achieve an improved quality:bitrate ratio.

To find research on model order estimation (also known as system identification), one must turn to the mathematical, statistical, and control literature. A vast array of model order estimation techniques exist [40]. Since order estimation was not a main consideration of the current work, a basic technique proposed by Akaike [41] was chosen for implementation. Known as the final prediction error (FPE) criterion, this technique attempts to balance model bias against mean square prediction error in a systematic fashion. The FPE criterion given by:

$$\mathrm{FPE}(i) = \left[1 + \frac{i+1}{N}\right] \cdot \left[\frac{N}{N-1-i}\right] \cdot \epsilon_i^2 \qquad \text{for} \quad 1 \le i \le N_{\max}; \tag{4.6}$$

where,

ϵ_i^2 is the current prediction error (Equation (3.12));
N is the transform block length;
$N_{\max}$ is an upper limit on the range of model orders considered feasible given *a priori* knowledge of the signals to be coded;
i is the order of the model;

is calculated at each stage, i, of the Levinson/Durbin recursion. The array FPE(i) is then searched for a minimum, and N_p is set equal to the value of the index i which gives a minimum of FPE(i):

$$\begin{aligned} \mathrm{FPE}(j) &= \min\left\{\mathrm{FPE}(i)\right\} \qquad \text{for} \;\; 1 \le i \le N_{\max}; \\ N_p &= j. \end{aligned} \tag{4.7}$$

This procedure is repeated once per block to adaptively match the spectrum estimate to the statistics of the input signal. The value N_p is quantized and stored once per block via the side information bitstream as a b_p bit quantity.

4.3 The Main Information Bitstream

The main information bitstream, using the bulk of the ATC coder total bitrate, carries the quantized transform coefficients. The procedures used to perform the required quantizations at a variable rate adapted to the statistics of the input audio signal, and in a way which prevents any perceived distortions, will be presented in this section.

4.3.1 The Bandwidth Estimate

Two fundamental factors influence the modifications which must be made to the traditional speech-specific ATC algorithm for archive use:

- the presence of varied audio sources (i.e. sources other than the human vocal tract must be considered);
- the potential variations in fidelity (bandwidth) of these sources (eg. variations due to recordings made on modern, versus older vintage recording devices).

In Section 3.1, the basic similarity between the spectra of speech and music was shown. The fundamental assumptions of this section are:

- for a given signal bandwidth, speech and music can be coded with equivalent quality using roughly equivalent mainstream bitrates;
- for constant quality, the coder bitrate required is directly proportional to the bandwidth of the input signal.

The bitrate goal for a signal block is thus set based on an estimate of the bandwidth of that block.

This estimate is formed by finding the transform coefficient $T^2(j)$, where

$$T^2(i) \leq \frac{N\lambda}{100} \qquad \text{for} \quad j \leq i \leq N{-}1. \tag{4.9}$$

The value j is thus the block index at which the signal power spectrum drops (and remains) below $\lambda\%$ of the total energy of the block[1]. The block bandwidth estimate

[1] The block length, N, also represents the total energy of the block in Equation (4.9), due to the normalization performed on the time domain input sequence, $x''(n)$, using Equation (4.3); and due to the

is thus:

$$f_{\max} = \frac{j \cdot f_s}{2N} \qquad \text{Hz}; \tag{4.10}$$

where f_s is the frequency at which the input signal was sampled. A damped signal bandwidth estimate, f_d, is then updated once per block, using the block bandwidth estimates:

$$f_d = \begin{cases} \beta \cdot f_d + (1-\beta) \cdot f_{\max}, & \text{for } f_{\max} < f_d; \\ f_{\max}, & \text{for } f_{\max} \geq f_d; \\ f_d, & \text{for } f_{\max} > 0.95\frac{f_s}{2}; \end{cases} \tag{4.11}$$

where, β is a damping factor whose value was determined empirically as given in Section 5.2.

This provides immunity to extraneous high frequency noise bursts, and allows f_d to rise quickly to the true bandwidth of the input signal, while still permitting long term bandwidth tracking.

A bitrate goal, G_b, for each block is then determined directly from f_d, via the linear relation:

$$G_b = \nu \cdot f_d + \phi \qquad \text{kb/s}. \tag{4.12}$$

The total mainstream bit allocation, B, for the block, required to meet the G_b bitrate goal, is thus:

$$B = \frac{N \cdot G_b}{f_s} \qquad \text{bits}. \tag{4.13}$$

This bit allocation will be modified as described in Section 4.3.2 to achieve silence deletion, and then applied as described in Section 4.3.3, to quantize the transform coefficients.

fact that for the DCT (which is a unitary transform):

$$\sum_{n=0}^{N-1} x''^2(n) = \frac{1}{N} \sum_{n=0}^{N-1} T^2(n).$$

4.3.2 Silence Deletion

In an archiving application, silence deletion is an attractive way to reduce audio storage requirements. The effective coding of silent periods at the beginning and end of each selection, as well as internal silences such as those found in question/answer format interviews, can provide large reductions in average signal bitrate with no perceived decrease in quality. A graduated silence deletion is used in this work rather than a threshold cut-off approach. Rather than simply deleting all blocks with a total energy less than the threshold value, research on the characteristics of the human ear suggests a graduated approach, where coder distortion is allowed to increase as the signal level decreases. This distortion is less easily detected by the human ear at low levels, resulting in no perceived reduction in quality [42]. This gradual reduction in bitrate avoids sudden changes in signal quality as signal level drops, and allows bitrate savings to begin well before simple thresholding occurs.

Silence deletion can thus be achieved through a bitrate multiplier, ρ, of the form:

$$\rho = \frac{V}{\tau}; \tag{4.14}$$

where V is the mean energy of the block (Equation (4.3)), and τ is a threshold factor, set as described in Section 5.2.

The number of bits, B', which will be allocated to quantize the transform coefficients can thus be drawn from Equation (4.13) as:

$$B' = \rho B. \tag{4.15}$$

Similarly, the number of predictor coefficients which will be used to generate the smooth spectrum estimate can be drawn from Equation (4.8) as:

$$N_p' = \rho N_p. \tag{4.16}$$

Both B', and N_p' are rounded to the nearest integer. To avoid premature deletion of the spectral estimate, N_p' is not allowed to drop below 1, until $B' < 10$.

4.3.3 Bit Allocation

The number of bits, $b(i)$, to be used to quantize transform coefficient $T(i)$ is obtained from equations (3.18) and (4.15) as:

$$b(i) = \frac{B'}{N} + \frac{1}{2}\log_2\left[T^{2u}(i)\right] - \frac{1}{2N}\sum_{j=0}^{N-1}\log_2\left[T^{2u}(j)\right] \qquad \text{for } 0 \le i \le N-1. \quad (4.17)$$

Each allocation, $b(i)$, is then rounded to the nearest integer, $b_i(i)$, and the calculations of Equation (4.17) are performed again, this time considering only those coefficients which were assigned $b(i) > 0$ bits during the first iteration. This brings the realized block bit assignment much closer to the goal B'[1]. After the second iteration, a series of tests is performed on $b_i(i)$ to prevent negative, and to limit positive, bit assignments. The final allocations, $b_f(i)$, are given by:

$$b_f(i) = \lfloor b_i(i) \rfloor \qquad \text{for} \quad 0 \le i \le N-1; \qquad (4.18)$$

subject to,

$$b_f(i) = \begin{cases} 0, & \text{if } b_i(i) \le 0; \\ b_i(i), & \text{if } 1 \le b_i(i) \le b_{\max}; \\ b_{\max}, & \text{if } b_i(i) > b_{\max}; \end{cases}$$

where $b_{\max}$ is an upper limit on the number of bits which may be assigned to any one transform coefficient, and $\lfloor x \rfloor$ denotes the nearest integer to x.

4.3.4 Quantizer Step-Size Adaptation

The optimum step-size for each transform coefficient quantizer, is calculated once per block from Equation (3.19) as:

$$\Delta(k) = Q \cdot v(k) \cdot \delta\left[b_f(k)\right] \qquad 0 \le k \le N-1. \qquad (4.19)$$

The quantizer output level, $L(k)$, for transform coefficient $T(k)$, is calculated using a uniform quantizer with step-size $\Delta(k)$. The set of quantizer levels $L(k)$, is written to the storage medium, along with the side information.

[1] The second iteration provides the greatest benefit in the case of signals with large negative bit allocations (eg. narrowband over-sampled signals). These negative values are rounded up to zero, resulting in an over-assignment of bits to the other transform coefficients in the block. The second iteration eliminates these negative values, and thus reduces the degree to which bits are over-assigned.

4.4 Post Filtering

During testing, it was found that after coding narrowband signals which had been over-sampled, high frequency noise appeared *beyond* the bandwidth of the original signal, resulting in a noticeably distorted reconstructed signal.

Filtering the decoded signal at its original bandwidth before playback removed all traces of distortion, with no noticeable decrease in signal quality. It is presumed that this distortion is the result of windowing, and block end effects, which allow small amounts of energy to propagate to frequencies beyond the bandwidth of the original signal. Since there is no signal energy in this region to provide masking, even small amounts of noise will cause considerable perceived distortion.

This problem does not occur in traditional speech coding where speech signals, whether narrowband or wideband, are usually *critically sampled* at 8 kHz or 16kHz respectively.

To reduce the effects of this distortion, a digital lowpass filter was included in the decoding algorithm. The cut-off frequency of the filter was adaptively adjusted to match the signal bandwidth estimate f_d, of Equation (4.11). The cut-off frequency, f_c, was damped and set to float slightly above f_d by:

$$f_c = \begin{cases} \psi \cdot f_c + (1-\psi) \cdot f_d \cdot 1.1, & \text{for } f_d \le f_c; \\ f_d \cdot 1.1, & \text{for } f_d > f_c; \end{cases} \tag{4.20}$$

where f_c is updated once per block, and ψ is a damping factor whose value was determined experimentally as given in Section 5.2.

A finite impulse response (FIR) filter based on a Blackman window design, and with N_t taps, was used (see [43]). Such a filter produces an output sequence $y_f(n)$, from the un-filtered sequence $y(n)$, via:

$$y_f(n) = \sum_{i=0}^{N_t-1} h(i) \cdot y(n-i); \tag{4.21}$$

where the set $h(i)$ of filter coefficients is calculated once per block as:

$$h(i) = \left[0.42 - 0.5\cos(\frac{2\pi i}{N_t-1}) + 0.08\cos(\frac{4\pi i}{N_t-1})\right] \sin(\frac{\omega(i-\alpha)}{\pi(i-\alpha)}) \qquad \text{for} \quad 0 \le i \le N_t-1;$$

and,

$$\alpha = \frac{N_t - 1}{2};$$

$$\omega = (\frac{2f_c}{f_s})\pi.$$

This filtering operation was performed on each block before the output samples were written to the output buffer for playback.

4.5 Algorithm Bitstream Summary

All parameters and data values multiplexed and written by the variable rate archiving algorithm to the mass storage device are summarized in this section.

4.5.1 Main Bitstream

The main information bitstream is composed entirely of transform coefficient quantizer levels, $L(k)$, representing the quantized transform coefficients. N values of $L(k)$ are stored for each block. The number of bits used to store each value $L(k)$ is calculated from the side information using Equation 4.18, and will vary from block to block.

4.5.2 Side Bitstream

The side information bitstream carries 3 parameters:

1) The reflection coefficients, $k(i)$ (Equation (3.12)), which specify the smooth spectrum estimate. N_p values of $k(i)$ are stored.

2) The block scaling parameter, V, (Equation (4.3)), which holds the mean energy of the block. Quantization is realized via a b_e bit exponent, and a b_m mantissa, once per block.

3) The number of predictors used to form the smooth spectrum estimate, N_p. N_p is derived by the coder from the autocorrelation function of the input signal using Equation (4.8), and is stored once per block, as a b_p bit quantity.

Chapter 5 Algorithm Simulation Results

5.1 The Simulation Process

The variable rate archiving algorithm proposed in Section 4 was tested through a 32 bit floating point Fortran simulation. The Fortran simulation provides an exact sample of the quality that would be obtained by a dedicated digital signal processing implementation of the variable rate algorithm, with the following exceptions:

- the simulation does not run in real time;
- internal calculations in the simulation are performed using a somewhat greater accuracy than would likely be practical in a dedicated real-time implementation.

Each of the test signals described in Section 5.3, was low pass (anti-alias) filtered, and then analog to digital (A/D) converted at sampling frequencies of 8–20 kHz, using a 16 bit linear quantizer. The resulting digital files were then stored for use as input to the coder, on the disk drives of the mainframe computer (Digital Equipment Corp. VAX 8600) used to perform the simulation. The coder then processed each digital input file, and stored an output file on the disk. Data could then be drawn from the input and output files, D/A converted, low pass filtered, and played back to monitor the distortions caused by the coding and decoding process.

In this way, using the test files of Section 5.3, the coder parameters of Section 5.2 were determined, and the test results of Section 5.5 were produced. A panel of listeners, as outlined in Section 5.4 was used to rate the quality of the coder.

Two benchmark comparisons are provided in sections 5.6 and 5.7 between the variable rate archiving algorithm, and a log-PCM and CCITT standard sub-band coder, respectively.

5.2 Coder Parameters

Throughout sections 3 and 4, many parameters were presented during the development of the variable rate coding algorithm. The values determined for these parameters will be given in this section. These values were set based on:

1) information available in the speech coding literature;
2) the results of simulation test runs using the speech and music files of Section 5.3.

The parameter values chosen and the equations in which they appear are summarized in Table 5.1.

Parameters used in the quantization of the reflection coefficients are listed separately, in Table 5.2. In both cases, parameters were adjusted to achieve a coded signal that was not perceptibly different[1] from the original signal. In cases where settings directly affected coder bitrate (eg. $N, M, a_i, b_i, b_{\max}, \phi$, etc.), parameter values were adjusted just beyond the point where perceptible distortions disappeared.

Parameter values could not be adjusted independently. Thus, the figures presented in Tables 5.1 and 5.2 represent the compromise set reached for the simulation runs of this work, and not necessarily the only, or optimum, parameter configuration.

Block sizes of $N = 32$ to $2,048$ samples were used during testing. It was found that at a 16 kHz sampling rate, block sizes of greater than $N = 256$ samples, produced

[1] A panel of un-trained listeners was used, as described in Section 5.4, to define the point at which differences became perceptible.

Parameter		Value	Units	Reference
Symbol	Function			Equation
N	Transform Block Length	256	samples	—
f_s	Sampling Frequency	16.0	kHz	4.10
f_c	Anti-alias Cut-off Frequency	7.5	kHz	—
M	Adjacent Block Overlap	32	samples	3.5
α	Pre/De–emphasis factor	0.7	—	4.2
b_e	Scaling Factor Word length (exponent)	4	bits	4.3
b_m	Scaling Factor Word length (mantissa)	12	bits	4.3
$N_{\max}$	Limit on Predictor Coefficient Count	18	predictors	4.8
λ	Block Bandwidth Limit	5	percent	4.9
β	Bandwidth Damping Factor	0.99	—	4.11
ν	Linear Bitrate Multiplier	0.0072	kb	4.12
ϕ	Linear Bitrate Offset	15.42	kb/s	4.12
u	Spectral Weighting Factor	1.0	—	4.17
$b_{\max}$	Limit on Bit Allocation	10	bits	4.18
N_t	Number of Taps in Post Filter	128	taps	4.21
Q	Quantizer Loading Factor	1.0	—	4.19
ψ	Post Filter Damping Factor	0.999	—	4.20
b_p	Predictor Count Word Length	4	bits	—
τ	Silence Deletion Threshold Factor	1000	—	4.14

Table 5.1 Archiving Algorithm Parameters

Coefficient Index, i	a_i	b_i	Bit Allocation
1	-0.99	0.900	9
2	-0.70	0.999	9
3	-0.90	0.850	8
4	-0.80	0.970	8
5	-0.80	0.850	8
6	-0.80	0.850	7
7	-0.80	0.850	7
8	-0.80	0.850	7
9	-0.70	0.800	6
10	-0.70	0.700	6
11	-0.60	0.600	6
12	-0.60	0.600	5
13–18	-0.60	0.600	4

Note: See equation (3.20)

Table 5.2 Reflection Coefficient Quantization Parameters

unacceptable noise levels, especially with files containing sharp transients, such as the piano selection (File 8, Section 5.3). The block size was thus fixed at $N = 256$

samples, to achieve adequate time domain resolution, while maintaining the lowest possible side information bitrate.

It was found as expected, that the cosine rolloff window (See Figure 3.5) reduced spectral smearing. The window was of the most benefit in over-sampled signals (eg. files 2, 5, 7, and 12), reducing somewhat, the level of extraneous high frequency spillover energy, and consequently reducing bit allocations in the high frequency region. The effect was not great however, even with large overlaps ($M = 128$). Thus, to provide the lowest possible bitrate, while still gaining some windowing benefit, a compromise overlap of $M = 32$ samples was chosen.

The predictor coefficient count was limited to a maximum of 18 predictors. For over-sampled signals, and musical signals such as the piano with its numerous high energy formants, or the jazz selection with its complex wideband spectrum, 18 predictors were often required to provide an accurate spectral estimate. Setting $N_{\text{max}} > 18$ increased the coder bitrate with little audible increase in quality. Setting $N_{\text{max}} < 18$ audibly increased coder distortion, particularly on the piano, jazz, and over-sampled signals. Although this distortion increase could be balanced by finer quantization of the transform coefficients, the required increase in the main information bitstream was greater than the reduction realized in the side information bitstream. A compromise limit of $N_{\text{max}} = 18$ predictors, was thus chosen.

The linear bitrate factors, ν and ϕ, were determined empirically, based on the average bitrate required to code a given input signal to the point where it was indistinguishable from the original. This bitrate requirement was compared to the average bandwidth of the signal in question, and the above process repeated for each of the 12 test files of Section 5.3. A line of best fit was calculated (using linear regression) to relate average signal bandwidth, to minimum required bitrate. This line is specified by the values determined for ν and ϕ, through equation (4.12).

The damping factors β and ψ were adjusted through listening tests until changes in block bandwidth estimate, and filter cut-off frequency, were slowed to the point where they became inaudible.

Spectral weighting of the bit allocation was found to have very little audible effect, due to the already high quality of the coder. Thus, no spectral weighting was employed, and $u = 1$.

Any change in the quantizer loading factor Q, from the value $Q = 1$, was found to produce audible distortion in all test files. Thus Q was not changed from its optimum Max–quantizer value of $Q = 1$.

The predictor count word length was set to 4 bits. This allowed 16 of the 19 possible predictor assignments to be stored. Thus the number of predictors used to form the smooth spectrum estimate was restricted to members of the set $N_p = \{0, 1, 2, 3, 4, 5, 7, 9, 10, 11, 13, 14, 15, 16, 17, 18\}$. Optimal predictor counts produced by the FPE criterion were increased to the next set member if necessary, before storage.

5.3 The Simulation Test Files

Twelve audio files were used to test the performance of the variable rate archiving algorithm. Speech and music selections, of varying fidelity were chosen in order to accurately represent the contents of a typical audio archive.

A description of each test file is provided in this section.

File 1

Sampling Frequency: 8,000 Hz
Anti-alias filter cutoff frequency: 3,500 Hz
File length (samples): 16,018
File length (seconds): 2.0
Sample size: 16 bits (linear quantizer)
File mean: -1.829
File standard deviation: 1,951.8
Maximum sample value: 10,363
Minimum sample value: -8,281
Contents:

Narrowband female voice: “Happy hour is over”.

File 2

Sampling Frequency: 16,000 Hz
Anti-alias filter cutoff frequency: 3,500 Hz
File length (samples): 25,790

File length (seconds): 1.6
Sample size: 16 bits (linear quantizer)
File mean: -8.75
File standard deviation: 2,152.9
Maximum sample value: 8,149
Minimum sample value: -9,125
Contents:

Narrowband over-sampled female voice: “Happy hour is over”.

File 3

Sampling Frequency: 14,000 Hz
Anti-alias filter cutoff frequency: 7,000 Hz
File length (samples): 28,000
File length (seconds): 2.0
Sample size: 16 bits (linear quantizer)
File mean: -1.82
File standard deviation: 1,952.0
Maximum sample value: 10,253
Minimum sample value: -8,342
Contents:

Wideband female voice: “Happy hour is over”.

File 4

Sampling Frequency: 8,000 Hz
Anti-alias filter cutoff frequency: 3,500 Hz
File length (samples): 30,976
File length (seconds): 3.9
Sample size: 16 bits (linear quantizer)
File mean: -25.0
File standard deviation: 1,209.4
Maximum sample value: 7,748
Minimum sample value: -8,843
Contents:

Narrowband male voice: ”The birch canoe slid on the smooth planks”.

File 5

Sampling Frequency: 16,000 Hz
Anti-alias filter cutoff frequency: 3,500 Hz
File length (samples): 27,045
File length (seconds): 1.7
Sample size: 16 bits (linear quantizer)
File mean: -0.14
File Standard deviation: 1,462.3
Maximum sample value: 8,941
Minimum sample value: -12,899
Contents:

Narrowband over-sampled male voice: “Happy hour is over”.

File 6

Sampling Frequency: 14,000 Hz
Anti-alias filter cutoff frequency: 7,000 Hz
File length (samples): 28,000
File length (seconds): 2.0
Sample size: 16 bits (linear quantizer)
File mean: -1.6
File Standard deviation: 1,523.0
Maximum sample value: 16,108
Minimum sample value: -10,195
Contents:

Wideband male voice: “Happy hour is over”.

File 7

Sampling Frequency: 16,000 Hz
Anti-alias filter cutoff frequency: 7,000 Hz
File length (samples): 19,642
File length (seconds): 1.2
Sample size: 16 bits (linear quantizer)
File mean: -1.14
File Standard deviation: 4,296.6
Maximum sample value: 6,410
Minimum sample value: -6,416
Contents:

400 Hz. tone.

File 8

Sampling Frequency: 16,000 Hz
Anti-alias filter cutoff frequency: 7,000 Hz
File length (samples): 148,699
File length (seconds): 9.3
Sample size: 16 bits (linear quantizer)
File mean: 1.96
File Standard deviation: 2,829.3
Maximum sample value: 15,622
Minimum sample value: -12,207
Contents:

Solo pianist (wideband).

File 9

Sampling Frequency: 16,000 Hz
Anti-alias filter cutoff frequency: 7,000 Hz
File length (samples): 161,710
File length (seconds): 10.1
Sample size: 16 bits (linear quantizer)
File mean: 1.18
File Standard deviation: 2,854.8

Maximum sample value: 18,176
Minimum sample value: -16,976
Contents:

Pan pipes, bombo, and gourd (wideband)

File 10

Sampling Frequency: 16,000 Hz
Anti-alias filter cutoff frequency: 7,000 Hz
File length (samples): 125,744
File length (seconds): 7.86
Sample size: 16 bits (linear quantizer)
File mean: 4.9
File Standard deviation: 995.9
Maximum sample value:6,535
Minimum sample value: -3,883
Contents:

Small jazz group (wideband).

File 11

Sampling Frequency: 20,000 Hz
Anti-alias filter cutoff frequency: 7,500 Hz
File length (samples): 100,743
File length (seconds): 5.0
Sample size: 16 bits (linear quantizer)
File mean: 0.75
File Standard deviation: 723.61
Maximum sample value: 4,335
Minimum sample value: -4,268
Contents:

Small jazz group, with male and female vocalists (wideband).

File 12

Sampling Frequency: 20,000 Hz
Anti-alias filter cutoff frequency: 3,500 Hz
File length (samples): 102,597
File length (seconds): 5.13
Sample size: 16 bits (linear quantizer)
File mean: 0.96
File Standard deviation: 721.65
Maximum sample value: 4,339
Minimum sample value: -4,116
Contents:

Small jazz group with male and female vocalists (narrowband, and over-sampled).

5.4 Test Methodology

As outlined in Section 2.2, the variable rate archiving algorithm was designed to duplicate the quality of a modern $\frac{1}{4}$ inch analog reel-to-reel tape deck. This, in effect, required that there be no perceptible differences between the original and the coded versions of each input signal. In Section 5.2, the coder parameter values chosen to achieve this high quality level were listed. In Section 5.5, the results of simulation runs using these parameter values are given. In this section, the procedures used to confirm that the original and coded versions of the test signals were indeed imperceptibly different are given.

A panel of un-trained listeners was assembled, and presented with a series of audio trials. The twelve test files of Section 5.3 were randomly organized into a series of comparison tests. Listeners were presented with a sequence of two audio signals, and asked to rate the signals as being:

1) Identical;
2) Definitely Different; or
3) Possibly Different.

The two signals presented to the listener were drawn randomly from one of the following pairings:

- Original Signal versus Coded Signal;
- Original Signal versus Original Signal:
- Original Signal versus Purposefully Degraded Signal.

The listener was, of course, not aware of the true nature of the signals being rated. The addition of Original vs. Original, and Original vs. Degraded signals to the test permitted the detection of any listener biases which might exist.

Listeners were allowed to hear each comparison twice, before assigning a similarity rating. A total of 10 listeners were presented with 40 comparison trials each, for a total data base of 400 trials. The trials were spread out evenly amongst the 12 test files, in random fashion. All listening tests were performed using loudspeaker monitors, in a controlled acoustic environment. The actual questionnaire used during testing is included in Appendix C. The results of the tests are given in Section 5.5.1.

5.5 Simulation Results

In this section, the results of the archiving algorithm Fortran simulation will be given. In Section 5.5.1 it will be shown that with two exceptions, test files passed through the coding algorithm were indistinguishable from the corresponding original files in over 98% of test trials considered.

In Section 5.5.2, the average bitrates required to store the test files of Section 5.3 are given. It is shown that the variable rate archiving algorithm offers an average storage saving of 75% over the original linear-PCM files, and 63% over log-PCM coded files.

5.5.1 Comparison Test Results

Coder comparison test[1] results show that in 89.9% of all test trials, the signal processed by the archiving algorithm, was judged to be indistinguishable from the original signal. This is an average figure, calculated using all twelve test files[2]. Percentage details, for each individual test file, are given in Table 5.3.

The table shows that for 10 of the 12 test files, listeners rated the coded signals as identical to the original signal in more than 90% of all trials. With two of the 12 test files however, the archiving algorithm gave a poorer than average performance. For File 2 (Narrowband over-sampled female speech), listeners often reported a faint low frequency buzz, and indicated on average, that the archive-coder processed signal and the original signal were identical only 33.6% of the time. Similarly, in the case

[1] See Section 5.4 for Comparison Test details.

[2] See Section 5.3 for test file details.

File	Contents	% Rated Identical
1	Narrowband Female Speech	100
2	Narrowband Over-sampled Female Speech	33
3	Wideband Female Speech	91
4	Narrowband Male Speech	100
5	Narrowband Over-sampled Male Speech	100
6	Wideband Male Speech	100
7	Wideband Over-sampled 400 Hz Tone	100
8	Wideband Piano	100
9	Wideband Pan Pipe, Bombo, and Gourd	100
10	Wideband Jazz group	92
11	Wideband Jazz Group (Male & Female vocalists)	100
12	Narrowband Oversampled Jazz Group (Male & Female vocalists)	33

Table 5.3 Comparison Test Ratings

of File 12 (Narrowband over-sampled music), a faint shimmering distortion was detected at the start of the file, resulting in identical ratings in only 33.2% of all trials. The performance of the archiving algorithm was thus found to be somewhat signal dependent. Further testing showed that the perceptible distortions heard in files 2 and 12 were due to spectral estimation and silence deletion errors respectively. These problems will be discussed in the conclusions of Section 6.

Thus, excluding files 2 and 12, the archiving algorithm showed very good performance; achieving on average, an identical rating 98.1% of the time.

The tests performed to detect listener bias produced the expected results. When two original signals were compared, listeners rated them identical 96.8% of the time. Signals that were purposefully degraded were rated different than the original 92.3% of the time. Thus listeners showed themselves to be quite capable of distinguishing high quality from low quality signals, under the test conditions used.

It can therefore be concluded, that the archiving algorithm is, in general, providing the high quality signal representations required in an audio archiving application.

5.5.2 Storage Requirements and Average Bitrates

In Table 5.4, the average bitrates required to store each test signal are compared for equivalent quality linear-PCM, log-PCM, and archiving algorithm coded signals. The totals show that on average, the archiving algorithm provides storage savings of more than 75% over linear-PCM, and more than 63% over log-PCM coders.

File	Contents	Average Bitrate (kb/s)		
		Archiving Algorithm	Linear-PCM	Log-PCM
1	Narrowband Female Speech	44.4	128	88
2	Narrowband Over-sampled Female Speech	42.7	256	208
3	Wideband Female Speech	66.3	224	154
4	Narrowband Male Speech	41.7	128	80
5	Narrowband Over-sampled Male Speech	43.8	256	192
6	Wideband Male Speech	66.0	224	126
7	Wideband Over-sampled 400 Hz Tone	31.2	256	192
8	Wideband Piano	76.1	256	160
9	Wideband Pan Pipe, Bombo, and Gourd	77.9	256	112
10	Wideband Jazz group	81.3	256	144
11	Wideband Jazz Group (Male & Female vocalists)	96.4	320	160
12	Narrowband Oversampled Jazz Group (Male & Female vocalists)	41.6	320	220
Average		59.2	240	153

Table 5.4 Coder Bitrate Results

The average bitrate results also show that the archiving algorithm adjusted well to the needs of each input signal. Bitrates ranged from 31.2 kb/s for the simplest signal (the 400 Hz tone) to 96.3 kb/s for the most complex (wideband jazz group with multiple vocalists). In addition, the algorithm demonstrated its ability to efficiently code over-sampled signals. Whether sampled at 8 kHz, or at 16 kHz, the same 3500 Hz bandwidth female voice segment was stored at essentially the same bitrate (44.4 versus 42.7 kb/s for files 1 and 2 respectively). Both of these rates, as expected, were significantly lower than the measured storage rate for the wideband (7 kHz) version of the same female voice segment: 66.3 kb/s (file 3). Similar results were obtained for male speech (files 4, 5, and 6), and for music (files 11, and 12).

5.6 Comparison With Log-PCM Coder

For benchmarking purposes, a comparison was performed between the variable rate archiving algorithm, and a basic μ-law log-PCM coder. The standard μ-law curve was used [2]:

$$c(x) = x_{\max} \frac{\log_e(1 + \mu \cdot \mid x \mid / x_{\max})}{\log_e(1 + \mu)}. \tag{5.1}$$

For each file, the number of bits used to log-quantize each sample was adjusted until the log-PCM and variable rate archiving coders produced signals of equivalent quality[1].

File	Contents	bits
1	Narrowband Female Speech	11
2	Narrowband Over-sampled Female Speech	13
3	Wideband Female Speech	11
4	Narrowband Male Speech	10
5	Narrowband Over-sampled Male Speech	12
6	Wideband Male Speech	9
7	Wideband Over-sampled 400 Hz Tone	12
8	Wideband Piano	10
9	Wideband Pan Pipe, Bombo, and Gourd	7
10	Wideband Jazz group	9
11	Wideband Jazz Group (Male & Female vocalists)	8
12	Narrowband Oversampled Jazz Group (Male & Female vocalists)	11

Table 5.5 Log-PCM Comparison Results

Based on the current bit allocation, b_i, the values of μ, and $x_{\max}$ in Equation (5.1) were set as follows:

$$\mu = 2^{b_i} - 1;$$

$$x_{\max} = 2^{b_i} - 1.$$

The comparison results are summarized in Table 5.5. The number of bits, b_i, required to quantize each sample was thus found to range from 9 to 13, depending on the nature of the source signal. The variable rate archiving algorithm produced an average (over

[1] Equivalence was determined using listening tests similar to those described in Section 5.4.

the 12 test files of Section 5.3) storage saving of 63.3% over a log-PCM coder of equivalent quality.

5.7 Comparison With CCITT Wideband Coder

The Comité Consultatif Internationale de Télégraphique et Téléphonique (CCITT) is currently standardizing a wideband (7 kHz bandwidth) 64 kb/s sub-band coding algorithm [1]. Although designed for speech coding (and up to 16 kb/s of data transmission), with music processing as only a minor consideration, the CCITT algorithm provides roughly the same signal bandwidth as the variable rate archiving algorithm tested in Section 5.5. A comparison of the two algorithms thus provides a second useful benchmark.

File	Contents	% Rated Identical		Average Bitrate (kb/s)	
		Archiving Coder	CCITT Coder	Archiving Coder	CCITT Coder
1	Narrowband Female Speech	100	20	44.4	64
5	Narrowband Over-sampled Male Speech	100	58	43.8	64
6	Wideband Male Speech	100	17	66.0	64
7	Wideband Over-sampled 400 Hz Tone	100	8	31.2	64
8	Wideband Piano	100	17	76.1	64
10	Wideband Jazz group	92	18	81.3	64

Table 5.6 CCITT Comparison Results

Table 5.6 shows the similarity ratings (i.e. the degree to which the coded signal was rated indistinguishable from the original), and the bitrates required by the archiving coder, and the CCITT coder, to process 6 test files from Section 5.3. The table shows that in 3 out of 6 cases, the archiving algorithm was able to provide higher signal quality at a lower bitrate than the CCITT coder. In the other 3 cases, the archiving algorithm increased its bitrate by up to 27% over that of the CCITT coder; however, the similarity ratings show that this increase was clearly necessary to maintain high signal quality.

Chapter 6 Conclusions

In this work, the unique coding needs of an audio archive have been examined. A variable rate adaptive transform coding algorithm has been developed to meet these needs, and a Fortran simulation has been run on a series of 12 audio test files, to examine its performance.

The Fortran simulation shows that the proposed archiving algorithm provides a high quality processed signal that listeners find indistinguishable from the original signal in 89.8% of the test cases considered. The algorithm adjusts well to the bandwidth and complexity of the input test signals, varying its bitrate from 31.2 to 96.3 kb/s, as required, to maintain high quality. These bitrates are reasonable for a high quality coder given the current state of the art in coder research. Test results also show that the algorithm performs equally well on speech and music signals, providing average storage savings of 75% over a 16 bit linear-PCM coder, and 63% over an equivalent–quality log-PCM coder.

A comparison between the archiving algorithm and the proposed CCITT standard wideband coder [1], shows that the archiving algorithm consistently provides higher quality, at bitrates ranging from 31.2–81.3 kb/s, versus the CCITT coder's fixed bitrate of 64 kb/s.

Algorithm performance was found however, to be somewhat dependent on the nature of the input signal. In 2 out of 12 test files, listeners reported small audible distortions in approximately 66% of all trials.

The first distorted segment occurred in File 2 (narrowband over-sampled female speech). Testing showed that it was due to errors in the FPE signal order estimation procedure. Although the introduction of the FPE criterion during coder development led to reduced bitrates over the fixed predictor count approach used to that point, it is clear that the basic FPE technique chosen is not always able to produce adequate results. A more complex signal identification method appears to be required, to produce a consistent high quality estimate. Alternatively, a return to the fixed predictor count approach would provide the required quality, but at an average side information bitrate increase of approximately 15% for the test files of Section 5.3.

The second distorted segment was heard in File 12 (narrowband over-sampled music). Testing showed this was due to unwanted silence deletion during a low volume section at the start of the file. Improvement to the archiving algorithm is also required in this area, to provide a consistent high quality output. Changes to the silence deletion threshold factor τ, a logarithmic deletion curve, a threshold cut-off approach, or a block look-ahead capability, should be considered in future implementations.

Neglecting the above two files however, the archiving algorithm produced a signal rated indistinguishable from the original 98.1% of the time. The archiving algorithm, with the improvements suggested above, thus shows great potential for storage savings in an audio archiving application.

6.1 Considerations For Real-Time Implementation

A prime consideration in the development of the variable rate archiving algorithm, as outlined in the design criteria of Section 2.2, was that it be implementable in real-time on existing digital signal processing (DSP) devices.

It has been shown in the speech coding literature that the adaptive transform coder can be implemented in real-time [20]. In [20], the homomorphic spectrum estimate procedure (Section 3.3.4) was used, rather than the linear prediction approach, but both are of similar complexity. Powerful DSP chips exist today which can in many ways out-perform the array processor used in [20], (eg. the TMS 320C30 by Texas Instruments, which can perform a 256 point DFT in approximately 1.5 ms versus the array processor's 3.28 ms requirement).

It thus appears that real time implementation of the variable rate algorithm proposed in Section 4 is feasible. The sampling rate required for archiving will however be much higher than for the narrowband speech system of [20]. Fortunately, in archiving, coding and decoding operations are never performed simultaneously, as is required in a transmission application, such as [20]. This cuts real-time requirements in half.

Should implementation in real-time prove difficult at the sampling rates required, several options exist:

1) The Use of Multi-processors

- systems have, for example, been constructed using multiple TMS 32010 chips that offer increased real-time capacity [44].

2) The Use of a Log-Average Spectrum Estimate

- results published in the speech coding literature indicate that at the high bitrates required in archiving, the Log-Average estimate procedure of Section 3.3.4 may be able to equal the performance of the LPC estimate with significantly less numerical calculation.

3) The Use of a Fixed Predictor Count

- using a fixed number of predictors to form the smooth spectrum estimate for each block, will eliminate the FPE criterion calculations.

4) The Use of Efficient DCT Algorithms

- new methods for efficiently calculating the DCT have been proposed [45], which could lead to reductions in real-time requirements.

6.2 Other Considerations

Two other considerations are worthy of note at this point. During development of the archiving algorithm, it became apparent that although all design criteria of

Section 2.2 had been met, Design Criteria 4 (Uneven coder/decoder workload) was not satisfied to the extent originally envisioned. In fact, the proposed archiving algorithm has a basically even workload, with the decoder performing all functions of the encoder except the computation of the signal autocorrelation and reflection coefficients, and the computation via the FPE criterion, of the number of predictors to be used to form the smooth spectrum estimate. Future work on the proposed algorithm, should address this issue, perhaps through the use of vector quantization of the side information.

Secondly, a somewhat arbitrary decision was made in Section 3.2 to persue development of an adaptive transform coder, over a sub-band coder. Modification of a sub-band coding algorithm to provide a variable bitrate could lead to interesting results. Such an approach would eliminate the prediction error problems encountered in the ATC approach, and could provide better rejection of extraneous high frequency noise in over-sampled signals through the deletion of entire high frequency sub-bands.

References

1. Taka, M., P. Mermelstein, P. Combescure, and F. Westall, **Overview of the 64 kb/s (7 kHz) Audio Coding Standard**, *Proc. IEEE International Conf. on ASSP*, 1986, pp. 17.1.1–17.1.6.

2. Jayant, N. S., and P. Noll, **Digital Coding of Waveforms**, *Prentice-Hall, Inc.*, Englewood Cliffs, N.J., 1984, Chapter 5.

3. Flanagan, J. L., et al, **Speech Coding**, *IEEE Transactions on Communication*, Vol. Com-27, No. 4, April 1979, pp. 710–736.

4. O'Shaughnessy, D., **Speech Communication**, *Addison-Wesley*, Reading, Mass., USA., 1987, Chapter 7.

5. Jayant, N. S., and P. Noll, **Digital Coding of Waveforms**, *Prentice-Hall, Inc.*, Englewood Cliffs, N.J., 1984.

6. Tribolet, J. M., and R. E. Crochiere, **Frequency Domain Coding of Speech**, *IEEE Transactions on Acoustics, Speech, and Signal Processing*, Vol. ASSP-27, No. 5, October 1979, pp. 512–530.

7. Zelinski, R., and P. Noll, **Adaptive Transform Coding of Speech Signals**, *IEEE Transactions on Acoustics, Speech, and Signal Processing*, Vol. ASSP-25, No. 4, August 1977, pp. 299–309.

8. Cox, R. V., and R. E. Crochiere, **Multiple User Variable Rate Coding for TASI and Packet Transmission Systems**, *IEEE Transactions on Communications*, Vol. COM-28, No. 3, March 1980, pp. 334–344.

9. Dubnowski, J. J., and R. E. Crochiere, **Variable Rate Coding of Speech**, *The Bell System Technical Journal*, March 1979, pp. 577–600.

10. Derby, J. H., and C. R. Galand, **Multirate Sub-Band Coding Applied to Digital Speech Interpolation**, *Proceedings of the International Conference on Acoustics, Speech, and Signal Processing*, 1985, pp. 43.3.1–43.3.4.

11. Ramstad, T. A., **Considerations on Quantization and Dynamic Bit-Allocation in Sub-Band Coders**, *Proceedings of the International Conference on Acoustics, Speech, and Signal Processing*, 1986, pp. 16.7.1–16.7.4.

12. Mensa, G., R. Montagna, and F. Rusina, **A Speech Coder for ISDN Terminals**, *Proceedings of the IEEE Global Telecommunications Conference*, 1986, pp.2.4.1–2.4.5.

13. Honda, M., and F, Itakura, **Bit Allocation in Time and Frequency Domains for Predictive Coding of Speech**, *IEEE Transactions on Acoustics, Speech, and Signal Processing*, Vol. ASSP-32, No. 3, June 1984, pp. 465–473.

14. Lundheim, L. M., and T. A. Ramstad, **Variable Rate Coding for Speech Storage**, *Proceedings of the International Conference on Acoustics, Speech, and Signal Processing*, 1986, pp. 56.14.1–56.14.4.

15. Riggs, J. L., **Engineering Economics**, *McGraw-Hill* , New York, N.Y., USA., 1977, p. 164.

16. Jayant, N. S., **Coding Speech at Low Bitrates**, *IEEE Spectrum*, August, 1986, pp. 58–63.

17. Cahn, F., **Pitch Translation of Trumpet Tones**, *Proceedings of the International Conference on Acoustics, Speech, and Signal Processing,* 1983, pp. 1380–1383.

18. Campanella, S. J., and G. S. Robinson, **A Comparison of Orthogonal Transformations for Digital Speech Processing**, *IEEE Transactions on Communications Technology*, Vol. COM-19, December 1971, pp. 1045–1049.

19. Wintz, P. A., **Transform Picture Coding**, *Proc. IEEE,* Vol. 60, July 1972, pp. 809–820.

20. Cox, R. V., and R. E. Crochiere, **Real-Time Simulation of Adaptive Transform Coding**, *IEEE Transactions on Acoustics, Speech, and Signal Processing*, Vol. ASSP-29, No. 2, April 1981, pp. 147–154.

21. Zelinski,R., and P. Noll, **Approaches to Adaptive Transform Speech Coding at Low Bit Rates**, *IEEE Transactions on Acoustics, Speech, and Signal Processing*, Vol. ASSP-27, No. 1, February 1979, pp. 89–95.

22. Huang, J. J. Y., and P. M. Schultheiss, **Block Quantization of Correlated Gaussian Random Variables**, *IEEE Transactions on Communications Systems*, September 1963, pp. 289–296.

23. Ahmed, N., T. Natarajan, and K. Rao, **Discrete Cosine Transform**, *IEEE Trans. Comput.*, Vol. C-23, 1974, pp. 90–93.

24. Oppenheim, A. V., and R. W. Schafer, **Digital Signal Processing**, *Prentice-Hall*, Englewood Cliffs, N.J., USA., 1975, Section 3.6.

25. Oppenheim, A. V., and R. W. Schafer, **Digital Signal Processing**, *Prentice-Hall*, Englewood Cliffs, N.J., USA., 1975, pp. 241–242.

26. Sloan, D., **Adaptive Transform Coding of Speech**, Masters Thesis, McGill University, Montreal, Quebec, July 1979.

27. Oppenheim, A. V., ed., **Applications of Digital Signal Processing**, *Prentice-Hall*, Englewood Cliffs, N.J., USA., 1978, Section 3.6.

28. Oppenheim, A. V., and R. W. schafer, **Digital Signal Processing**, *Prentice-Hall*, Englewood Cliffs, N.J., USA., 1975, Chapter 2.

29. Jury, E. I., **Theory and Application of the Z-Transform Method**, *J. Wiley & Sons, Inc.*, New York, N.Y., USA., 1964.

30. Jayant, N. S., and P. Noll, **Digital Coding of Waveforms**, *Prentice-Hall*, Englewood Cliffs, N.J., USA., 1984, p. 268.

31. Nash, J. C., **Compact Numerical Methods for Computers**, *J. Wiley & Sons*, New York, N.Y., USA., 1979.

32. Jayant, N. S., and P. Noll, **Digital Coding of Waveforms**, *Prentice-Hall*, Englewood Cliffs, N.J., USA., 1984, p. 527.

33. Segall, A., **Bit Allocation and Encoding for Vector Sources**, *IEEE Transactions on Information Theory*, Vol. IT-22, No. 2, March 1976, pp. 162–169.

34. Max, J., **Quantizing for Minimum Distortion**, *IRE Transactions on Information Theory*, Vol. IT-6, 1960, pp. 7–12.

35. Gray, A. H., and J. D. Markel, **Quantization and Bit Allocation in Speech Processing**, *IEEE Transactions on Acoustics, Speech, and Signal Processing*, Vol. ASSP-24, No. 6, December 1976, pp. 459–473.

36. O'Shaughnessy, D., **Speech Communication**, *Addison-Wesley*, Reading, Mass., USA., 1987, Chapter 8.

37. Gray, A. H., R. M. Gray, and J. D. Markel, **Comparison of Optimal Quantizations of Speech Reflection Coefficients**, *IEEE Transactions on Acoustics, Speech, and Signal Processing*, Vol. ASSP-25, No. 1, February 1977, pp. 9–23.

38. Soong, F., and B. Juang, **Line Spectrum Pair (LSP) and Speech Data Compression**, *Proceedings of the International Conference on Acoustics, Speech, and Signal Processing*, 1984, pp. 1.10.1–1.10.4.

39. Kabal, P. and R. Rabipour, **Adaptive Transform Coding of Speech**,*INRS–Telecommunications Report No. 83-07*, INRS–Telecommunications, 3 Place du Commerce, Verdun, Québec, H3E 1H6, April 1983.

40. Gooijer, J. G., B. Abraham, A. Gould, and L. Robinson, **Methods for Determining the Order of an Autoregressive–Moving Average process: A Survey**, *International Statistical Review*, Vol. 53, No. 3, 1985, pp. 301–329.

41. Akaike, H., **Fitting Autoregressive Models For Prediction**, *Ann. Inst. Statist. Math.*, Vol. 21, 1969, pp. 243–247.

42. Hirsh, I. J., **The Measurement of Hearing**, *McGraw-Hill*, New York, N.Y., USA., 1952.

43. Oppenheim, A. V., and R. W. Schafer, **Digital Signal Processing**, *Prentice-Hall*, Englewood Cliffs, N.J., USA., 1975, Section 5.5.

44. Morley, R. E., A. M. Engebretson, and J. G. Trotta, **A Multiprocessor Digital Signal Processing System for Real-Time Audio Applications**, *IEEE Transactions on Acoustics, Speech, and Signal Processing*, Vol. ASSP-34, No. 2, April 1986, pp. 225–231.

45. Vetterli, M., and H. J. Nussbaumer, **Simple FFT and DCT Algorithms with Reduced Number of Operations**, *Signal Processing*, Vol. 6, 1984, pp. 267–278.

Appendix A.

Analog/Digital Archiving Equipment

The equipment presented in figures 2.2 and 2.3, and the costs presented in figures 2.4a and 2.4b, are based on the following:

Figures 2.2 and 2.4a

Revox B 77 $\frac{1}{4}$ inch Reel-to-Reel Tape Deck.

Cost: $3500.

Tape speed used for archiving: 19 cm/s.

Ampex 456 $\frac{1}{4}$ inch reel tape (1.5 mil).

Reel Size: 18 cm (7.5 inch).

Tape Length: 366m (1200 feet).

Cost: $5/reel.

A 10,000 hour archive collection consists of 20,000 tapes.

Labour costs (weighted): $30/hr.

Time required to copy one tape: 0.1 hr.

Figures 2.3 and 2.4b

Fujitsu NM2505A 13cm WORM Optical Disk Drive.

Capacity: 300 Mb/side (2 data sides).

Cost: $3500.

Best MKII IBM PC/XT Compatible Computer.

with: 20 M hard disk, and 640 k floppy disk drives.

Cost: $2595.

Loughborough TMS32020 DSP Board.

with 16 bit D/A and A/D converters.

IBM PC plug-in compatible.

Maximum sampling frequency: 50 kHz.

Cost: $2595, including development software.

13cm WORM Optical Disk (300 Mb/side).

Cost: $75.

A 10,000 hour archive collection consists of 3,552 disks, @ 59.2 kb/s.

The cost of audio amplifiers (common to both the analog and digital systems) was not included. The cost of fixed anti-aliasing filters for the digital system was neglected. Labour costs in year 1, are assumed identical for the analog and digital systems, and are not included in Figure 2.4.

Year 1 costs for the analog system (Figure 2.4a) include: analog tape deck, and magnetic tape. In years 5 and 10, costs include: magnetic replacement tape, and labour costs. For the digital system (Figure 2.4b), year 1 costs include: optical disk drive, IBM PC/XT compatible computer, and optical disks.

Appendix B.

DSP Card Suppliers

The following suppliers manufacture IBM PC compatible plug-in digital signal processing products, based on the Texas Instruments TMS320 series of digital signal processors:

1) **Atlanta Signal Processors, Inc.**

770 Spring Street
Atlanta, Georgia, USA. 30308.
(404) 892-7265.

2) **Loughborough Sound Images Ltd.**

The Technology Centre
Epinal Way, Loughborough, Leicestershire
UK. LE11 0QE.
(509) 321843.

3) **Microcraft Corporation**

P.O. Box 513
Thiensville, Wisconsin
USA. 53092.
(414) 241-8144.

4) **OROS SA**

Chemin des Prés
ZIRST – 38240 Meylan
France.
(33) 76 90 62 36.

5) **Sky Computers, Inc.**

Foot of John Street
Lowell, MA
USA. 01852.
(617) 454-6200.

Appendix C.

Comparison Test Questionnaire

Coder Comparison Test

Instructions

Thank you for offering/agreeing to take the coder comparison test. You are about to hear 40 pairs of audio files. Each pair will consist of an original version of the input signal followed by a coded version of the same signal. Your task is to indicate on the following pages whether or not the two versions sound identical, definitely different, or possibly different.

For each signal, the original-version vs. coded-version comparison will be repeated **twice.** After hearing the comparison for the second time, please circle the appropriate similarity description to the right of the signal name.

- Circle **Identical** if you hear no difference between the original and the coded versions of the signal.

- Circle **Definite Difference** if you are absolutely certain you hear a difference (even a small one) between the two versions.

- Circle **Possibly Different** if you are almost certain there is a difference, but can't be 100% sure.

The test will start with two example signals: Tone 1, and Tone 2. The examiner will stop the tape after these example signals to see if you have any questions. Remember to listen to each original-signal vs. coded-signal comparison **twice**, before circling a responce. You will have 4 seconds to draw your circle before the next signal on the list is played. The test will take 18.3 minutes.

Please listen carefully.

Thank you.

Example Signals:			
Tone 1	Identical	Possible Difference	Definite Difference
Tone 2	Identical	Possible Difference	Definite Difference
Jazz	Identical	Possible Difference	Definite Difference
Piano	Identical	Possible Difference	Definite Difference
Male Voice (Low fidelity)	Identical	Possible Difference	Definite Difference
Male Voice (High fidelity)	Identical	Possible Difference	Definite Difference
Pipes	Identical	Possible Difference	Definite Difference
Tone	Identical	Possible Difference	Definite Difference
Male Voice (Low Fidelity)	Identical	Possible Difference	Definite Difference
Male Voice (High fidelity)	Identical	Possible Difference	Definite Difference
Female Voice (High fidelity)	Identical	Possible Difference	Definite Difference
Piano	Identical	Possible Difference	Definite Difference
Jazz	Identical	Possible Difference	Definite Difference
Male Voice (Low fidelity)	Identical	Possible Difference	Definite Difference
Female Voice (Low fidelity)	Identical	Possible Difference	Definite Difference
Female Voice (High fidelity)	Identical	Possible Difference	Definite Difference
Female Voice (Low fidelity)	Identical	Possible Difference	Definite Difference
Jazz	Identical	Possible Difference	Definite Difference

Male Voice (Low fidelity)	Identical	Possible Difference	Definite Difference
Music (Low fidelity)	Identical	Possible Difference	Definite Difference
Music (High fidelity)	Identical	Possible Difference	Definite Difference
Pipes	Identical	Possible Difference	Definite Difference
Tone	Identical	Possible Difference	Definite Difference
Female Voice (Low fidelity)	Identical	Possible Difference	Definite Difference
Male Voice (Low fidelity)	Identical	Possible Difference	Definite Difference
Female Voice (Low fidelity)	Identical	Possible Difference	Definite Difference
Piano	Identical	Possible Difference	Definite Difference
Female Voice (Low fidelity)	Identical	Possible Difference	Definite Difference
Male Voice (Low fidelity)	Identical	Possible Difference	Definite Difference
Male Voice (High fidelity)	Identical	Possible Difference	Definite Difference
Music (High fidelity)	Identical	Possible Difference	Definite Difference
Music (Low fidelity)	Identical	Possible Difference	Definite Difference
Jazz	Identical	Possible Difference	Definite Difference
Tone	Identical	Possible Difference	Definite Difference
Male Voice (Low fidelity)	Identical	Possible Difference	Definite Difference
Female Voice (Low fidelity)	Identical	Possible Difference	Definite Difference

Male Voice (High fidelity)	Identical	Possible Difference	Definite Difference
Female Voice (High fidelity)	Identical	Possible Difference	Definite Difference
Piano	Identical	Possible Difference	Definite Difference
Music (High fidelity)	Identical	Possible Difference	Definite Difference
Tone	Identical	Possible Difference	Definite Difference
Jazz	Identical	Possible Difference	Definite Difference