Fractional Rate Multitree Speech Coding

Jerry D. Gibson, *Senior Member, IEEE,* and Wen-Whei Chang, *Member, IEEE*

Abstract— **We present both forward and backward adaptive speech coders that operate at 9.6, 12, and 16 kb/s using integer and fractional rate trees, weighted squared error distortion measures, the (M, L) tree search algorithm, and incremental path map symbol release. We introduce the concept of multitree source codes and illustrate how the multitree structure allows scalar quantizer-based codes and scalar adaptation rules to be used for fractional rate tree coding. With a frequency weighted distortion measure, the forward and backward adaptive multitree coders produce near toll quality speech at 16 kb/s, while the backward adaptive 9.6 kb/s multitree coder substantially outperforms adaptive predictive coding and has an encoding delay less than 2 ms. Performance results are presented in terms of unweighted and weighted signal-to-noise ratio and segmental signal-to-noise ratio, sound spectrograms, and subjective listening tests.**

I. Introduction

THERE is considerable interest in speech coding at 4 to 16 kb/s for a wide variety of applications including speech storage, voice mail, military communications, commercial telephony, and land mobile radio. Important speech coding techniques for the upper half of this range (8–16 kb/s) are analysis-by-synthesis predictive coders, such as multipulse linear predictive coding (MLPC) and code-excited linear prediction (CELP) [1], [2], adaptive predictive coding with adaptive bit allocation (APC–AB) [3], [4], and subband coding [5]. Additionally, at 16 kb/s, the recent tree coder design of Iyengar and Kabal [6] and the predictive trellis coded quantization (TCQ) system of Marcellin *et al.* [7] offer good performance.

In this paper we present both forward and backward adaptive speech coding structures that operate at 9.6, 12, and 16 kb/s using integer and fractional rate tree codes. This work constitutes the first application of fractional rate trees to speech coding. Additionally, the introduction of the multitree structure allows the output values from standard scalar quantizers to be used as branch labels in fractional rate trees and provides a method whereby the familiar Jayant one-word memory, scalar quantizer step size adaptation rules can be used for fractional rate tree coding. In Section II, the basic components of a tree coder, namely, the code generator, the distortion measure, the tree search algorithm, and the path map symbol release rule, are described. Algorithms for forward and backward adaptation of the code generator are presented in Sections III and IV, respectively. Fractional rate tree codes are discussed, and the new multitree coders are developed, in Section V. Comparative performance results for various coder configurations are given in Section VI in terms of unweighted and weighted signal-to-noise ratio and segmental signal-to-noise ratio, sound spectrograms, and subjective listening tests.

Paper approved by the Editor for Quantization Speech/Image Coding of the IEEE Communications Society. Manuscript received February 21, 1989; revised June 19, 1990. This work was supported in part by BNR, Inc. through the University Interaction Program, Texas Instruments, Inc., Dallas, TX, and the National Science Foundation under Grant NCR-8914496. This paper was presented at the IEEE Global Telecommunications Conference, Dallas, TX, November 27–30, 1989 and the IEEE International Symposium on Information Theory, San Diego, CA, January 14–19, 1990.

J. D. Gibson is with the Department of Electrical Engineering, Texas A & M University, College Station, TX 77843.

W.-W. Chang is with the Department of Communication Engineering, National Chiao-Tung University, Taiwan, Republic of China.

IEEE Log Number 9144897.

II. Tree Coders

Predictive coders have been widely studied for speech coding at 8–32 kb/s [8]–[11]. A waveform encoding technique closely related to predictive coding is that of tree coding or delayed encoding. Classical predictive coding systems operate without delay in the sense that for an input sample at time instant k, only data at times $j \leq k$ are used in the encoding process. Tree coders attempt to improve on this approach by delaying the encoding decision for a few samples, say L, which allows the input samples at time instants $j \leq k+L$ to be used to encode the input sample at time k. Slight delays are often not critical to the operation of communication systems, and this delay allows all possible encoding sequences through time $k+L$ to be examined for a best fit. Each different encoding sequence is called a path, and hence, tree coding is a multipath search procedure whereas classical predictive coders exhibit a single path search [11].

The earliest investigations of multipath searching coders seem to be by Aughenbaugh, Irwin, and O'Neal [12] for synthetic sources and by Cutler [13] for television signals. Similar investigations followed [14], [15]. These studies consisted of using multipath searching in conjunction with a known coder structure such as delta modulation (DM) or differential pulse code modulation (DPCM), and hence, these approaches were called delayed decision systems or delayed encoding. The motivation for this work was the intuitive notion that looking ahead should provide better waveform following, and the desire to have a more responsive coder while still maintaining stability. Drawing upon rate distortion theory results [16], [17], Anderson and Bodie [18] studied DPCM-based tree coders for speech which used an efficient, instrumentable tree search algorithm called the (M, L) algorithm, and while they achieved notable increases in signal-to-quantization noise ratio (SNR) over DPCM, there was little or no improvement in output speech quality.

The filter or structure that synthesizes the coder output for a given path map sequence is called the code generator. Since Anderson and Bodie investigated only fixed code generators, that is, fixed quantizers and fixed predictors in DPCM,

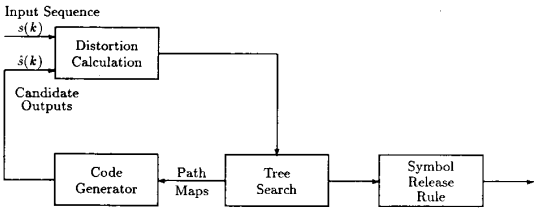


Fig. 1 Functional diagram of a tree coder.

subsequent research examined adaptive quantizers with fixed or adaptive predictors [19]–[26], [32]. Unfortunately, however, significant improvement in subjective performance was not obtained. Additional tree coding research for speech sources has considered tree search algorithms [27], [28], the distortion measure [23], the path map symbol release rule [25], synthetic speech-like sources [29]–[32], performance bounds [33], and stochastic codebooks [6].

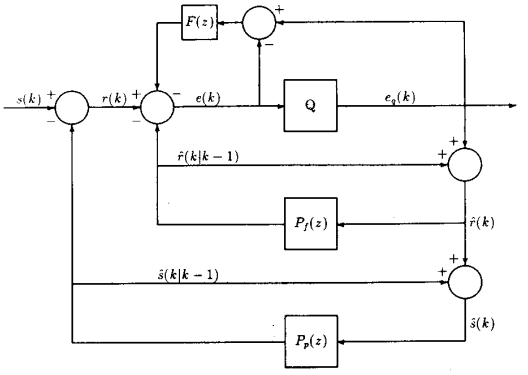
A functional block diagram of a tree coder (transmitter only) is shown in Fig. 1. The input to a data compression system is usually called the *source*, hence, the source sequence in Fig. 1 is $\{s(k)\}$. The distortion between the source sequence and each possible reconstructed sequence to some depth L in the tree is calculated, and the path through the tree with the smallest distortion (to depth L) is selected as the best path. Path map digits corresponding to this path (or some portion thereof) are then released as encoder output digits and sent to the decoder or receiver for reconstruction. The path map digits defining the minimum distortion path are also provided to the code generator at the encoder. The optimum or minimum distortion path is then extended to depth L, and the process is repeated. The source sequence is reconstructed (to some fidelity) at the receiver by applying the encoder output digits to the code generator input. Design of a tree coder consists of selecting a code generator, a distortion measure, a tree search algorithm, and a path map symbol release rule. We begin by developing a tree coder based upon an APC system code generator.

An APC system with noise spectral shaping is shown in Fig. 2 [11]. Note that APC is a single path search procedure, since at any time instant k, only one of the possible quantizer output levels is used for generating $\hat{s}(k)$. No other $\hat{s}(k)$ values are examined. An APC system can be used as the basis for a tree coder design, however. The part of an APC system transmitter which emulates the APC receiver can function as a code generator, and by delaying the transmission of $e_q(k)$ and basing the decision as to which $e_q(k)$ value to send on the distortion between $s(j)$ and $\hat{s}(j)$, for $j \leq k + L$, we obtain a tree coder. An APC based tree coder transmitter is illustrated in Fig. 3 where any spectral shaping is incorporated into the distortion measure. The receiver is unmodified.

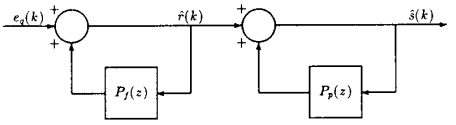
In Figs. 2 and 3, $P_p(z)$ represents the long-term or pitch predictor and is given by

$$P_p(z) = \beta_1 z^{-(M_1-1)} + \beta_2 z^{-M_1} + \beta_3 z^{-(M_1+1)} \quad (1)$$

where M_1 is the pitch period length and the $\{\beta_i, i = 1, 2, 3\}$ are weighting coefficients. We consider both forward and



(a) Transmitter



(b) Receiver

Fig. 2 APC with noise spectral shaping. (a) Transmitter. (b) Receiver.

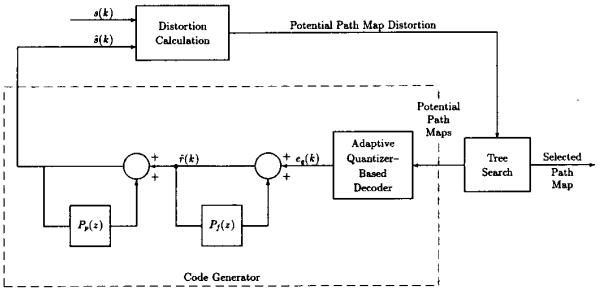


Fig. 3 APC-based tree coder (encoder or transmitter only).

backward updates of the weighting coefficients and M_1. The short-term or formant predictor has the form

$$P_f(z) = \sum_{i=1}^{N} a_i z^{-i} \quad (2)$$

where N is preselected (here typically, $N = 8$) and the coefficients $\{a_i, i = 1, 2, \cdots, N\}$ are calculated using either forward or backward adaptation as described in Sections III and IV, respectively. We mention here that the ordering of the predictors shown in Fig. 3 is sometimes reversed for APC systems in the literature. We found in some previous work on forward adaptive APC that the structure in Fig. 3 gave the best results [53] by a slight margin. This ordering also seems preferable for backward adaptation, since the algorithm for the short-term predictor coefficients adapts based upon a pitch-removed residual.

The "adaptive quantizer-based decoder" in Fig. 3 is a device that takes path map inputs and generates an output sequence

with values taken from the alphabet of an adaptive quantizer. Thus, for a four-level, integer rate 2 b/sample tree, the output values for $e_q(k)$ are computed from the output of a four level, minimum mean squared error (MMSE) Gaussian assumption quantizer with (forward or backward) adaptive step size $\Delta(k)$ [10].

The code generator output $\hat{s}(k)$ for all possible path map sequences to depth L is compared to the input signal $s(k)$ according to some distortion measure. The familiar single-letter, squared error distortion measure for each depth-L path is given by

$$d(s,\hat{s}) = d(s-\hat{s}) = \frac{1}{L}\sum_{i=1}^{L}[s(i)-\hat{s}(i)]^2 \tag{3}$$

where the $s(i)$ and $\hat{s}(i)$ in (3) refer to the values currently at depth i in the tree. We employ a weighted squared error criterion,

$$d((s-\hat{s})_w) = d(\varepsilon_w) = \frac{1}{L}\sum_{i=1}^{L}\varepsilon_w^2(i) \tag{4}$$

where $\varepsilon_w(k)$ is generated by passing $\varepsilon(k) = s(k) - \hat{s}(k)$ through a transfer function of the form [1], [11]

$$W(z) = \frac{1-\sum_{i=1}^{N} a_i z^{-i}}{1-\sum_{i=1}^{N}\mu^i a_i z^{-i}} \tag{5}$$

and μ is chosen by experiment to be 0.86.

The depth-L path with the smallest distortion can be found by exhaustively searching all possible paths to this depth, however, with 4 branches per level, this requires that 4^L paths be searched. Such exponential growth in search complexity can preclude the use of search depths L greater than 10, so it is common to employ alternative tree search strategies. One such algorithm is the (M,L) algorithm investigated by Anderson and Bodie [18] that only retains a fixed number M of paths at any depth. Only the (M,L) algorithm is used in the sequel.

Once the path through the tree to depth L that has the smallest distortion is found, path map symbols describing this path must be sent or released to the receiver. It is usual to release only a single symbol at any time instant [18], although some limited variable symbol release studies have been performed [25], [34]. Recent investigations on the exponential metric tree indicate that a single symbol release rule performs well [34]. All integer rate trees studied here use single symbol release, while the fractional rate trees release a fixed, small number of path map symbols at any time instant.

III. Forward Adaptation

We describe here techniques for calculating and quantizing the parameters of a forward adaptive APC code generator to be used in a fractional rate tree coding structure. The methods are the same ones used for single-path APC speech coders and have been previously investigated within that context [11]. The selections of the various particular parameter values, such as predictor orders and frame lengths, and the choice of quantization methods, are not claimed to be optimal, but are selected as examples to illustrate the flexibility in coder design afforded by using fractional rate tree codes. However, the performance studies presented in Section VI show that excellent results are obtained with the specifications given in this section.

For forward adaptation of the coder parameters $\{\beta_1, \beta_2, \beta_3, M_1, a_1, a_2, \cdots, a_N, \Delta\}$, we examine a frame or block of speech samples 20 or 25 ms long, depending on whether the sampling rate is 8000 or 6400 samples/s, respectively. The long-term predictor parameters $\beta_1, \beta_3, \beta_3$, and M_1 are calculated according to the techniques in [8], [11], [35], [37]. In particular, with reference to Fig. 3, the pitch lag M_1 is selected as that value of m that maximizes [37]

$$\rho = \left[\frac{\langle s(k)s(k-m)\rangle}{\langle s^2(k)\rangle\langle s^2(k-m)\rangle}\right]^{1/2} \tag{6}$$

where $\{s(k)\}$ is the input speech sequence, $\langle\cdot\rangle$ denotes time averaging over a frame length, and m varies over all pitch period values of interest (2 to 20 ms here). After M_1 is found, the coefficients $\{\beta_1, \beta_2, \beta_3\}$ are selected to minimize

$$\varepsilon_p^2 = \left\langle\left[s(k) - \sum_{i=1}^{3}\beta_i s(k-M_1-i+2)\right]^2\right\rangle \tag{7}$$

which yields the set of linear simultaneous equations [8], [11], [35] [see (8) below]
with

$$\phi(i,j) = \langle s(k-i)s(k-j)\rangle \tag{9}$$

where $\langle\cdot\rangle$ indicates averaging over all k in the frame. In order to guarantee the stability of this predictor while still maintaining a high prediction gain, we found it necessary to use the stability tests and scaling procedures developed by Ramachandran and Kabal [35]. We leave these details to the reference.

The input speech samples in the frame are passed through $1 - P_p(z) = 1 - \sum_{i=1}^{3}\beta_i z^{-M_1-i+2}$, and the resulting sequence is used in the autocorrelation method [8], [10] to compute the short term predictor coefficients $\{a_i, i = 1, \cdots, N\}$.

$$\begin{bmatrix} \phi(M_1-1,M_1-1) & \phi(M_1-1,M_1) & \phi(M_1-1,M_1+1) \\ \phi(M_1,M_1-1) & \phi(M_1,M_1) & \phi(M_1,M_1+1) \\ \phi(M_1+1,M_1-1) & \phi(M_1+1,M_1) & \phi(M_1+1,M_1+1) \end{bmatrix}\begin{bmatrix}\beta_1\\ \beta_2\\ \beta_3\end{bmatrix} = \begin{bmatrix}\phi(0,M_1-1)\\ \phi(0,M_1)\\ \phi(0,M_1+1)\end{bmatrix}, \tag{8}$$

More explicitly, the $\{a_i\}$ are chosen to minimize

$$\varepsilon_f^2 = \left\langle \left[v(k) - \sum_{i=1}^{N} a_i v(k-i) \right]^2 \right\rangle \tag{10}$$

where $v(k) = s(k) - \sum_{i=1}^{3} \beta_i s(k - M_1 - i + 2)$ and the time averaging is again over a frame length. Note that $r(k) = s(k) - \sum_{i=1}^{3} \beta_i \hat{s}(k - M_1 - i + 2)$ from Fig. 2 does not equal $v(k)$, since $r(k)$ uses past reconstructed values $\{\hat{s}(\cdot)\}$ in the prediction process. The minimization of ε_f^2 in (10) results in the set of linear simultaneous equations

$$\boldsymbol{\Phi} \boldsymbol{A} = \Psi \tag{11}$$

where $\boldsymbol{\Phi}$ is an $N \times N$ symmetric, Toeplitz matrix with components $\phi(i,j) = \phi(|i-j|) = \langle v(k)v(k+|i-j|)\rangle, i,j = 1,2, \cdots, N, \boldsymbol{\Psi}^T = [\phi(1)\phi(2)\cdots\phi(N)]$, and $\boldsymbol{A}^T = [a_1, a_2, \cdots, a_N]$. The step size Δ for the forward adaptive quantizer is computed from the resulting minimum mean squared prediction error according to

$$\Delta = \left\langle \left[v(k) - \sum_{i=1}^{N} a_i^{\text{opt}} v(k-i) \right]^2 \right\rangle^{1/2}. \tag{12}$$

Henceforth, we shall drop the superscript "opt" on the $\{a_i\}$ for economy.

At a sampling rate of 8000 samples/s, the frame length is chosen to be 20 ms. For transmission to the receiver, the short-term predictor coefficients are transformed into PARCOR or reflection coefficients and linearly quantized, the β_i and pitch are linearly quantized, and the step size is logarithmically quantized. With the bit allocations in Table I, the bit rate required for the side information is 3950 b/s. When this is combined with a rate 3/2 b/sample tree code, the total transmitted bit rate becomes 15950 b/s ~ 16 kb/s.

We note that there are numerous possible quantization methods for the several parameters to be transmitted as side information. For example, the partial correlation coefficients could be transformed using the inverse sine or the inverse hyperbolic tangent transformation and then uniformly quantized [11], [38], [39] or log area ratios [38], [39], or line spectrum pairs [40] could be quantized instead of reflection coefficients. Vector quantization techniques might also be used advantageously [41]. We have performed limited experiments with the scalar quantization approaches and have observed only slight differences in output speech. The methods adopted here are not claimed to be optimal, although they perform well, and the various methods should be carefully investigated for a given, particular application.

TABLE I
FORWARD ADAPTIVE CODING OF SIDE INFORMATION

Frame Bit Allocations	8000 samples/s 20 ms	6400 samples/s 25 ms
β_1	7	4
β_2	7	5
β_3	7	3
M_1	7	7
$\mathcal{K}_1$	8	7
$\mathcal{K}_2$	8	7
$\mathcal{K}_3$	7	6
$\mathcal{K}_4$	7	6
$\mathcal{K}_5$	4	3
$\mathcal{K}_6$	4	3
$\mathcal{K}_7$	4	2
$\mathcal{K}_8$	4	2
Δ	5	5

When operating at 6400 samples/s, the frame length is changed to 25 ms and the same methods for quantization of side information described previously are used. The bit allocations in Table I for this sampling rate and frame size thus give a required bit rate for the side information of 2400 b/s, so that with a rate 3/2 b/sample tree code, the total bit rate for this coder is 12 kb/s.

IV. BACKWARD ADAPTATION

Backward adaptation of the pitch parameters is only performed every 20 samples (2.5 ms at 8000 samples/s) due to the excessive computations required for more frequent updating [6], [36]. The pitch estimate at time instant k is determined by searching for the lag j that maximizes the normalized correlation function [36]

$$\gamma_k(j) = \frac{\phi_k(0,j)}{\sqrt{\phi_k(0,0)\phi_k(j,j)}} \tag{13}$$

where

$$\phi_k(i,j) = \sum_{m=1}^{J} \hat{s}(k - J + m - i)\hat{s}(k - J + m - j), \tag{14}$$

and J is the number of samples in the frame. The search range is limited to 2–20 ms, which covers most pitch periods encountered in speech. Note the differences between (6) and (13) and that the algorithm is backward adaptive because only past reconstructed samples are involved in (14).

After the pitch lag M_1 is determined, the pitch predictor coefficients, β_1, β_2, and β_3, are found by minimizing the sum of the squares of the pitch prediction residual over a frame of J samples. This minimization leads to the set of equations [see (15) below]

$$\begin{bmatrix} \phi_k(M_1-1,M_1-1) & \phi_k(M_1-1,M_1) & \phi_k(M_1-1,M_1+1) \\ \phi_k(M_1,M_1-1) & \phi_k(M_1,M_1) & \phi_k(M_1,M_1+1) \\ \phi_k(M_1+1,M_1-1) & \phi_k(M_1+1,M_1) & \phi_k(M_1+1,M_1+1) \end{bmatrix} \begin{bmatrix} \beta_1 \\ \beta_2 \\ \beta_3 \end{bmatrix} = \begin{bmatrix} \phi_k(0,M_1-1) \\ \phi_k(0,M_1) \\ \phi_k(0,M_1+1) \end{bmatrix} \tag{15}$$

which must be solved for the desired coefficients. As before, the resulting β_i's do not guarantee that $1/(1 - P_p(z))$ is stable, and so we employ the procedures in [35]. Further, backward adaptation implies that the frame over which the sum of the squared errors is minimized does not correspond to the frame over which the pitch predictor is applied. One possible approach to reducing adverse effects from this mismatch is to "soften" the predictor by introducing some pseudonoise term, which is accomplished by adding a small quantity to the diagonal elements of the 3 by 3 matrix in (15). Hence, we replace the diagonal elements $\phi_k(i,i)$ by $(1+\eta)\phi_k(i,i), i = M_1 - 1, M_1, M_1 + 1$, with $\eta = 0.001$ [36].

Clearly, (13) and (15) are variations on the standard forward adaptive techniques, and we mention that gradient-based, backward adaptive algorithms have been proposed and studied by Melsa *et al.* [42] (see also, [8]), Pettigrew and Cuperman [43], and Cuperman *et al.* [44]. Our studies with these algorithms are incomplete, but the results to date indicate that improved performance over the algorithms in (13)–(15) is possible.

There are a number of alternatives for the structure for the short-term or formant predictor. A fixed predictor and eight backward adaptive algorithms were considered for this predictor, including a second-order all-pole fixed predictor, the two-pole, six-zero CCITT 32 kb/s standard algorithm [10], a two-pole, six-zero adaptive gradient transversal predictor, a four-pole, ten-zero adaptive gradient transversal predictor, a fourth-order all-pole least squares lattice predictor [45], an eighth-order all-pole least squares lattice predictor [45], an eighth-order exponential window lattice predictor [6], [36], an eighth-order signal-driven lattice predictor [46], and an eighth-order residual-driven lattice predictor [46]. Some comparative performance results on the transversal predictors and the least squares lattice predictor for a DPCM code generator are given in [47], while comparisons among the four eighth-order lattice predictors are available in [48]. The general result is that the least squares, exponential window, and signal-driven lattices have essentially equivalent performance and all three outperform their transversal counterparts. The residual-driven lattice has a somewhat lower performance than the other lattice structures since it is designed to adapt only on $e_q(\cdot)$ values, as opposed to $\hat{s}(\cdot)$, and hence it is more robust to errors than the other three lattices which use $\hat{s}(\cdot)$ in their adaptation. Since we have not examined channel error effects and since the least squares, signal driven, and exponential window lattice algorithms have similar performance, we report results here only for the least squares lattice. However, depending upon the application, the signal driven or exponential window lattice algorithms may be preferable to the least squares lattice since they are less complex.

The lattice predictor structure is shown in Fig. 4 where the forward and backward prediction errors are updated by the recursions (the notation in the following refers to Fig. 4 only)

$$e_{l+1}(k) = e_l(k) - \mathcal{K}^b_{l+1}(k) r_l(k-1) \tag{16}$$

$$r_{l+1}(k) = r_l(k-1) - \mathcal{K}^f_{l+1}(k) e_l(k) \tag{17}$$

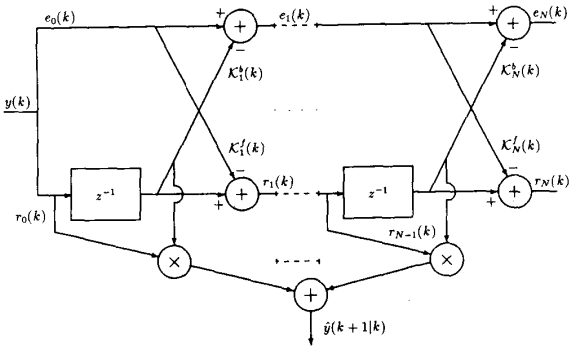


Fig. 4. Lattice form predictor structure.

with the predicted value computed as

$$\hat{y}(k+1\,|\,k) = \sum_{l=0}^{N-1} r_l(k) \mathcal{K}^b_{l+1}(k). \tag{18}$$

The adaptation of the coefficients proceeds as follows [45]. Begin with the initial conditions

$$\beta_0(k) = 0.0,$$
$$e_0(k) = r_0(k) = y(k)$$

where $y(k)$ is the DPCM output [$\hat{r}(k)$ in Fig. 3] and

$$R^b_1(k) = R^f_1(k) = 0.99 R^f_1(k-1) + e^2_0(k).$$

Perform the following recursions, in order, for $l = 1, 2, \cdots, N$:

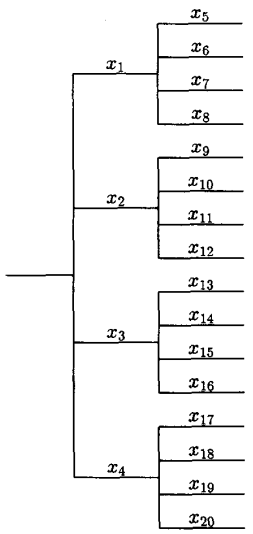
$$\begin{aligned}
\overline{\mathcal{K}}_l(k) &= 0.99\overline{\mathcal{K}}_l(k-1) \\
&\quad + (e_{l-1}(k) r_{l-1}(k-1)/(1-\beta_{l-1}(k))), \\
\mathcal{K}^f_l(k) &= \overline{\mathcal{K}}_l(k)/R^f_l(k), \\
\mathcal{K}^b_l(k) &= \overline{\mathcal{K}}_l(k)/R^b_l(k-1), \\
R^f_{l+1}(k) &= \left(R^f_l(k) - \mathcal{K}^b_l(k)\overline{\mathcal{K}}_l(k)\right)/0.98^2, \\
R^b_{l+1}(k) &= \left(R^b_l(k-1) - \mathcal{K}^f_l(k)\overline{\mathcal{K}}_l(k)\right)/0.98^2, \\
\beta_l(k) &= \beta_{l-1}(k) + r^2_{l-1}(k-1)/R^b_l(k-1), \\
e_l(k) &= e_{l-1}(k) - \mathcal{K}^b_l(k) r_{l-1}(k-1),
\end{aligned}$$

and

$$r_l(k) = r_{l-1}(k-1) - \mathcal{K}^f_l(k) e_{l-1}(k).$$

To complete the description of the backward adaptive code generator, we must specify the allowable values for the sequence $\{e_q(k)\}$ in Fig. 3, including the "step size" or gain adaptation algorithm for the "adaptive quantizer-based decoder" block. The step size adaptation algorithm is modified for the different trees studied, however, for a point of reference here, we give the backward adaptation rule for the integer rate 2, four-level per node tree. For this tree, the step size evolves according to the robust Jayant adaptive algorithm

$$\Delta(k+1) = \Delta^{\gamma}(k) F(|I(k)|) \tag{19}$$



$R = \frac{1}{1}\log_2 4 = 2$ bits/symbol

Fig. 5 Rate 2 b/symbol tree.

where the leakage factor γ is chosen to be $127/128$, and $F(\cdot)$ is a time-invariant multiplier function that is 0.8 for inner levels and 1.6 for outer levels. An extensive discussion of the branch labels in the tree is given in Section V.

V. Fractional Rate Tree Codes

Virtually all of the work performed on tree coding of speech has emphasized integer rate trees at rates 1 and 2 bits/symbol (sample) [6], [18], [23], [47]. Fig. 5 shows a rate $R = 2$ b/symbol tree with four levels per node and one symbol per branch to a depth $L = 2$. The symbols in this tree, and subsequently described trees, are possible $e_q(k)$ values. For APC and DPCM code generators, the branch symbols are usually taken to be the output values for a scalar quantizer, and as described in Section II, we choose the branch symbols for the tree in Fig. 5 to be the output values of a MMSE Gaussian quantizer [10].

The classical approach to achieving fractional rates is to place more than one symbol on each branch of the tree [17], which since the rate of a code tree is given by

$$R = \frac{1}{\beta}\log_2 \alpha \tag{20}$$

where α = number of branches per node and β = number symbols per branch, allows great flexibility in choosing a coding rate. A fractional rate tree with $R = 3/2$ b/symbol formed using this classical approach is shown in Fig. 6 where only the upper path is shown extended to depth 3 to allow greater detail. Of course, although not explicitly shown, all other paths would be similarly extended. Branch labels in Fig. 6 could be taken from a stochastic (random) codebook or from the codebook of a vector quantizer (VQ) designed for the $\{e_q(k)\}$ sequence [41], [49].

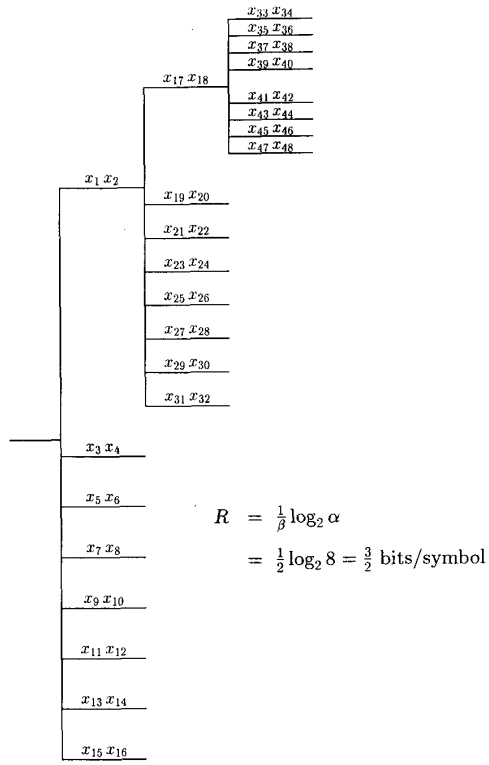


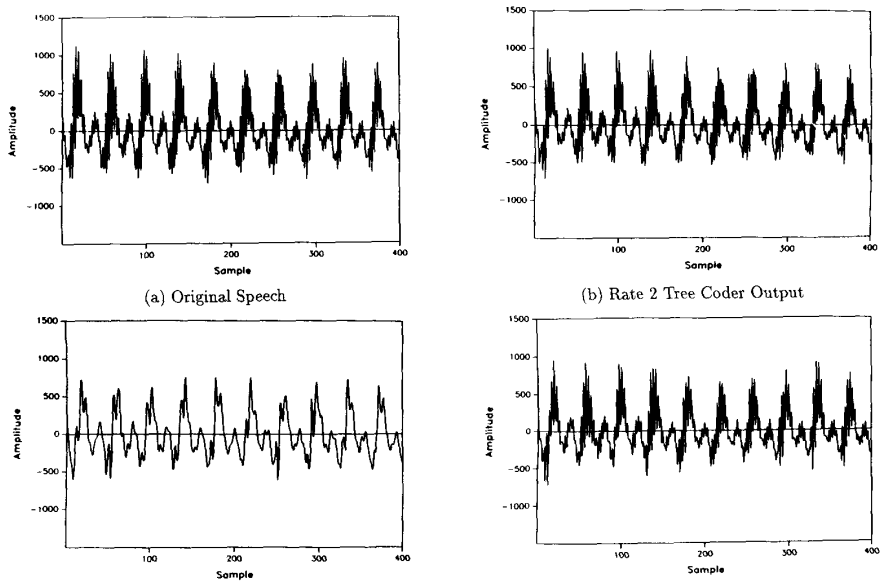
Fig. 6 Fractional rate $R = 3/2$ b/symbol tree.

Gain (or step size) adaptation is needed for both of these approaches, and in the former case for stochastic codebooks, the gain can be calculated using an algorithm like that employed in [6] for integer rates. Thus, the gain $\Delta(k)$ for stochastically populated fractional rate trees can be obtained as

$$\Delta^2(k+1) = \eta\Delta^2(k) + (1-\eta)e_q^2(k) \tag{21}$$

where $0 < \eta < 1$ and $e_q^2(k)$ is the branch symbol at time instant k. To adapt the gain for VQ codebooks one might choose one of the several techniques investigated by Chen and Gersho [50].

When one tries to use fractional rate trees as in Fig. 6 in conjunction with scalar quantization methods, two difficulties arise. First, when selecting branch symbols, we are confronted with the predicament of choosing all symbols on each branch to be the same. This is because scalar quantizer output values are specified according to which level they fall on, and these output levels correspond to branches in the tree. This situation is the same for both forward and backward adaptive quantizers. The second difficulty only occurs for backward adaptation of the step size. Consider the backward adaptation rule for the step size given in (19). Note that the step size expands or contracts depending upon whether the output value falls on an outer or inner level, respectively. This is no problem for trees with one $e_q(k)$ symbol per branch as in Fig. 5. However, for multiple symbols per branch as in Fig. 6, a method as in (19) will cause the step size to be expanded, contracted, or held



(a) Original Speech

(b) Rate 2 Tree Coder Output

(c) Rate 3/2 Classical Tree Coder Output

(d) Rate 3/2 Multi-Tree Coder Output

Fig. 7. Tree coder output waveform comparison. (a) Original speech. (b) Rate 2 tree coder output. (c) Rate 3/2 classical tree coder output. (d) Rate 3/2 multitree coder output.

constant for every symbol on the branch since the adaptation decision is made based on which branch occurs.

These two problems in applying scalar quantization methods to classical fractional rate trees cause a loss of high frequencies in the code generator excitation (the $\{e_q(k)\}$ sequence) and the synthesized output speech. To illustrate this problem, consider the waveforms shown in Fig. 7. Fig. 7(a) is a plot of a segment of original speech samples (taken at 8000 samples/s) and Fig. 7(b) is a plot of a backward adaptive tree coder output using the rate $R = 2$ b/sample tree in Fig. 5 and the code generator algorithms in Section IV, including (19) for the step size. In contrast, Fig. 7(c) is a plot of the output of a backward adaptive tree coder using the rate 3/2 b/sample tree code in Fig. 6 with scalar quantizer branch symbols and the backward adaptive algorithms in Section IV. The loss of high frequencies in going from Fig. 7(a) and (b) to Fig. 7(c) is clear. There is also a noticeable degradation in speech quality and intelligibility for this scalar-quantizer-based rate 3/2 tree code.

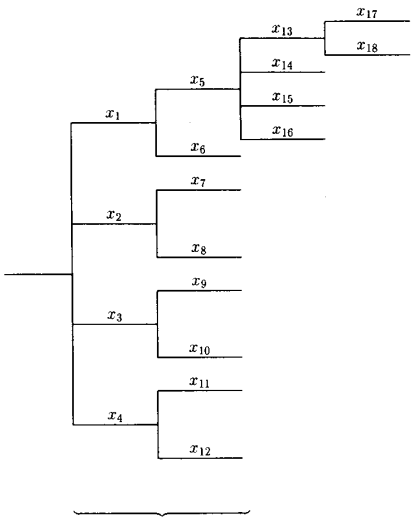
The backward adaptation of the step size in Fig. 7(c) used a Jayant-type algorithm for eight output levels similar to (19) that adapts depending upon which branch of the tree is followed, and we tried various modifications to the step size adaptation rule to cause a different adaptation by the second symbol on each branch. However, there is little information to guide the adaptation, and we could not generate speech much different than that in Fig. 7(c) for fractional rate trees like the one in Fig. 6.

To preserve the high frequencies and still have a fractional rate tree containing $e_q(k)$ symbols with scalar adaptation rules, we devised the concept of a multitree source code, which consists of different rate trees interleaved with each other. An example of a fractional rate multitree is shown in Fig. 8 where the number of branches emanating from a node is alternately 4 levels and 2 levels. With one symbol per branch, the rate of this tree code is the arithmetic mean of the rates of the component trees or $R = (2 \text{ b/symbol} + 1 \text{ b/symbol})/2$. Equation (20) can also be used to calculate the rate if we consider the nodes where the multitree structure repeats as supernodes and define α = number of paths out of a supernode (or between supernodes) and β = number of symbols per path between supernodes, so $R = \frac{1}{2}\log_2 8 = 3/2$ b/symbol.

To get the symbol values for the 4–2 multitree in the forward adaptive case, we use the 4-level and 2-level MMSE Gaussian quantizer characteristics [10], both scaled by the transmitted step size. For backward adaptation, the step size determined when a 4-level symbol occurs is used at the next time instant for a 2-level symbol and the step size calculated at the time of occurrence of a 2-level symbol is used for the 4-level symbol at the following time instant. We use (19) when a 4-level symbol occurs and a delta modulator adaptation scheme when a 2-level symbol occurs, given by [10]

$$\Delta(k+1) = \begin{cases} 1.5\Delta^{\gamma}(k), \operatorname{sgn}(e_q(k))\ \operatorname{sgn}(e_q(k-1)) = +1, \\ \frac{1}{1.5}\Delta^{\gamma}(k), \operatorname{sgn}(e_q(k))\ \operatorname{sgn}(e_q(k-1)) = -1 \end{cases} \tag{22}$$

where $\gamma = 127/128$ and $\operatorname{sgn}(\cdot) = +1$ for positive arguments and -1 for negative arguments. Thus, the step size is expanded if the polarity of the current symbol on the path and the polarity of the preceding symbol agree. If they differ, the step size is contracted. Fig. 7(d) is a plot of the output of a backward adaptive tree coder using the rate $R = 3/2$ multitree of Fig. 8. Note by comparison with Figs. 7(a)–(c) how the high frequencies have reappeared. Narrow-band spectrograms of the original speech, the rate 2 tree coder output, the classical rate 3/2 tree coder output, and the rate 3/2 multitree coder output



$$R = \tfrac{1}{2}\log_2 8 = \tfrac{3}{2} \text{ bits/symbol}$$

Fig. 8 Fractional rate $R = 3/2$ b/symbol multitree.

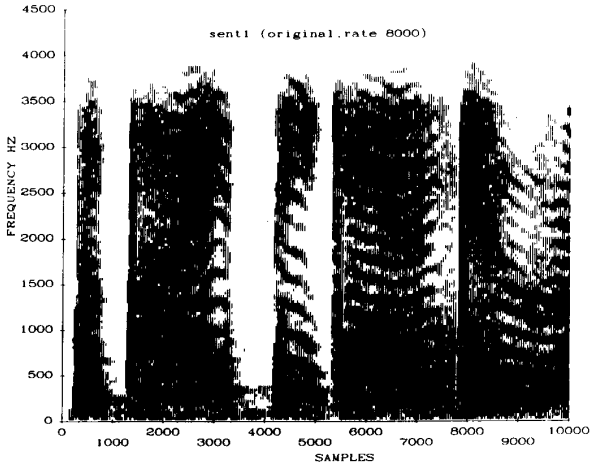


Fig. 9. Spectrogram of original speech segment (8000 samples/s).

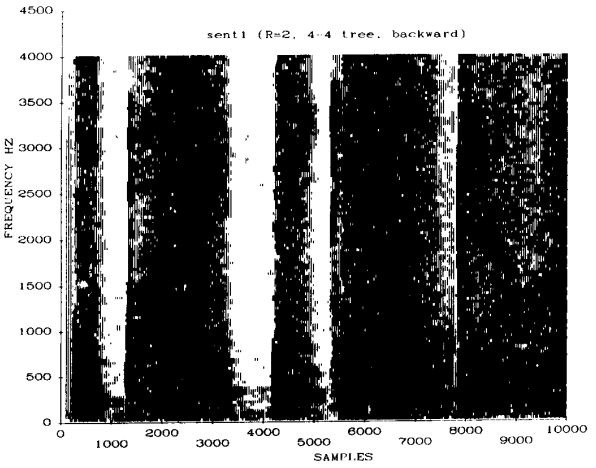


Fig. 10. Spectrogram of rate 2 backward adaptive tree coder output.

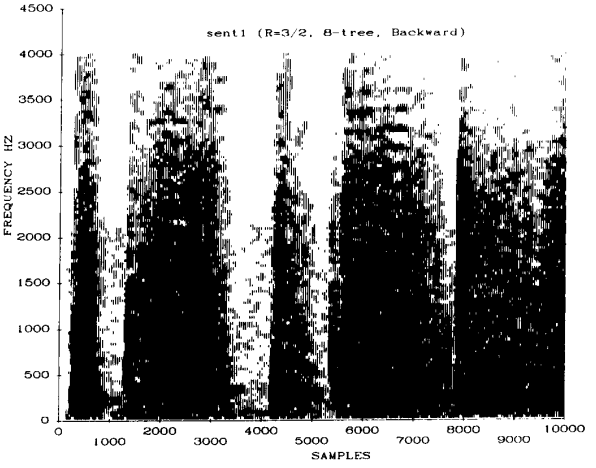


Fig. 11. Spectrogram of rate 3/2 tree coder with Fig. 6 tree.

are shown in Figs. 9–12 where the utility of the multitree concept is evident.

It is obvious from (20) that stochastically populated trees and VQ based trees can achieve virtually any desired rate by adjusting α and β. For example, with $\alpha = 2$ and $\beta = 2$, that is, two branches per node and two symbols per branch, we get $R = 1/2$ b/symbol. Multitree codes at low rates based on scalar quantizer principles can be obtained by using only one branch per node in one or more of the trees being interleaved, with the branch symbol being zero. Consider, for example, the multitree in Fig. 13 where nodes are indicated by black dots. This figure shows a rate 1/2 b/symbol multitree where the symbols on the two branch per node subtree could be the output levels of a two-level, Gaussian MMSE scalar quantizer and the one branch per node symbols could be 0. For forward adaptive code generators such a multitree is completely viable, however, for a backward adaptive code generator, it imposes some limitations. Specifically, with reference to Fig. 13, we see that the quantizer only has polarity information on every other sample. Therefore, a backward adaptation rule for the step size might be to use (22) with $e_q(k-1)$ replaced by $e_q(k-2)$, and to keep the step size unchanged when a zero level (one branch/node) occurs. Further, if too many zero levels occur (are used), the backward adaptation of the short-term predictor may be affected. To address this latter problem, it is possible to use dithering whenever a zero-level occurs, as described by Mark [51]. Since we are interested in data rates of 8 to 16 kb/s here, we do not investigate $R < 1$ b/symbol multitrees further.

Fractional rate tree codes, whether they have the classical structure in Fig. 6 with either stochastic or VQ-based codebooks or they have the multitree structure, greatly increase the options for waveform coder design. In particular, in those

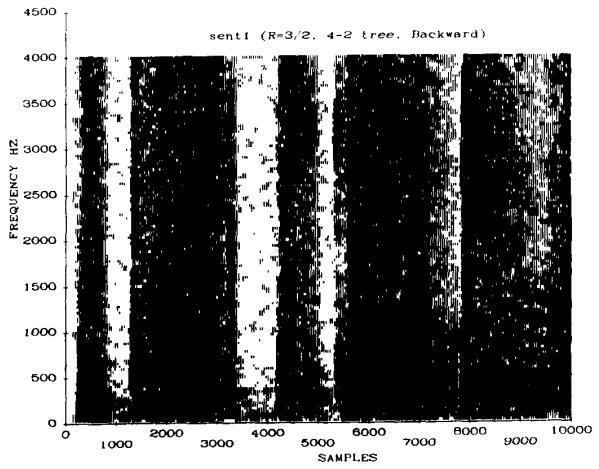


Fig. 12. Spectrogram of rate 3/2 multitree coder output.

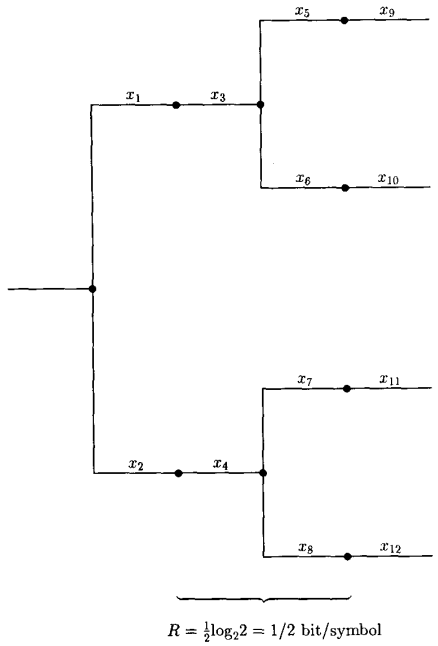


Fig. 13. Fractional rate $R = 1/2$ b/symbol multitree.

applications where the sampling rate is fixed and cannot be changed, such as some telephone network situations, virtually any data rate from 8 to 16 kb/s can be obtained by defining an appropriate rate tree. When the sampling rate can be changed, the fractional rate codes allow the designer to take advantage of this additional degree of freedom. For example, a data rate of 16 kb/s can be achieved with a backward adaptive coder by using a sampling rate of 8000 samples/s and $R = 2$ b/sample, a sampling rate of 6400 samples/s and 5/2 b/sample, or 10 000 samples/s and 8/5 b/sample, to name a few. Furthermore, each of these systems may have different subjective performance [48]. The fractional rate trees

TABLE II
OBJECTIVE PERFORMANCE OF 16 AND 12 KB/S FORWARD ADAPTIVE MULTITREE SPEECH CODERS

	SNR/SNRSEG/SNRFW (dB)	
	16 kb/s ($R = 3/2$ b/sample, 8000 samples/s)	12 kb/s ($R = 3/2$ b/sample, 6400 samples/s)
Male Speakers	17.22/17.78/18.64	15.32/15.49/16.39
Female Speakers	22.20/21.16/23.50	19.53/17.52/20.37
All Speakers	19.91/19.46/21.25	17.51/16.42/18.44

are perhaps even more valuable for forward adaptive coder design because the side information takes some portion of the available data rate. Fractional rate trees then allow the remaining data rate to be fully utilized.

The use of fractional rate trees with stochastic codebooks or VQ–based codebooks has not been reported, and hence, the fractional rate multitree coding results presented in this paper constitute the first investigation into fractional rate tree coding of speech. It is not evident at present which method for designing a fractional rate tree code, stochastic, VQ, or multitree, is preferable for the various applications and much research is needed on these topics.

Pearlman and Jakatdar [52] have previously investigated tree codes with varying branching factors and varying numbers of code letters per branch for transform coding of stationary Gaussian sources. The variation of the branching factor and the number of code letters per branch was used to achieve the desired bit allocation among the transform coefficients. Hence, their tree code is much different from the time-domain multitree codes introduced here.

VI. PERFORMANCE COMPARISONS

Extensive simulations were conducted to establish the performance attainable with the multitree codes developed in Section V for both forward and backward adaptation. The speech database for these studies consisted of the five sentences described in Appendix B sampled at both 8000 and 6400 samples/s. SNR/SNRSEG/SNRFW(dB) results are presented in Table II for forward adaptive multitree coders using the 4–2 multitree of Fig. 8 with the weighted distortion measure at 15.95 kb/s $\cong$ 16 and 12 kb/s. Bit allocations to achieve these rates are those in Table I. The (M, L) tree search algorithm with $M = 16$, $L = 8$ was employed for tree searching and the path map symbol release rule consisted of releasing both the first four-level symbol and the first two-level symbol on the best path. This is a variation of the incremental single symbol release rule that is somewhat analogous to the release of a single eight-level symbol in the classical rate 3/2 tree of Fig. 6. Results for the unweighted distortion measure are not given since the synthesized speech is of clearly lower quality than that for the weighted distortion measure.

Objective performance results for two backward adaptive 16 kb/s tree coders with the weighted distortion measure are presented in Table III. The integer rate 2 b/sample, 8000 sample/s tree coder uses a standard 4-level tree as in Fig. 5 and the $(M, L) = (8, 10)$ search algorithm with incremental single symbol release. The $R = 5/2$ b/sample,

TABLE III
OBJECTIVE PERFORMANCE OF BACKWARD ADAPTIVE 16 KB/S TREE AND MULTITREE SPEECH CODERS

	SNR/SNRSEG/SNRFW (dB)	
	16 kb/s ($R = 2$ b/sample, 8000 samples/s)	16 kb/s ($R = 5/2$ b/sample, 6400 samples/s)
Male Speakers	16.17/17.71/17.33	15.32/18.09/16.35
Female Speakers	22.15/21.56/23.12	22.59/22.21/23.14
All Speakers	19.56/19.67/20.59	19.69/20.22/20.34

TABLE IV
OBJECTIVE PERFORMANCE OF BACKWARD ADAPTIVE 12 AND 9.6 KB/S MULTITREE SPEECH CODERS

	SNR/SNRSEG/SNRFW (dB)	
	12 kb/s ($R = 3/2$ b/sample, 8000 samples/s)	9.6 kb/s ($R = 3/2$ b/sample, 6400 samples/s)
Male Speakers	14.00/14.40/15.06	12.42/13.21/13.56
Female Speakers	18.66/16.89/19.68	17.33/15.40/18.01
All Speakers	16.47/15.58/17.51	15.06/14.22/15.90

6400 samples/s coder employs an 8-4 multitree code with MMSE Gaussian scalar quantizer output levels, Jayant step size adaptation, the $(M, L) = (8, 10)$ search algorithm, and a two symbol (8-level/4-level) path map symbol release rule. Table IV contains objective performance results for 12 and 9.6 kb/s backward adaptive multitree coders with the weighted distortion measure. These coders employ the 4-2 multitree code of Fig. 8, the (M, L) tree search algorithm with $M = 8, L = 10$, and the 4-level/2-level path map symbol release rule used with the forward adaptive coders. The step size is adapted on four-level symbols according to (19), while the step size is adapted when a 2-level symbol occurs using the previous 4-level symbol polarity and the current 2-level symbol according to (22).

A few comments must be made concerning the objective results in Tables II–IV. First, comparisons should only be made between coders having the same sampling rate, since coders operating at different sampling rates have different input sequences, and hence, different reference sequences for the SNR/SNRSEG/SNRFW calculations. Second, although SNR and SNRSEG values are presented for completeness, these quantities are of lesser importance than SNRFW in Tables II–IV since the coders all employed a weighted distortion measure.

Comparing the SNRFW of the 16 kb/s forward adaptive (FA) coder in Table II and the $R = 2$ b/sample 16 kb/s backward adaptive (BA) coder in Table III, we see a slight advantage for the FA system. The usual caveat that the FA coder has a 160 sample delay compared to a 10 sample delay for the BA system is applicable here, and it must be noted that M and L also differ. Optimization over M and L for the FA and BA coders was not performed. The presence of the side information complicates the design of an FA coder at 9.6 kb/s, and we have not developed an FA coder at this rate as yet.

Results for a $R = 5/2$ b/sample, 6400 sample/s, 16kb/s BA multitree coder are also listed in Table III, and the SNRFW values are comparable to the other two 16 kb/s coders, however, since the sampling rate is different for this coder, SNRFW comparisons are subject to question. Subjective performance comparisons are valid for coders operating at different sampling rates, and general conclusions based upon informal subjective listening tests are that all three 16 kb/s coders in Tables II and III produce comparable speech quality, with perhaps a slight advantage going to the $R = 5/2$ b/sample multitree coder. We feel that the FA and BA 16 kb/s tree coders in Tables II and III approach the quality and intelligibility of multipulse LPC at 16 kb/s. Having also performed some limited simulation studies of the fully backward adaptive 16 kb/s coder of Iyengar and Kabal [6], [36], which is said to be toll quality, we believe that our forward and backward adaptive tree coders at 16 kb/s have equivalent performance. It is possible to make a more direct comparison to the 16 kb/s (2 b/sample, 8000 sample/s) trellis coded quantization (TCQ) system of Marcellin *et al.* [7], since that work was based upon the same five sentences used here. Generally, SNRSEG values from [7] are about the same as those given in Tables II and III, although the values in [7] seem to fluctuate less across utterances. Of course, the coders in Tables II and III were designed to optimize a weighted distortion measure and not SNRSEG, and therefore, this comparison is not precise. Informal subjective listening tests indicate that our tree coders and the TCQ coder [7] produce approximately the same speech quality at 16 kb/s. Two notes of caution concerning this comparison with the TCQ results in [7] are that the Marcellin *et al.* [7] system does not utilize a long-term predictor as we do here, and that the TCQ results as reported in [7] are based on a large 1024 sample delay because of the trellis search. Recent work by Marcellin and Fischer [54] includes a pitch loop and reduces the encoding delay of the TCQ system to 5 ms. The performance of this new coder is not substantially different from that in [7].

For both the FA and BA systems, there is an audible loss in quality as the data rate is reduced from 16 to 12 kb/s. The 9.6 kb/s coder in Table IV incurs a further decrement in output speech quality compared to the 12 kb/s multitree coders. However, the subjective quality and intelligibility at 9.6 kb/s is quite good with an extremely low level of granular noise. In comparison to 9.6 kb/s APC systems that we have studied extensively [53], the 9.6 kb/s multitree coder exhibits none of the spectral distortions of heavily center clipped APC systems and far less granular noise than SNRSEG optimized APC systems. We have not conducted performance comparisons with multipulse LPC and CELP at 9.6 kb/s.

The complexity of the backward adaptive system is greater than that of the forward adaptive system because of the necessity to adapt separately for each of the paths pursued. Of course, the BA coder delay is much less.

VII. CONCLUSION

Multitree structures for achieving fractional rates with scalar quantizer-based codes and scalar adaptation rules have been introduced. Their performance is evaluated for speech coding with deterministic code generators at 16 and 12 kb/s for

forward adaptive systems and at 16, 12, and 9.6 kb/s for backward adaptive systems. Using the (M, L) tree search algorithm and a frequency weighted distortion measure, the multitree coders yield speech ranging from near toll quality at 16 kb/s to speech with good quality and intelligibility at 9.6 kb/s. The 9.6 kb/s backward adaptive multitree coder substantially outperforms APC and has an encoding delay less than 2 ms.

Appendix A

The objective performance measures used in this work are the signal-to-noise ratio (SNR) defined as

$$\mathrm{SNR} = 10 \log_{10} \frac{\langle s^2(k) \rangle}{\left\langle [s(k) - \hat{s}(k)]^2 \right\rangle} \tag{A.1}$$

where $\langle \cdot \rangle$ denotes time averaging over the entire utterance, the segmental SNR(SNRSEG) given by

$$\mathrm{SNRSEG} = \frac{1}{K} \sum_{j=1}^{K} \mathrm{SNRB}_j \tag{A.2}$$

where SNRB_j is the SNR in (A.1) over the jth block of speech data, and the weighted SNR (SNRFW) calculated as

$$\mathrm{SNRFW} = 10 \log_{10} \frac{\langle s^2(k) \rangle}{\left\langle [(s(k) - \hat{s}(k))_w]^2 \right\rangle}. \tag{A.3}$$

Appendix B

The five sentences used in this work are as follows:

1) "The pipe began to rust while new." (Female speaker).
2) "Add the sum to the product of these three." (Female speaker).
3) "Oak is strong and also gives shade." (Male speaker).
4) "Thieves who rob friends deserve jail." (Male speaker).
5) "Cats and dogs each hate the other." (Male speaker).

Acknowledgment

The authors gratefully acknowledge stimulating discussions with Dr. P. Mermelstein concerning this work and the efforts of H. C. Woo and Y. C. Cheong for performing the simulations in Table III. The authors also appreciate the constructive comments of the referees and the Associate Editor.

References

[1] P. Kroon and E. F. Deprettere, "A class of analysis-by-synthesis predictive coders for high quality speech coding at rates between 4.8 and 16 kbits/s," *IEEE J. Select. Areas Commun.*, vol. 6, pp. 353–363, Feb. 1988.

[2] J.-H. Chen, "A robust low-delay speech coder at 16 Kbits/s," in *Conf. Rec., 1989 IEEE Global Telecommun. Conf.*, Dallas, TX, Nov. 27–30, pp. 1237–1241.

[3] M. Honda and F. Itakura, "Bit allocation in time and frequency domains for predictive coding of speech," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-32, pp. 456–473, June 1984.

[4] K. Irie, Y. Tada, and K. Honma, "APC-AB codec modules operating at 16 and 8 kbits/s," *IEEE J. Select. Areas Commun.*, vol. 6, pp. 383–390, Feb. 1988.

[5] R. V. Cox, S. L. Gay, Y. Shoham, S. R. Quackenbush, N. Seshadri, and N. S. Jayant, "New directions in subband coding," *IEEE J. Select. Areas Commun.*, vol. 6, pp. 391–409, Feb. 1988.

[6] V. Iyengar and P. Kabal, "A low delay 16 Kbits/sec. speech coder," in *Proc. 1988 IEEE Int. Conf. Acoust., Speech, Signal Processing*, New York, NY, Apr. 11–14, 1988, pp. 243–246.

[7] M. W. Marcellin, T. R. Fischer, and J. D. Gibson, "Predictive trellis coded quantization of speech," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. 38, pp. 46–55, Jan. 1990.

[8] J. D. Gibson, "Adaptive prediction in speech differential encoding systems," *Proc. IEEE*, vol. 68, pp. 488–525, Apr. 1980.

[9] ——, "Adaptive prediction for speech encoding," *ASSP Mag.*, Acoust., Speech, Signal Processing Soc., pp. 12–26, July 1984.

[10] N. S. Jayant and P. Noll, *Digital Coding of Waveforms: Principles and Applications to Speech and Video.* New York: Prentice-Hall, 1984.

[11] B. S. Atal, "Predictive coding of speech at low bit rates," *IEEE Trans. Commun.*, vol. COM-30, pp. 600–614, Apr. 1982.

[12] G. W. Aughenbaugh, J. D. Irwin, and J. B. O'Neal, Jr., "Delayed differential pulse code modulation," in *Conf. Rec., Princeton Conf. Inform. Sci.*, Mar. 1968, pp. 125–130.

[13] C. C. Cutler, "Delayed encoding: Stabilizer for adaptive coders," *IEEE Trans. Commun.*, vol. COM-19, pp. 898–907, Dec. 1971.

[14] L. H. Zetterberg and J. Uddenfeldt, "Adaptive deltamodulation with delayed decision," *IEEE Trans. Commun.*, vol. COM-22, pp. 1195–1198, Sept. 1974.

[15] T. S. Koubanitsas, "Application of the Viterbi algorithm to adaptive delta modulation with delayed decision," *Proc. IEEE*, vol. 63, pp. 1076–1077, July 1975.

[16] R. M. Gray, "Information rates of autoregressive processes," *IEEE Trans. Inform. Theory*, vol. IT-16, pp. 412–421, July 1970.

[17] T. Berger, *Rate-Distortion Theory.* Englewood Cliffs, NJ: Prentice-Hall, 1971.

[18] J. B. Anderson and J. B. Bodie, "Tree encoding of speech," *IEEE Trans. Inform. Theory*, vol. IT-21, pp. 379–387, July 1975.

[19] D. W. Becker and A. J. Viterbi, "Speech digitization and compression by adaptive predictive coding with delayed decision," in *Conf. Rec., Nat. Telecommun. Conf.*, 1975, pp. 46-18–46-23.

[20] A. J. Goldberg, "Predictive coding with delayed decision," in *Conf. Rec., 1977 IEEE Int. Conf. Acoust., Speech, Signal Processing*, 1977, pp. 405–408.

[21] R. S. Cheng, "Application of CVSD with delayed decision to narrowband/wideband tandem," in *Conf. Rec., 1977 IEEE Int. Conf. Acoust., Speech, Signal Processing*, 1977, pp. 437–439.

[22] 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.

[23] S. G. Wilson and S. Husain, "Adaptive tree encoding of speech at 8000 bps with a frequency-weighted error criterion," *IEEE Trans. Commun.*, vol. COM-27, pp. 165–170, Jan. 1979.

[24] S. G. Wilson, "Adaptive tree encoding of discrete-time sources with speech applications," in *Conf. Rec., 1978 Nat. Telecommun. Conf.*, 1978, pp. 19.5.1–19.5.5.

[25] A. C. Goris and J. D. Gibson, "Incremental tree coding of speech," *IEEE Trans. Inform. Theory*, vol. IT-27, pp. 511–516, July 1981.

[26] H. C. Chan and J. B. Anderson, "Adaptivity versus tree searching in DPCM," *IEEE Trans. Commun.*, vol. COM-30, pp. 1254–1259, May 1982.

[27] J. B. Anderson, "Recent advances in sequential encoding of analog waveforms," in *Conf. Rec., 1978 Nat. Telecommun. Conf.*, Birmingham, AL, Dec. 3-6, 1978, pp. 19.4.1–19.4.5.

[28] S. Mohan and J. B. Anderson, "Speech encoding by a stack algorithm," *IEEE Trans. Commun.*, vol. COM-28, pp. 825–830, June 1980.

[29] ——, "Computationally optimal metric-first code tree search algorithms," *IEEE Trans. Commun.*, vol. COM-32, pp. 710–717, June 1984.

[30] J. B. Anderson and C.-W. Law, "Real-number convolutional codes for speech-like quasi-stationary sources," *IEEE Trans. Inform. Theory.*, vol. IT-23, pp. 778–782, Nov. 1977.

[31] M. L. Sethia and J. B. Anderson, "Low-rate tree coding of autoregressive sources," *IEEE Trans. Inform. Theory*, vol. IT-29, pp. 279–284, Mar. 1983.

[32] H. G. Fehn and P. Noll, "Multipath search coding of stationary signals with applications to speech," *IEEE Trans. Commun.*, vol. COM-30, pp. 687–701, Apr. 1982.

[33] J. Uddenfeldt, "A performance bound for tree search coding of speech with minimum-phase codes," in *Conf. Rec., Int. Conf. Commun.*, 1978, pp. 34.2.1–34.2.5.

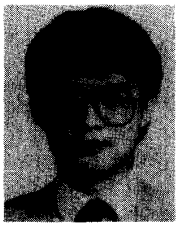
[34] W. W. Chang and J. D. Gibson, "Path map symbol release algorithms and the exponential metric tree," in *Proc., 1988 Conf. Advances in Commun. Contr. Syst.*, Baton Rouge, LA, Oct. 19–21, pp. 229–240.

[35] R. P. Ramachandran and P. Kabal, "Stability and performance analysis of pitch filters in speech coders," *IEEE Trans. Acoust., Speech, Signal Processing,* vol. ASSP-35, pp. 937–946, July 1987.
[36] V. Iyengar, "A low delay 16 kbit/s coder for speech signals," Master of Eng. thesis, Dep. Elec. Eng., McGill Univ., Aug. 1987.
[37] B. S. Atal and M. R. Schroeder, "Adaptive predictive coding of speech signals," *Bell Syst. Tech. J.,* vol. 49, pp. 1973–1986, Oct. 1970.
[38] R. Viswanathan and J. Makhoul, "Quantization properties of transmission parameters in linear predictive systems," *IEEE Trans. Acoust., Speech, Signal Processing,* vol. ASSP-23, pp. 309–321, June 1975.
[39] A. H. Gray, Jr., and J. D. Markel, "Quantization and bit allocation in speech processing," *IEEE Trans. Acoust., Speech, Signal Processing,* vol. ASSP-24, pp. 459–473, Dec. 1976.
[40] N. Sugamura and N. Farvardin, "Quantizer design in LSP speech analysis-synthesis," *IEEE J. Select. Areas Commun.,* vol. 6. pp. 432–440, Feb. 1988.
[41] J. Makhoul, S. Roucos, and H. Gish, "Vector quantization in speech coding," *Proc. IEEE,* vol. 73, pp. 1551–1588, 1985.
[42] J. L. Melsa *et al.* "Study of sequential estimation methods for speech digitization," Final Rep., Contract DCA 100-74-C-0037, Dep. Elec. Eng., Univ. Notre Dame, June 1975.
[43] R. Pettigrew and V. Cuperman, "Backward pitch prediction for low-delay speech coding," in *Conf. Rec., 1989 IEEE Global Telecommun. Conf.,* Nov. 27–30, Dallas, TX., pp 1247–1252.
[44] V. Cuperman *et al.* "Backward adaptation for low delay vector excitation coding of speech at 16 kbits/s," in *Conf. Rec., 1989 IEEE Global Telecommun. Conf.,* Nov. 27–30, Dallas, TX., pp. 1242–1246.
[45] R. C. Reininger and J. D. Gibson, "Backward adaptive lattice and transversal predictors in ADPCM," *IEEE Trans. Commun.,* vol. COM-33, pp. 74–82, Jan. 1985.
[46] P. Yatrou and P. Mermelstein, "Ensuring predictor tracking in ADPCM speech coders under noisy transmission conditions," *IEEE J. Select. Areas Commun.,* vol. 6, pp. 249–261, Feb. 1988.
[47] W. W. Chang and J. D. Gibson, "A comparison of adaptive code generators for tree coding of speech," *31st Midwest Symp. Circuits Syst.,* St. Louis, MO, Aug. 10–12, 1988.
[48] J. D. Gibson, Y. C. Cheong, H. C. Woo, and W.-W. Chang, "Backward adaptive prediction algorithms in multi-tree speech coders," in *Advances in Speech Coding,* B. S. Atal, V. Cuperman, and A. Gersho, Eds. New York: Kluwer, 1990.
[49] A. Gersho and V. Cuperman, "Vector quantization: A pattern-matching technique for speech coding," *IEEE Commun. Mag.,* vol. 21, pp. 15–21, Dec. 1983.
[50] J.-H. Chen and A. Gersho, "Gain adaptive vector quantization with application to speech coding," *IEEE Trans. Commun.,* vol. COM-35, pp. 918–930, Sept. 1987.
[51] J. W. Mark, "Adaptive predictive run-length encoding for analog sources," *Proc. IEE,* vol. 123, pp. 1189–1196, 1976.
[52] W. A. Pearlman and P. Jakatdar, "A transform tree code for stationary Gaussian sources," *IEEE Trans. Inform. Theory,* vol. IT-31, pp. 761–768, Nov. 1985.
[53] J. D. Gibson and W. W. Chang, "Objective and subjective optimization of APC system performance," *IEEE Trans. Acoust., Speech, Signal Processing,* vol. 48, June 1990.
[54] M. W. Marcellin and T. R. Fischer, "A trellis-searched 16 Kbit/sec speech coder with low delay," in *Advances in Speech Coding,* B. S. Atal, V. Cuperman, and A. Gersho, Eds. New York: Kluwer, 1990.

Jerry D. Gibson (S'73–M'73–SM'83) was born in Fort Worth, TX, on May 12, 1946. He received the B.S. degree in electrical engineering from the University of Texas at Austin in 1969, and the M.S. and Ph.D. degrees from Southern Methodist University, Dallas, TX, in 1971 and 1973, respectively.

He has held positions at General Dynamics-Fort Worth (1969–1972), the University of Notre Dame (1973–1974), the Defense Communications Agency (Summer 1974), and the University of Nebraska-Lincoln (1974–1976). In 1976 he joined Texas A&M University where he currently holds the J. W. Runyon, Jr., Professorship in the Department of Electrical Engineering.

Dr. Gibson is coauthor of the book *Introduction to Nonparametric Detection with Applications* (New York: Academic, 1975) and was Associate Editor for Speech Processing for the IEEE TRANSACTIONS ON COMMUNICATIONS from 1981 to 1985. He recently published the textbook *Principles of Digital and Analog Communications* (New York: Macmillan, 1989) and a chapter (with K. Sayood) entitled "Lattice Quantization" in *Advances in Electronics and Electron Physics* (New York: Academic, 1988). He is currently Associate Editor for Communications for the IEEE TRANSACTIONS ON INFORMATION THEORY and a member of the IEEE Information Theory Society Board of Governors (1990–1992). He is General Co-Chairman of the 1993 International Symposium on Information Theory to be held in San Antonio, TX. His research interests include speech processing, data compression, digital communications, and image processing. Dr. Gibson received the 1990 Frederick Emmons Terman Award from the American Society for Engineering Education. He is a member of Eta Kappa Nu and Sigma Xi.



Wen-Whei Chang (S'86–M'89) was born in Chungli, Taiwan, on December 4, 1958. He received the B.S. degree in communication engineering from National Chiao-Tung University, Hsinchu, Taiwan, in 1980, and the M. Eng. and Ph.D. degrees in electrical engineering from Texas A&M University, College Station, TX, in 1985 and 1989, respectively.

Since August 1989, he has been an Associate Professor in the Department of Communication Engineering at National Chiao-Tung University, Hsinchu, Taiwan. His current research interests are in data compression and speech synthesis.

Dr. Chang is a member of Tau Beta Pi.