A Real-Time ADPCM Encoder Using Variable Order Prediction

Frederick L. Kitson
Hewlett Packard Laboratories
1501 Page Mill Road, Bldg. 29
Palo Alto, California 94304

Kenneth A. Zeger *
Mass. Institute of Technology
Department of EECS
Cambridge, MA. 02139

ABSTRACT

The compression of digitally encoded speech in real-time is of interest to those working on voice transmission and storage systems. This paper evaluates the application of a variable order predictor to Adaptive Differential Pulse Code Modulation (ADPCM) to lower the bit rate without mitigating the speech quality. The technique encodes the speech waveform by using a lattice prediction filter to predict the input speech waveform and then adaptively quantizing the prediction error. The prediction error is monitored and the minimum order predictor filter is determined for a prescribed error for each analysis speech segment. The order selection algorithm is derived and a program flow chart given for the fixed point implementation. By using the minimum order filter, one reduces the number of coefficients that have to be coded. A comparison is then done to contrast the new variable order algorithm with the more conventional fixed order LPC algorithm. Results include the favorable SNR performance of the coder for various bit rates in the mid-band (10-32 Kb/s) range as well as the coding aspects on the Digital Signal Processing Chip (TMS320). Several programming innovations such as adaptive bit allocation for correlation coefficient normalization are also described. Work is now in progress to extend this research to lower bit rates and includes the application of noiseless compression techniques on the encoded residual.

I. INTRODUCTION

Various forms of Adaptive Predictive Coding [1] have been used to compress speech in which an estimate is made of the input speech scalar or vector and the difference residual is coded. The prediction can be made on the basis of spectral envelope or fine spectral structure. Additionally, the prediction can be of the forward or backward variety and can be based on block techniques or recursive algorithms [2]. Here we consider the block diagram of Figure 1 which incorporates the lattice predictor of Figure 2 which uses the so called partial correlations (PARCOR) or reflection coefficients [3]. The desirable property of having the i^{th} order optimum prediction available at the i^{th} stage of the P^{th} order filter is necessary to this work. In this block diagram, side information about the predictor must be transmitted or stored with the residual. The aim of this paper is to find an algorithm that will provide a mechanism for deciding when high order predictors are indeed reducing the prediction error and identifying those situations when a low order predictor will suffice. For fixed order, either fixed coefficients or adaptive coefficients the performance as a function of order is known [4]. This paper will provide results for when the order is adaptive in an ADPCM coder so that the average order necessary will be reduced and consequently the side information without sacrificing the subjective or quantitative speech quality.

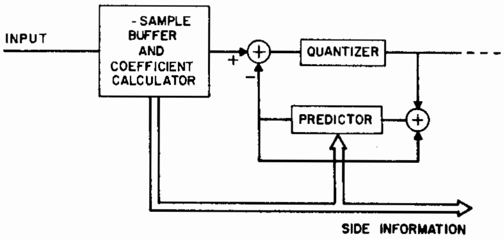

Fig. 1 ADPCM block diagram with Forward Adaptive Prediction (APF) [4].

II. PREDICTION FILTER

This work considers the use of a linear transversal predictor based on the spectral envelope of the speech waveform with the general form:

$$p[n] = \sum_{i=1}^{P} a_i s[n-i] \qquad (1)$$

The prediction residual is the difference between the input speech scalar and the estimate p[n] which is the indicated inner product of a prediction coefficient vector and P previous speech samples. The residual waveform is then

$$e[n] = s[n] - p[n] \qquad (2)$$

This error signal can be minimized with respect to the a_i's in a least mean square sense by minimizing

$$E = \sum_{n=0}^{N-1} e^2[n] = \sum_{n=0}^{N-1} \left[s[n] - \sum_{i=0}^{P} a_i s[n-i] \right]^2 \qquad (3)$$

for i = 1,2,...,P. Setting $\partial E / \partial a_i = 0$ for i = 1,...,P yields P well known equations of the following form

$$R(j) = \sum_{i=1}^{P} a_i R(|i-j|) \qquad j=1,\dots,P \qquad (4)$$

$$where \quad R(j) \equiv \sum_{n=0}^{N-1} s[n] s[n-j]$$

are the autocorrelation coefficients of the input speech signal $s[n]$. It also was decided to use a lattice type implementation of equation (1) for its property of having a coefficient vector of k_i's bounded individually in magnitude and also possessing good quantization pro-

* Presently at Dept. of ECE, UC Santa Barbara, CA, 93106.

perties. This factor makes this form more desirable for fixed point implementations. The prediction $p[n]$ is derived from a sum of certain states within the filter.

III. VARIABLE ORDER PREDICTION

For the block coding method here, it is necessary to determine the set of coefficients a_i's (or k_i's) which minimizes in some sense the prediction error over the frame interval of N speech samples. The appropriate coefficient vector is computed and the filter updated between the last sample of one block and the first sample of the new block. The predictor then, as a function of time, can be represented as a sequence of PARCOR vectors :

$$\left\{k_1^1, k_2^1, k_3^1, \ldots, k_P^1\right\}, \left\{k_1^2, \ldots, k_P^2\right\}, \ldots \left\{k_1^j, \ldots, k_P^j\right\} \cdots$$

where j indicates the frame number. If the sampling period is T, the frame size is N, and on the average b bits are used to quantize each k_j, then Pb / NT bits/sec must be transmitted to the receiver to describe the predictor. For $N=128, T=125*10^{-6} sec, p=8, \text{and} b=8$, this gives 4K bits/sec to be transmitted. In the range of $0-32K\ bits / sec$ systems, this is substantial. In a variable order prediction system, a predictor of the form of equation (1) with $P = q(j)$. is used where $q(j)$ is the order of the predictor used for the j^{th} frame. We require $1 \le q(j) \le P_m$, where P_m is the maximum predictor order allowed due to implementation constraints. Using such a predictor, only $a_1^j, a_2^j, \ldots, a_{q(j)}^j$, or equivalently $k_1, \ldots, k_{q(j)}^j$ need be transmitted for the j^{th} frame to predict the N speech samples of that block.

The goal in varying the predictor order is to form a prediction waveform which generates a residual with lower or equal power than a fixed prediction filter. Let P be the order of a fixed-order predictor, with $1 \le P \le P_M$. It is desired that $time\ avg\ (q(j)) \equiv \bar{q}(j) < P$ with

$$\sum_{n=jn}^{(j+1)N-1} \left\{ s[n] - \sum_{i=1}^{q(j)} a_i^{q(j)} s[n-i] \right\}^2 \equiv E(\bar{q}(j)) \le E(P) \quad (5)$$

for all j≥0. This makes variable order prediction as accurate, in the least squares sense, as fixed-order prediction, but in addition has a reduced bit rate. The reduction in the average bit rate needed for transmission is $(P-q(j))*b*(NT)^{-1}$.

Several algorithms for determining $q(j)$, the order of the j^{th} frame's predictor, were derived and tested. The most promising will be shown in detail keeping in mind that computation efficiency and storage requirements are important parameters of a real-time implementation.

Now considering Durbin's recursion below which will generate the a_i's and k_i's that minimize $E^{(P)}$ one can obtain the equation

$$E^{(0)} = R(0) \quad (6)$$

$$E^{(i)} = (1-k_i^2)E^{(i-1)} \quad (7)$$

$$k_i = \frac{R(i) - \sum_{j=1}^{i-1} a_j^{(i-1)} R(i-j)}{E^{(i-1)}} \qquad 1 \le i \le P \quad (8)$$

$$a_i^{(i)} = k_i \quad (9)$$

$$s_j^{(i)} = a_j^{(i-1)} - k_i a_{i-j}^{(i-1)} \qquad 1 \le j \le i-1 \quad (10)$$

$$E^{(i)} = R(0) \prod_{j=1}^{i} (1-k_j^2) \qquad -1 \le k_i \le 1 \quad (11)$$

which gives the prediction error for the i^{th} order predictor. Consequently the error is bounded by $R(0)$. The signal to noise ratio (predictor error) of the i^{th} order predictor is given in dB by

$$SNR(i) = 10 \log_{10} \left[\frac{E^{(0)}}{E^{(i)}} \right] = -c \sum_{j=1}^{i} \ln(1-k_j^2) \quad (12)$$

where c = 10 / ln 10. Equation (12) can be modified using

$$-ln(1-x) = \sum_{n=1}^{\infty} \frac{x^n}{n} \qquad |x| < 1. \quad (13)$$

Since we are guaranteed that the reflections coefficients are bounded so that $|k_j^2| < 1$ for all j, using P to represent predictor order we get,

$$SNR(P) = c \sum_{j=1}^{P} \sum_{n=1}^{\infty} \frac{k_j^{2n}}{n} = c \sum_{n=1}^{\infty} \frac{1}{n} \sum_{j=1}^{P} k_j^{2n} . \quad (14)$$

A Taylor series approximation will be used for (14) where L will be a limit for the summation on n. This eliminates a look-up table and allows one to specify the SNR in dB so that the error increment per stage is additive instead of multiplicative as in equation (11). If a P^{th} order linear predictor is 'upgraded' to a $(P+1)^{th}$ order predictor, the increase in the signal-to-noise-ratio associated with the predictor is $-10\log_{10}(1-k_P^2)$ from equation ().

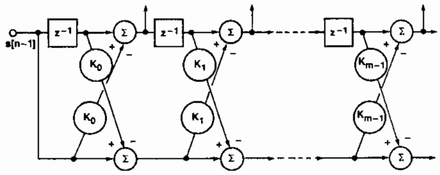

Fig. 2 Lattice form linear predictor.

IV. IMPLEMENTATION

This particular implementation was constrainted to support a single half duplex channel so that real time transmission or storage could be accomplished on a continuous basis. The input speech was first stored in a circular buffer of length 2N to allow speech statistics to be collected and for buffer "elasticity". The hardware essentially consists of one TMS320 digital signal processing chip and a "combo" μ-law A/D-D/A chip which includes the switched capacitor anti-aliasing and reconstruction filters. The speech waveform is bandlimited to be between 200 and 3200 hz and subsequently sampled at 8 KHz using the 8-bit μ-law 255 converters. The μ-law was chosen because of its popularity in modern telephony so that the processor could interface directly to a μ-law switching stream.

The processor is interrupted every speech sample and first converts the sample to a linear form before performing a high frequency pre-emphasis filtering with a transfer function of $H(z) = 1. - .8z^{-1}$. Such a filtering process enhances the subsequent modeling process [4]. For each frame j of speech consisting of 128 samples, P lattice prediction filter coefficients are derived using a modified version of the fixed point LeRoux and Guegen algorithm [5]. The input to the algorithm is an estimate of the P normalized autocorrelation coefficients for the particular frame. The advantage of this procedure over others it that all the internal variables are bounded in magnitude and so avoid some of the potential for overflow in a fixed point computation scheme. The problem however is the normalization of the $R^{(j)}(i)$'s. The digital signal processor (TMS320) has a sixteen bit architecture and so has a 16 by 16 bit hardware multiplier, 16 bit high speed registers and data paths, and special logic for 16 bit divisions. The autocorrelation function,

$$R^{(j)}(i) = \sum_{n=N(j-1)}^{jN-1-i} s^{(j)}[n]*s^{(j)}[n+1] \tag{15}$$

for i = 0,1,...,P_m where j is the block number is computed with a double precision accumulate. The speech waveform can have a dynamic range of over 60 db in this system and subsequently the energy computation of the $R^{(j)}(i)$'s requires that the *16 most significant bits* of the result will begin in any one of at least 20 positions in the 32 bit words being accumulated. To optimize the dynamic range so that neither soft nor loud utterances are mapped into a sub-optimum prediction vector, an adaptive scaling algorithm was employed for this computation. Since $R^{(j)}(0)$ is the energy in the j^{th} frame (or block) of speech, and $R^{(j)}(0) \geq R^{(j)}(i)$ for *all* $i > 0$, $R^{(j-1)}(0)$ can be used to indicate the maximum value of $R^{(j)}(i)$. The value b defined as $b = \log_2[R^{(j-1)}(0)/R^{(j)}(0)]$ is small in magnitude. The quantity represents the number of extra or the number of fewer leading zero bits that $R^{(j)}(0)$ will have compared to those of $R^{(j-1)}(0)$ in the accumulator. Typically, $b \approx 0,1,2,3$.

A. Adaptive Scaling

The strategy employed is to premultiply each speech sample $s[n]$ by $2^{m(j)}$, where $m(j)$ is an integer associated with the j^{th} frame such that

$$m(j) = \frac{1}{2}(31-[\log_2 R^{(j-1)}(0)]-\varepsilon) \tag{16}$$

where [] indicates the greatest integer function and where ε is a prechosen integer to provide flexibility for occasional rapidly increasing waveform power. In this implementation ε was chosen to be 4. Denoting the scaled version of the speech signal and its corresponding autocorrelation function by using a subscript M, one has:

$$s_M^{(j)} = s^{(j)}[n]*2^{m(j)} \tag{17}$$

$$R_M^{(j)}(i) = \sum_n s_M^{(j)}[n]*s_M^{(j)}[n+i] = s^{2m(j)}R^{(j)}(i) \tag{18}$$

$$\leq 2^{31-\varepsilon}[R^{(j-1)}(0)]^{-1}R^{(j)}(0) \tag{19}$$

To avoid accumulator overflow we must satisfy $R_M^{(j)}(i) < 2^{31}$ or equivalently

$$2^{-\varepsilon}[R^{(j-1)}(0)]^{-1}R^{(j)}(0) < 1 \tag{20}$$

As indicated with the choice of epsilon, equation [20] will be satisfied in a statistically significant manner. The end result is that $R^{(j)}(i)$ will 'fill' the accumulator by minimizing leading zeroes so that the 16 highest order bits of the result can be stored with the dynamic range necessary to characterize accurately the autocorrelation function. The next step is the normalization of $R_M^{(j)}(i)$'s by an accurate 16 bit division by $R_M^{(j)}(0)$ which is equal to $R^{(j)}(i)/R^{(j)}(0)$.

B. Quantizer

A robust adaptive quantizer was employed [6] in which the error residual $e[n]$ is first scaled to a fixed step size quantizer and then at the decoder (and predictor) is multiplied by the inverse scaler $\Delta[n]$. The step size is updated as shown below,

$$\Delta(n+1) = (\Delta(n))^{\gamma *}\Psi(C(n)) \,. \tag{21}$$

where γ, the leakage constant, is typically .98 and the $\Psi()$ is a non-linear expansion/contraction function of the present quantizer output codeword C(n). The algorithm of Boddie et al [7] uses the log of $\Delta(n)$ and $\exp(-\Delta(n))$ so that

$$d(n) = \gamma * d(n) + \Gamma(C(n)). \tag{22}$$

The 64 values of $log(\Delta(n))$ and its reciprocal are calculated based on the necessary dynamic range and then stored in ROM for easy lookup. The TMS320 is ideally suited to the adaptive step-size control since the processor can do the one multiply and add of equation (22) and a "table read" command which puts the accumulator on the address bus and loads from the next step size in the table.

V. RESULTS

The prediction gain as a function of predictor order increases asymptotically as $|k_i|$ decreases asymptotically to zero. One can use many criterion of the lattice to determine this point of diminishing return. Here, three possible functions of the signal-to-noise ratio in dB are examined as to their appropriateness for calculating $q(j)$ or the number of coefficients ie $k_1^{(j)} \cdots k_{q(j)}^{(j)}$ that are sufficient for the prediction of the j^{th} frame. The number $q(j)$ will be encoded with 3 bits as part of the side information where $1 \leq q(j) \leq 8$. The three candidate criterion for optimal predictor order selection are:

$$1. \quad \min i : \qquad SNR(i) > thresh_1 \tag{23}$$

$$2. \quad \min i : \ SNR(i+1) - SNR(i) < thresh_2 \tag{24}$$

$$3. \quad \min i : \ SNR(i+2) - SNR(i) < thresh_3 \tag{25}$$

The three methods were chosen to emphasize a determination by the absolute SNR, the first order difference in the SNR, and the second order difference in SNR respectively. Methods 2 and 3 are looking at a measure of the prediction trend and the relative strength of adjacent prediction gain while method 1 is essentially making a decision about voiced or unvoiced speech. Method 1 will be presented in some detail as the other two follow in a similar fashion. Performance results are given for all three for comparison.

Method 1 is then an implementation of the following rule:

$$q = \begin{cases} 1 & SNR(1) < thresh_{1.5} \\ \min i \ : \ SNR(i) > thresh_1 & 1 \leq i \leq P_M \\ P_M & otherwise \end{cases} \tag{26}$$

Since $SNR(1) = 1/1-k_1^2$, one need only check $|k_1|$ against a prestored constant which from equation (8) is equivalent to comparing the ratio of $R(1)$ to $R(0)$. This is a first order correlation test to see is the signal is more like a "white spectrum signal" or a first order autoregressive process. This can also be considered a unvoiced or voiced speech waveform test in which case more PARCOR coefficients will not be able to properly predict the speech waveform. In fact, more coefficients can reduce performance due to quantization errors. Using equation (26) the threshold test will be:

$$q = \min i \ : \sum_{j=1}^{i}\left[\sum_{n=1}^{L}\frac{k_j^{2n}}{n}\right] > \frac{1}{c} * thresh_1 \tag{27}$$

The flowchart for implementation on the DSP processor is given in Figure 3(a) where $L = 4$ is a sufficient number of terms in the Taylor series.

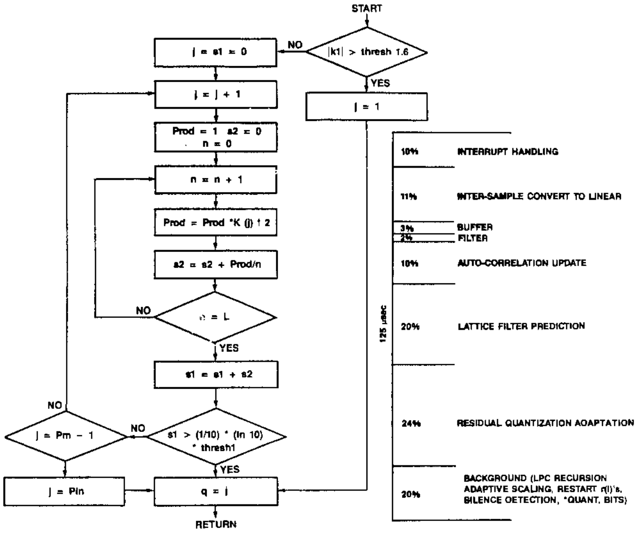
START
NO
j = s1 = 0
\|k1\| > thresh 1.6
YES
j = j + 1
i = 1
Prod = 1 s2 = 0 n = 0
n = n + 1
Prod = Prod *K (j) ↑ 2
s2 = s2 + Prod/n
NO
n = L
YES
s1 = s1 + s2
NO
NO
j = Pm − 1
s1 > (1/10) * (ln 10) * thresh1
YES
j = Pin
q = j
RETURN
10% INTERRUPT HANDLING
11% INTER-SAMPLE CONVERT TO LINEAR
3% BUFFER
2% FILTER
10% AUTO-CORRELATION UPDATE
125 μsec
20% LATTICE FILTER PREDICTION
24% RESIDUAL QUANTIZATION ADAPTATION
20% BACKGROUND (LPC RECURSION ADAPTIVE SCALING, RESTART r(i)'s, SILENCE DETECTION, *QUANT. BITS)

Fig. 3 Variable order prediction ("Method 1") (a) Flowchart (b) DSP Processor time allocation.

The algorithms developed were tested in real time by apply phonetically balanced prerecorded sentences through the system many times. A monitoring system was used to collect the encoded residual and resynthesized speech for analysis from the TMS320 processor which could be programmed for various coding options. The utilization of the processor running at ~200 nsec/instruction for the Variable Order Prediction is shown in Figure 3(b). ADPCM encoding with fixed prediction was carried out for $P = 1,2,...,8$. An example of the function $q(j)$ and the corresponding speech waveform is shown is Figure (4).

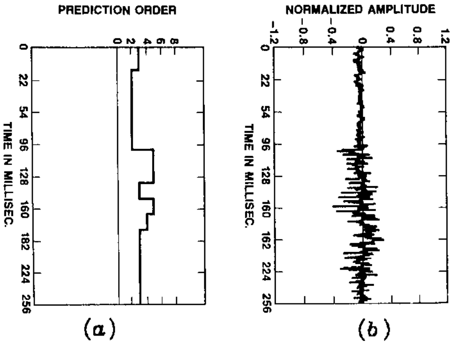

Fig. 4 Optimum order selection vs time. (a) Optimum order. (b) Speech waveform.

The RMS prediction error was recorded for each fixed predictor and is graphed vs predictor order in Figure (5). For variable order prediction cases, each "method" (1,2,3) was tested with the appropriate threshold constants varied each time a "speech sentence" was passed through the coding hardware. In this way the time average predictor order was varied so that $\overline{q}(j)$) ranged from 1 to $P_m = 8$. Again prediction error vs prediction order is plotted in Figure (5). The three sentences used for comparison for all cases were: *1. "She wants to speak about the ant". 2. "My T.V. has a twelve inch screen". 3. "All the flowers are in bloom"*. The results indicate (see ref. [8]) that significant prediction gains in speech can be achieved at low average predictor orders. The largest gains are between average predictor orders of 1 to about 3. Method 1 appears to offer the best performance at these average predictor orders for all cases tried which was substantiated during subjective listening tests. Assuming PARCOR coefficients are represented by 8-bit numbers , a savings of 500 bits/sec is achieved for each reduction in average order by 1. For example, for an RMS prediction error of .45 in Sentence 1, Method 1 provides a savings of about 5 orders of linear prediction (from 6.5 to 1.5). This amounts to a 2.5 Kbit/sec reduction in overall bit rate. This is to be compared to residual bit rates of 6 to 16 Kb/sec in the system being described here.

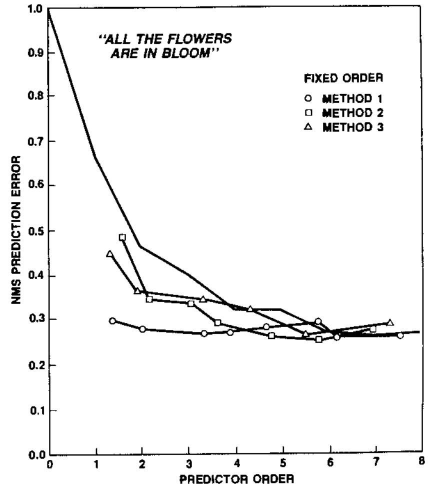

Fig. 5 Normalized prediction error vs prediction order.

REFERENCES

[1] Atal, S. A., "Predictive Coding of Speech at Low Bit Rates", IEEE Trans. on Comm., Vol. COM-30, No. 4, pp. 600-614, April 1982.

[2] Reininger, R. C., Gibson, J. D., "Backward Adaptive Lattice and Transversal Predictors in ADPCM ", IEEE Trans. on Comm., Vol. COM-33, No. 1, pp. 74-82, Jan. 1985.

[3] Makhoul, J., "A Class of All-Zero Lattice Digital Filters: Properties and Applications", IEEE Trans. on ASSP, Vol. ASSP-26, No. 4, pp. 304-314, Aug. 1978.

[4] Jayant, N.S., Noll, P., *Digital Coding of Waveforms*, Prentice Hall, 1984.

[5] Le Roux, L., Gueguen, C., "A Fixed Point Computation of Partial Coefficients", IEEE Trans. on ASSP, pp. 257-259, June 1977.

[6] Goodman, D. J., Wilkinson, R. M., "A Robust Adaptive Quantizer", IEEE Trans. Comm., Vol. COM-23, pp. 1362-1365, Nov 1975.

[7] Boddie, J. R. et al, "Adaptive Differential PCM Coding", BSTJ, Vol. 60, No. 7, Part 2, pp. 1547-1561, Sept. 1981.

[8] Zeger, K. A., "A Real-Time Variable Prediction Order, ADPCM Speech Encoding System", Masters Thesis, MIT, May 1984.