COMPRESSION OF HIGH QUALITY AUDIO SIGNALS

1. Description of the problem

Fairlight Instruments, who brought the problem to the MISG, have developed a high quality "Computer Musical Instrument" (CMI) which is used for creating and manipulating music and sounds (mostly of musical instruments) for use in music recording, editing, mixing and composition for the music, film and advertising industries. These industries require sound reproduction of at least the quality of the sound stored on compact discs. [On a compact disc, sounds are sampled at 44.1kHz and digitised to 16 to 20 bits.] The CMI stores most of its input and output on multi-track magnetic tape, but there would be great advantages in speed and flexibility if the sounds could be stored on random access read/write discs. The high sampling rates required for the sound quality make this prohibitively expensive unless a method of substantially compressing the data can be developed. The compression of high-quality audio signals also has other applications; significant effort is going into the development of methods for transmitting high fidelity music along ISDN (telephone) channels.

General methods of data compression are widely available but the compression of music has two special features. Firstly, the compression and decompression must occur in real time. To produce the quality of sound from compact discs 44,100 samples of 16 - 20 bits must be compressed per second. [Decompression will generally be a faster operation than compression.] This implies that the compression will be done using a program on a digital signal processing (DSP) or customised VLSI chip and it imposes restrictions on the complexity of the compression algorithm used. Secondly, in many circumstances it will be acceptable for the result of compressing and then decompressing a signal to be just perceptually identical to the original. Perceptually identical signals may be substantially different.

Shortly before the study group, papers were presented at a conference in the USA (85th Convention of the Audio Engineering Society) which described in general terms an algorithm which compressed music from a Compact Disc in real-time, from 16 bits per sample to 1.45 bits per sample (Brandenburg et al, 1988). The main components of this algorithm were

(1) A transformation of the data to the frequency domain by the discrete cosine transform.

(2) Modification of the coefficients by a "psycho-acoustic weighting function"

(3) Quantisation of the coefficients

(4) Compression using an entropy (Huffman) coder.

The Huffman coder and compression algorithms like it are reversible - the original signal can be recovered identically from the compressed signal. Because of the quantisation of the coefficients of the discrete cosine transform, the algorithm of Brandenburg is irreversible - the decompressed signal may be significantly degraded even though perceptually identical to the original signal. There is a third possibility - a near-reversible algorithm - which introduces a very slight degradation but from which the original signal may be very nearly recovered.

Fairlight expressed an interest in all types of compression algorithm: reversible, near reversible and irreversible. It seemed , however, that matters such as quantisation and psycho-acoustic weighting would be determined by experiment and were beyond the scope of the one week study group and would be determined by experimental methods. We therefore decided to look at reversible and near-reversible algorithms.

2. Compression using autoregressions

A general method for compressing data is to use a parametric model to predict observations. The parameters are estimated from the data and, instead of saving the raw data, the estimates of the parameters together with the residuals from the model are saved. If the model is a good one, the residuals have significantly smaller variance than the raw data and so can be stored (or transmitted) using fewer bits.

The principal method of compression considered was the modelling of the data series using an autoregression, a method also known as linear predictive coding (LPC). In this method, the sequence of observations, $x(n)$, is supposed to come from a model

$$x(n) + b_1 x(n-1) + ... + b_p x(n-p) = e(n)$$

and the $e(n)$'s are assumed to be white noise (i.e. independently and identically distributed random variables with mean zero and variance constant).

For a basic implementation, an autoregression is fitted to a block of data and, as described above, the estimates of the b_j's, the 'start up' values $x(1),...,x(p)$ and the residuals $e(n)$ are stored. For more sophisticated implementations, the final values from one block can be used as the starting values for the next. The use of the autoregressive model is attractive because there are fast algorithms

for estimating the parameters of the autoregression. Such algorithms operate in real time when carefully programmed on a Digital Signal Processing (DSP) chip for reasonably useful values of n and p. These algorithms are recursive in the order of the model: given the parameter estimates for an autoregression of order p, it is easy to compute quickly the parameter estimates for an autoregression of order $p+1$.

The issues that remain to be considered are:

(i) the size of the block of data to be used.

(ii) the order of the autoregression to be fitted.

The two issues are related; as the size of the block of data becomes bigger, so it becomes more likely that the parameters of the model best describing the data will vary over the block - even the generating process itself will change.

One approach is to decide upon a fixed block length and then for each block to fit various models (autoregressions of different orders) and to choose the best one. What is 'best' and the method of determining which model is best will depend on the final aim. For data compression, Rissanen (1978) showed that if a sequence of autoregressions of increasing order is fitted then the order yielding the encoded data of minimum length is that which minimizes

$$BIC(p) = ln\ s_p^2 + p\, lnT/T$$

where T is the length of the data and s_p^2 is the residual mean square after fitting the autoregression of order p. To test the applicability of these ideas, a small number of experiments were conducted on digitised piano music provided by Bruce Tulloch of Fairlight Instruments.

3. Data analysis

A set of 8000 data points from some piano music were analysed by fitting autoregressions . Figure 1 shows a plot of 1000 observations (0.02 seconds of music) and their periodogram. To investigate the importance of block size, autoregressions were fitted with the data divided into 16 blocks of length 500, and 8 blocks of 1000 and finally 1 block of 8000. In each case the order of the autoregression was determined using the BIC approach, with the maximum allowable order being $2\sqrt{T}$. In Table 1, the orders of the selected models and the corresponding residual variances are given for each block and each block size.

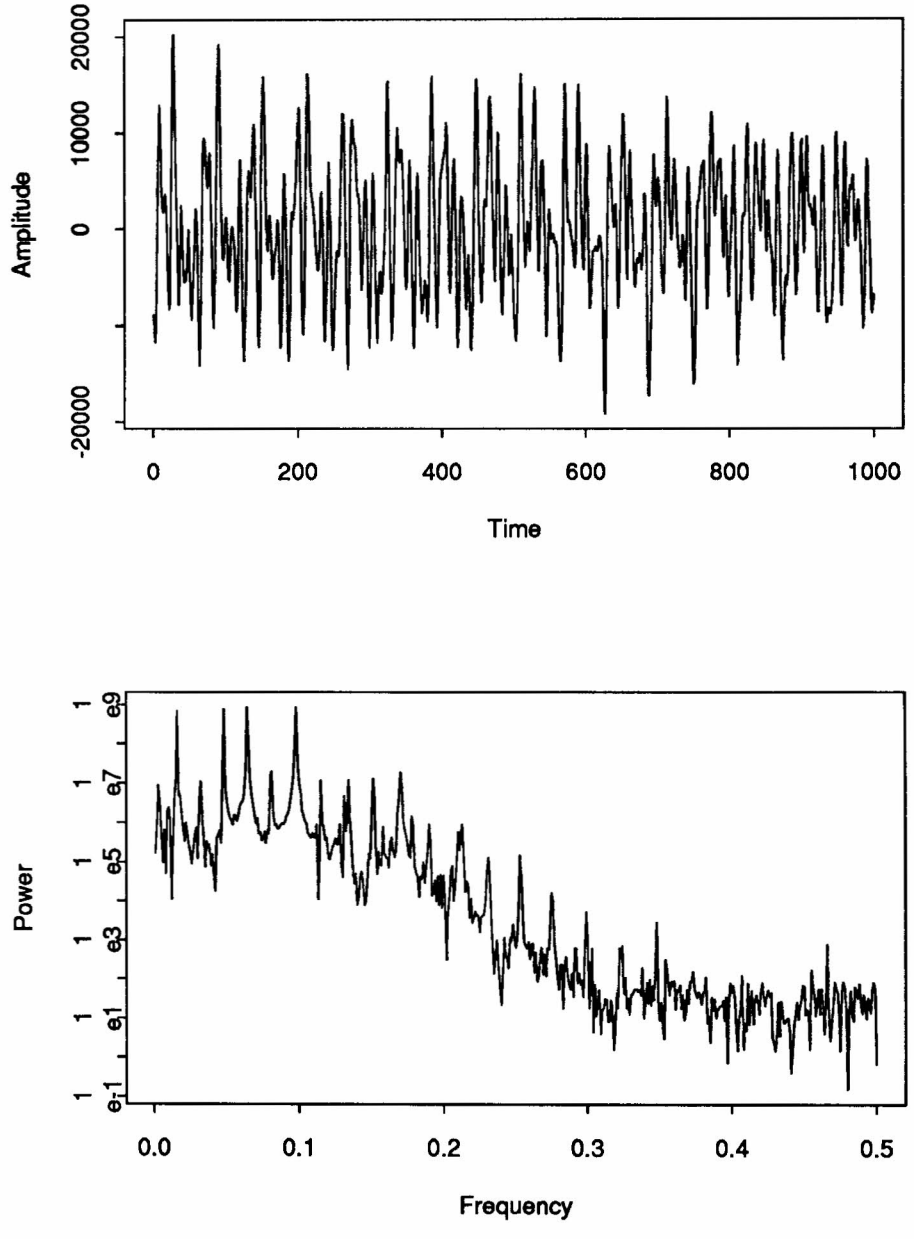

Figure 1: (a) Plot of 1000 observations (0.02 secs) of piano music. (b) The periodogram (squared amplitude of Fourier coefficients of the data plotted in (a)).

Table 1. Results of autoregression fitting

Block size = 500			Block size = 1000			Block size = 8000		
Block var	AR order	NSR	Block var	AR order	NSR	Block var	AR order	NSR
53.3	34	0.016	47.6	58	0.008	19.8	110	0.00175
42.0	33	0.019						
28.1	32	0.025	25.0	59	0.012			
21.9	7	0.036						
20.0	8	0.038	19.1	59	0.008			
18.3	6	0.035						
15.0	22	0.024	15.7	58	0.010			
16.3	15	0.031						
17.1	20	0.020	16.6	58	0.008			
16.2	21	0.012						
14.3	22	0.012	14.0	57	0.007			
13.8	23	0.016						
12.5	15	0.020	11.6	60	0.006			
10.7	19	0.009						
10.2	16	0.002	9.3	58	0.003			
8.4	19	0.008						

*NSR = Ratio of residual variance to signal (block) variance.

From this table, we can see, for example, that to fit a model to each of the first two blocks of 500 observations a total of 67 coefficients would be required, whereas the entire block of 1000 requires a model of only 58 parameters and this model has smaller residual variance than either of the models fitted to the smaller blocks. In fact the autoregression fitted to all 8000 observations requires only four parameters more than are required to fit autoregressions to each of the first four blocks of 500. The residual variance after fitting the model to all 8000 observations corresponds to a compression of about 4.5 bits per observation. There remains substantial scope for quantisation of the data (roughly removing the least significant bits from each observation) and thus compressing the data further. A further possibility would be to develop a 'vocabulary' of autoregressions. Then that member of the vocabulary which best fitted the data would be used to compute residuals. The benefit of this would be that the parameters of the autoregressions would not need to be stored each time, but rather would be stored in a dictionary. As a result, there need not be the same penalty for storing coefficients and so smaller blocks of data could be used for the same size model.

4. Compression of Fourier coefficients

Further data compression is possible. Firstly, the residuals from the selected autoregression are not uncorrelated because the process generating the data is varying with time, and so a second autoregression could be fitted to the residuals from the first. A second approach is to transform to the frequency domain and perform data compression there. This may be particularly useful after fitting the autoregression, for the effect of that fitting is to make the amplitudes of the Fourier coefficients more nearly constant (that is to flatten or whiten the spectrum). A method is to divide the frequency range into bands, to compute the mean amplitude within the band and then instead of storing the amplitudes, to store the mean amplitude and the results of dividing each of the amplitudes by the mean amplitude. For the present data, the effect of this approach was to reduce the range of the amplitudes by a factor of 100.

Clearly the effectiveness of this compression of Fourier coefficients depends on a good choice of bands. Cameron (1987) and Hannan and Rissanen (1988) provide methods for choosing the bands.

After these compressions in the time and frequency domains a Huffman coder would then be used to provide a further compression of the signal. Thus a complete compression algorithm might consist of:

1. Transform to frequency domain
2. Use psychoacoustic weighting
3. Fit autoregressions and select that providing greatest compression
4. Use spectrum smoothing technique to reduce dynamic range of Fourier coefficients
5. Use entropy coder.

The effectiveness of each of these steps and the interactions between them must be determined in part by experimentation. This was beyond the scope of the Study Group, but the results of the preliminary work done here suggest that the autoregression fitting would provide significant improvement in the compression of audio signals.

5. References

Brandenburg, K. and Seitzer, D. (1988) OCF: Coding high quality audio with data rates of 64 kbit/sec. *Proceedings of the 85th Convention of the Audio Engineeering Society.*

Cameron, M.A. (1987) An automatic, non-parametric spectrum estimator. *Journal of Time Series Analysis*, **8**, 379-387.

Hannan, E.J. and Rissanen, J. (1988) The width of a spectral window, *J. Appl. Prob.*, **25A**, 301-307.

Rissanen J. (1978) Modelling by shortest data description. *Automatica,* **14**, 465-471.