Proc. IEEE Int. Conf. Acoustics, Speech, Signal Processing (New York, NY), pp. 243-246, April 1988

A LOW DELAY 16 KBITS/SEC SPEECH CODER

V. Iyengar[1] and P. Kabal[1,2]

[1]Electrical Engineering
McGill University
Montreal, Quebec H3A 2A7

[2]INRS-Télécommunications
Université du Québec
Verdun, Quebec H3E 1H6

Abstract

This paper studies a speech coder using a tree code generated by a stochastic innovations tree and backward adaptive synthesis filters. The synthesis configuration uses a cascade of two all-pole filters — a pitch (long time delay) filter followed by a formant (short time delay) filter. Both filters are updated using backward adaptation. The formant predictor is updated using an adaptive lattice algorithm. The multipath (M, L) search algorithm is used to encode the speech. A frequency weighted error measure is used to reduce the perceptual loudness of the quantization noise. The speech coder has low delay (1 ms) and has been evaluated through formal subjective testing to have speech quality that is equivalent to that for 7-bit log-PCM.

1. Introduction

This paper addresses the problem of low delay coding of speech signals at an encoding bit rate of 16 kbits/sec. Such a coder has application in the switched telephone network. Encoding delay is an important consideration in the design of speech coding algorithms for use as part of a terrestrial common carrier network, due to the problem of disturbing echoes generated at the hybrid interface between two-wire and four-wire lines. Also, it is important that the coder achieve toll quality for wide spread acceptability as part of the switched telephone network.

Traditionally, toll quality coders have been waveform coders. Among waveform coders are the commonly used log-PCM coders, and the more recent ADPCM coding schemes. ADPCM achieves a lower transmission rate at the expense of increased complexity. ADPCM schemes employ an adaptive predictor and an adaptive quantizer matched to the short term statistics of the input speech. The CCITT has formally approved an ADPCM coding algorithm that provides toll quality speech at 32 kbits/sec. The use of backward adaptation schemes for the predictor and quantizer allows for very low encoding delay.

If the encoding rate of the CCITT algorithm is reduced much below 32 kbits/sec, the quality produced drops off significantly. This occurs because of the interaction between the predictor update algorithm and the quantizer. When the quantizer noise increases, the performance of the predictor (which is determined by the reconstructed samples) drops, further increasing the quantization noise effects. Most previous attempts at high quality coding at rates near 16 kbits/sec have used forward adaptation for the predictor. However, this strategy tends to introduce large amounts of coding delay.

A major advance towards improving the performance of waveform coders at low bit rates comes with the use of multipath tree search algorithms with differential waveform coders. A characteristic of differential coders is that the possible quantized output sequences are arranged in the form of a *tree code*. The quantizer and predictor pair essentially plays the role of a tree code generator. Encoding in conventional schemes proceeds by a *single path* search of this code tree to find the best output sequence. This has been identified as being a clear shortcoming of conventional DPCM and ADPCM. Much can be gained with the use of delayed decision schemes which employ multipath searches to make more efficient use of the tree code. The delays are usually of the order of a few samples, and can be kept within network echo delay constraints.

Tree coding involves two basic issues. The first is the choice of an effective code tree, and the second involves the choice of a search algorithm to search the code tree for the output sequence that best matches the input. Multipath tree searching of codes generated by a fixed deterministic quantizer and a fixed predictor was first studied by Anderson and Bodie [1]. Jayant and Christensen [2] have studied the use of multipath searching of a code generated by a backward adaptive quantizer and a fixed predictor. Although delayed decision coding in the above studies provides gains in terms of both perceived speech quality and measurable signal-to-noise ratio over conventional differential encoders at a rate of 16 kbits/sec, the output speech quality was still reported to be characterized by easily perceived quantization noise. This is due to two main shortcomings. First, the encoding algorithm uses fixed and not adaptive prediction. Second, the tree code is a deterministic tree code. Much can be gained with the use of so-called stochastic tree codes as will be shown later. Multipath tree searching with forward adaptive prediction has been studied in [3], [4] and [5]. Unfortunately, forward adaptation of the predictor usually entails a large amount of encoding delay, of the order of 10–20 ms. Recently Gibson and Haschke [6] have studied deterministic code trees with backward adaptive formant synthesis filters, but without a pitch loop.

The scope of this paper is as follows. First, a generalized predictive coder is briefly described. This configuration allows the use of a frequency weighted error measure. Subsequent sections then view this coding scheme in the more general tree coding context. The predictive coding scheme is then extended to the multipath search case with a stochastically populated innovations tree. Multipath searching is done with the (M, L) algorithm [7]. Both objective and subjective test results of the coding algorithm are presented. This paper presents the final coder configuration used for performance evaluation and can only hint at the experimentation that was used to arrive at this configuration.

2. Generalized Predictive Coder

The block diagram of a generalized predictive coder with a short-term or formant predictor is shown in Fig. 1 [8]. This configuration allows for adaptive adjustment of the noise spectrum in relation to the speech spectrum. The formant predictor $F(z)$ acts on short-term redundancies in the input speech signal, while the quantization noise is shaped by $N(z)$,

$$F(z) = \sum_{i=1}^{P} a_i z^{-i}, \qquad N(z) = \sum_{i=1}^{P} b_i z^{-i} . \tag{1}$$

Fig. 1 also includes a pitch predictor. The use of pitch prediction is motivated by the fact that voiced speech segments exhibit considerable similarity between adjacent pitch periods. In this work, a 3-tap pitch predictor is used, having the system function of

$$P(z) = \beta_1 z^{-M_p+1} + \beta_2 z^{-M_p} + \beta_3 z^{-M_p-1} , \tag{2}$$

where M_p is the pitch period in samples.

For the configuration shown in the Fig. 1, the spectrum of the reconstruction error is given by

$$S(z) - \hat{S}(z) = \frac{Q(z)}{1-P(z)} \frac{1-N(z)}{1-F(z)} . \tag{3}$$

where the quantization error is given by $Q(z)$. With the usual assumptions of uncorrelated quantization noise, the quantization error has a flat power spectral density. The shape of the reconstruction error spectrum can be controlled by choosing $N(z)$ appropriately. It is usual to choose $N(z)$ as a bandwidth expanded version of $F(z)$, i.e., $N(z) = F(z/\mu)$ where $0 < \mu \leq 1$. The value of μ is usually chosen to be between 0.75 and 0.9. Such a value of μ has the effect of decreasing the noise power in the valleys (regions between the formants) of the speech spectral envelope and increasing the noise power in the formant regions (where it is more tolerable).

2.1 Formant Filter Adaptation

The update algorithm used for the formant predictor in this work is the adaptive lattice algorithm [9]. The prediction error filter $1 - F(z)$ can be formulated as a lattice filter as shown in Fig. 2. The reflection coefficients K_m are allowed to vary with time to track the modes of stationarity of the input $s(n)$. The reflection coefficients K_m are therefore a function of time n, and shown explicitly by writing $K_m(n)$, $m = 1, 2, \ldots, P$. The update method is based on the minimization of a weighted error of the form

$$E_m(n) = \sum_{k=-\infty}^{n} w(n-k)e_m^2(k) \; . \tag{4}$$

where $e_m^2(k)$ is a sum of forward and backward residual energies,

$$e_m^2(k) = f_m^2(k) + b_m^2(k) \; . \tag{5}$$

and $w(n)$ is a causal window function. Minimizing $E_m(n)$ with respect to $K_m(n)$ yields the update $K_m(n+1)$,

$$K_m(n+1) = \frac{2\sum_{k=-\infty}^{n} w(n-k)f_{m-1}(k)b_{m-1}(k-1)}{\sum_{k=-\infty}^{n} w(n-k)\left[f_{m-1}^2(k) + b_{m-1}^2(k-1)\right]} = \frac{C_m(n)}{D_m(n)} \tag{6}$$

A simple one-pole or exponential window given by

$$W(z) = \frac{1}{1-\beta z^{-1}} \; , \tag{7}$$

is used in the final configuration. The update equations for $C_m(n)$ and $D_m(n)$ are then given by

$$\begin{aligned} C_m(n) &= \beta C_m(n-1) + f_{m-1}(n)b_{m-1}(n-1) \\ D_m(n) &= \beta D_m(n-1) + f_{m-1}^2(n) + b_{m-1}^2(n-1) \end{aligned} \; . \tag{8}$$

The value of β is chosen to be 0.986. Adaptation of the formant filter is done using the reconstructed speech samples.

The resulting reflection coefficients are converted to direct form coefficients for use in the filtering operations.

2.2 Pitch Filter Adaptation

Adaptation of the pitch predictor requires both the pitch lag and pitch coefficients to be updated. Conventionally, pitch prediction uses forward adaptation. In this work, we deviate from this practice and employ a backward adaptive pitch filter. Minimizing the mean square error over a frame of length N samples results in the following system of linear equations,

$$\sum_{n=1}^{N} r(n)r(n-M_p+2-i) = \sum_{j=1}^{3} \beta_j \sum_{n=1}^{N} r(n-M_p+2-i)r(n-M_p+2-j) \; , \tag{9}$$

for $i = 1, 2$, and 3. This in turn can be written compactly in matrix form as $\boldsymbol{\Phi\beta} = \boldsymbol{\alpha}$, where $\boldsymbol{\Phi}$ is the 3 by 3 correlation matrix. In a backward adaptation scheme, the sequence $r(n)$ would correspond to the past quantized formant residual signal.

In operation, first an estimate of the pitch period M_p is obtained using the method described in [10], then Eq. (9) is used to obtain the set of predictor coefficients β_i.

If the pitch predictor is backward adapted, the frame over which the mean square prediction error is minimized does not correspond to the frame over which the pitch predictor is applied. While this scheme works well in segments where the lag is relatively constant, it does not perform as well in transition regions. This is because the pitch lag and coefficients are too finely tuned to the analysis frame. Some of the adverse effects of backward pitch predictor adaptation can be tempered by '*softening*' the pitch predictor. Conceptually this is achieved by adding uncorrelated white noise to the input to the pitch predictor and using this perturbed signal to solve for the pitch lag and pitch predictor coefficients. This approach is implemented by adding a noise term to the diagonal elements of the correlation matrix $\boldsymbol{\Phi}$ — the diagonal elements $\phi(i,i)$ of the matrix $\boldsymbol{\Phi}$ are replaced by $(1+\alpha)\phi(i,i)$ where α is a small factor.

3. Tree Codes

Conventional differential encoders such as the generalized predictive coder described above, can be viewed in a more general setting as so-called tree coding algorithms. These tree codes can be classified into two categories, ***deterministic*** and ***stochastic*** tree codes. The tree codes associated with conventional waveform coders such as PCM and ADPCM are examples of deterministic tree codes. With differential encoders, it is useful to make a distinction between the ***innovations code tree*** which represents the quantized residual signal and the ***reconstruction code tree*** which represents the reconstruction output signal. The reconstruction code tree is obtained by passing each of the quantized residual sequences of the innovations code tree through the synthesis filter. The nodes of the reconstruction code tree are then populated with the output values of the filter. With a backward adaptive synthesis filter, the reconstruction code tree is completely specified by the filter (update algorithm and initial conditions) and the innovations code tree. The encoder configuration in Fig. 1 reflects the use of a frequency weighted error measure in searching through the code tree.

In a deterministic tree, each quantized residual sample can take on one of 2^R values, where R is the number of bits per sample used to encode the input. Stochastic trees are richer in that they allow more generality in populating the innovations tree. The approach taken here to obtain good codes for speech signals is to capture the long term statistics of formant and pitch predicted residual signals in a stochastically populated innovations tree. The essential pitch and formant structure is then inserted by passing the innovations sequence through backward adaptive pitch and formant synthesis filters.

Each node in the stochastic innovations tree is associated with a unique path map or branch number sequence from the root up to that particular node. The N most recent bits of the path map are used as an index into a dictionary containing 2^N values. The node takes on the value from the dictionary associated with that index, after multiplication by a gain factor. For proper decoding of the quantized residual sequence at the receiver, identical copies of the dictionary must be available to both the transmitter and the receiver.

The gain adaptation is achieved as follows. The dictionary value assigned to a node is multiplied by the node gain G to yield an innovations sample e. The node gain is then updated according to

$$\hat{G}^2 = \delta G^2 + (1-\delta)e^2 \; , \qquad 0 < \delta < 1, \tag{10}$$

where $\hat{G}$ is the new gain value, and δ is a parameter that controls the effective memory of the gain update.

3.1 Multi-path Search Algorithm

The (M,L) algorithm was used to search the code tree. The (M, L) algorithm is controlled by two parameters, M and L. The maximum number of paths kept in contention at any stage is at most M. The length of these paths is equal to L. Each of the saved paths is first extended to the nodes corresponding to the next sampling instant. The cumulative errors for each of the paths are then calculated, and the extended path with the lowest error is identified. This lowest error path will extend from a node L time samples back. The branch number for this node is then transmitted. This corresponds to an incremental mode of operation where each search involving sequences of length $L+1$ is followed by the release of one branch number. Only valid paths that extend from the chosen node L time samples back are in contention, the others are eliminated. Among the valid paths, at most M lowest error paths are kept and saved for the next stage.

4. Encoding Algorithm

The dictionary for the innovations tree was populated with 4096 samples from a Laplacian pseudo-random number generator, and a value of 0.86 was used for the parameter δ, used in the gain adaptation. We will first consider the case where only a formant synthesis filter is used. The filter $F(z)$ along each path is updated at each time instant using the backward adaptive algorithm described. If there is no delay in the update, the formant filter $F(z)$ will evolve differently along different paths of the tree. Proceeding along different parallel paths of the code tree will therefore involve a separate update of the formant filter along each path. By exploiting the fact that all saved paths stem from a single released node L time samples back, the computational complexity of the encoding scheme can be reduced. If the formant filters are updated with a delay of L samples, the filters for each of the saved paths evolve in an identical way. All the saved paths are therefore associated with a single filter which evolves via a delay of L samples. Furthermore, for the exponential adaptive lattice window, experiments show that the prediction gain actually increases with increasing update delays (up to delays of 8 samples) and then levels off.

With the inclusion of a pitch synthesis filter, the encoder and decoder configurations of Fig. 1 are used along each path of the tree. The pitch predictor is updated using the past released quantized residual samples, i.e., using the output of the pitch synthesis filter. The pitch filter is updated every 20 samples. The pitch lags are constrained to lie between 20 and 120 samples, and a frame length of 100 samples is used in solving for the pitch predictor coefficients.

5. Objective Test Results

The encoding algorithm was simulated on a VAX 8600 computer using FORTRAN. All arithmetic operations were carried out using floating point arithmetic. The value of L is fixed at 8.

The objective results using only a formant synthesis filter are presented first. An eighth order formant filter was used. The plot in Fig. 3 shows the segSNR values versus M for a particular sentence, with a stochastically populated innovations tree. The segSNR is the average of the decibel values of the SNR calculated for 16 ms blocks. The segSNR value increases rapidly with M at first, and then finally saturates with M to an almost constant value. Other sentences show a very similar behaviour.

Figure 3 also shows the performance with M with a deterministic innovations tree. The underlying 4 level quantizer was made adaptive using a Jayant step size update [11] (multiplier values of 0.9 and 1.6). The results show that with the deterministic tree, the (M, L) algorithm achieves a performance similar to that for a full search (Viterbi algorithm) [6]. Comparing the deterministic code with the stochastic code brings out some interesting points. First, saturation with M occurs at a much lower value of M with a deterministic tree than with a stochastic tree. This is in keeping with the view that good codes require a larger value of M to find their better paths. Second, the segSNR value for $M = 1$ with a deterministic tree is larger than that obtained with a stochastic tree. The stochastic code performs well with a multipath search, but gives very poor performance with a single path search. The segSNR value, once saturation with M is attained, is higher with a stochastic tree than with a deterministic tree. Subjective performance is also better with a stochastic tree, for large enough values of M. The results indicate that the choice of the code tree and the search algorithm are by no means independent design issues.

The objective performance with the inclusion of a pitch synthesis filter was investigated. In solving for the pitch predictor coefficients, the diagonal elements of the correlation matrix were perturbed using a value of $\alpha = 0.01$. Figure 4 shows a plot of segSNR versus time, both with and without the inclusion of a pitch synthesis filter. Note that an increase in segSNR of about 2–10 dB is attained during the voiced segments. The segSNR remains about the same during unvoiced segments.

6. Subjective Test Results

A subjective test of the coder was carried out by conducting a preference test between tree coded utterances and log-PCM coded utterances of various bit rates. The listeners consisted of mostly 'naive listeners' (students working in areas other than speech coding) and a few trained listeners (those working in the speech coding area). The 'naive listeners' were more inclined towards the tree coded speech than 'trained listeners'. The following parameters were used for the tree coded sentences: a 3 tap pitch predictor with noise perturbation adaptation ($\alpha = 0.01$), an 8 tap formant predictor using a one-pole error window ($\beta = 0.986$), noise shaping ($\mu = 0.85$) and tree searching with $M = 16$ and $L = 8$. The tree coded sentences were compared with 5, 6, 7, and 8 bit/sample log-PCM coded sentences. The subjective test presented pairs of sentences, a tree coded version and a log-PCM version, with both orderings represented. The various test pairs were randomly ordered in the presentation. Results of the subjective tests are shown in Fig. 5. The vertical axis shows the fraction of times that the tree coded sentences were preferred over the corresponding log-PCM coded sentences. For example, tree coded sentences were preferred over 5 bits/sample log-PCM coded sentences every time. The equal preference point is achieved at about 7 bit/sample log-PCM. One can therefore conclude that the tree coding scheme achieves a level of subjective quality equal to 7 bit/sample log-PCM.

Informal listening tests indicate that the addition of the pitch filter, while not generating a substantial overall improvement in quality, is beneficial in certain crucial segments of the test utterances. Subsequent to the formal subjective tests, an adaptive postfilter [12] was added to the system. This adaptive postfilter helps in a small way to further improve the quality of the reconstructed speech.

7. Summary

A code tree generated by a stochastically populated innovations tree with a backward adaptive gain, and backward adaptive synthesis filters were considered. The code tree was searched using the multipath (M,L) search algorithm. For large values of M, the stochastic code tree was found to give better performance than the deterministic tree, both objectively and subjectively. The addition of the pitch filter gives 2–10 dB increase in segSNR in the voiced segments. Subjective testing has shown that the coder attains a subjective quality equivalent to 7 bits/sample log-PCM, with an encoding delay of 8 samples (1 ms with an 8 kHz sampling rate).

References

1. J. B. Anderson, and J. B. Bodie, "Tree encoding of speech", *IEEE Transactions on Information Theory*, vol. IT-21, pp. 379–387, July 1975.
2. N. S. Jayant, and S. A. Christensen, "Tree encoding of speech using the (M,L)-algorithm and adaptive quantization", *IEEE Trans. Commun.*, vol. COM-26, pp. 1376–1379, Sept. 1978.
3. S. G. Wilson, and S. Husain, "Adaptive tree encoding of speech at 8000 bits/s with a frequency-weighted error criterion", *IEEE Trans. on Commun.*, vol. COM-27, pp. 165–170, Jan. 1979.
4. T. Svendsen, "Tree coding of the LPC residual", *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing*, pp. 10.11.1–10.11.4, San Diego, Cal., April 1984.
5. H. G. Fehn and P. Noll, "Multipath search coding of stationary signals with applications to speech", *IEEE Trans. on Commun.*, vol. COM-30, pp. 687–701, April 1982.
6. J. D. Gibson and G. B Haschke, "Adaptive code generators for tree coding of speech", *Proc. IEEE Int. Conf. Commun.*, pp. 1142–1146, Seattle, Wash., June 1987.
7. F. Jelinek and J. B. Anderson, "Instrumentable tree encoding of information sources", *IEEE Trans. Inform. Theory*, vol. IT-17, pp. 65–76, Jan. 1971.
8. B. S. Atal and M. R. Schroeder, "Predictive coding of speech signals and subjective error criteria", *IEEE Trans. Acoust. Speech, Signal Processing*, vol. ASSP-27, pp. 247–254, June 1979.
9. J. I. Makhoul and L. K. Cosell, "Adaptive lattice analysis of speech", *IEEE Trans. Acoust., Speech, and Signal Processing*, vol. ASSP-29, June 1981.
10. R. P. Ramachandran and P. Kabal, "Pitch prediction filters in speech coding", to appear *IEEE Trans. Acoust., Speech, Signal Processing*.
11. N. S. Jayant, "Adaptive Quantization with a One Word Memory", *Bell Syst. Tech. J.*, Vol. 52, pp. 1119–1144, Sept. 1973.
12. J.-H. Chen and A. Gersho, *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing*, pp. 2185–2188, Dallas, Texas, April 1987.

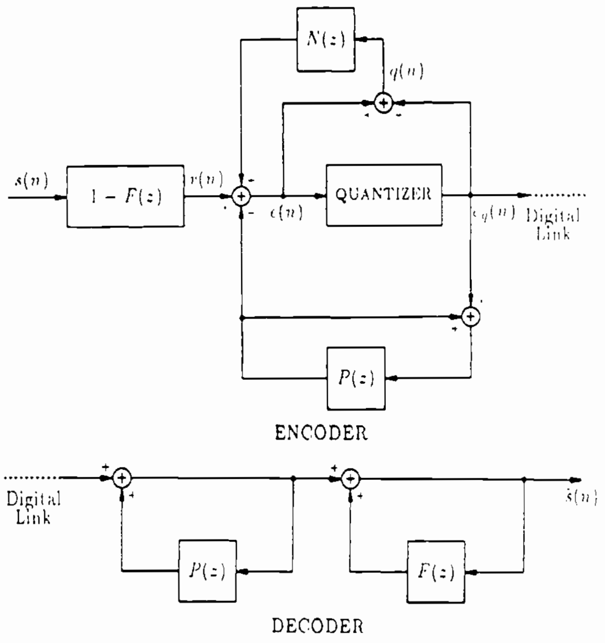

Fig. 1 Generalized predictive coder with pitch prediction

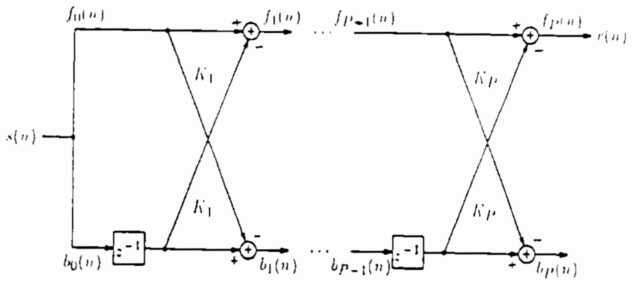

Fig. 2 Lattice filter

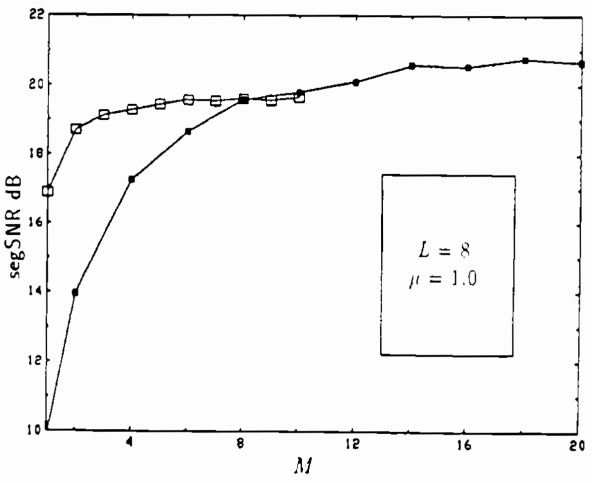

Fig. 3 Performance with M for the stochastic tree (shown by the filled squares) and determinstic tree (shown by the unfilled squares).

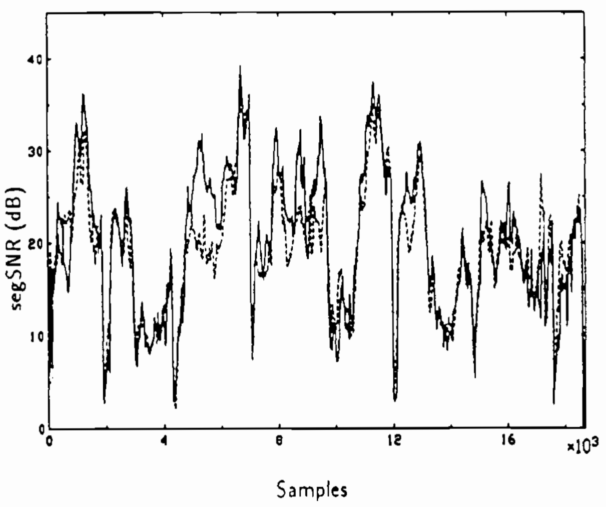

Fig. 4 Segmental SNR with pitch prediction. The dashed curve shows the segSNR without pitch prediction, while the solid curve shows the segSNR with pitch prediction.

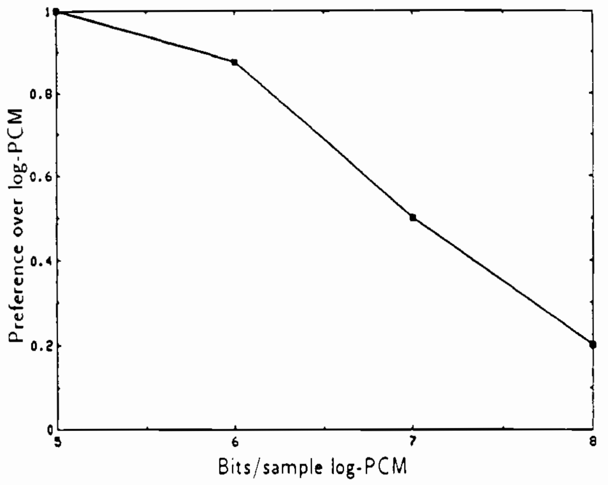

Fig. 5 Subjective preference curve (16 kbits/sec coder compared to log-PCM)