nondecreasing, Shoham and Gersho [125] present an algorithm that finds the optimum bit allocation.

Note that in the case where the distortions do not obey (94), the minimum distortions for the different components after bit allocation will generally not be equal.

B. Vector Rotation for Correlated Sources

We saw above how one can take advantage of differences in component variances to decrease the distortion, for a given rate, over that of equal bit allocation. The development did not make any assumptions about possible dependence among components. We present below a procedure that takes advantage of any correlation that may exist among the vector components.

Fig. 16 shows a block diagram for the scalar quantization of a vector source, which includes a vector rotation $\boldsymbol{A}$ before quantization, followed by an inverse rotation $\boldsymbol{B}$ at the receiver. $\{q_k, 1 \leqslant k \leqslant N\}$ are N independent scalar quantizers that quantize the components of the rotated vector $\boldsymbol{u}$. In Section IV-A we considered the case of $\boldsymbol{A} = \boldsymbol{B} = \boldsymbol{I}$. Here we ask the question: What are the rotations $\boldsymbol{A}$ and $\boldsymbol{B}$ and the quantizers $\{q_k\}$ that minimize the distortion at the output, subject to a given bit rate? The answer to this question is not known for general distributions. However, for a source $\boldsymbol{x}$ whose components are jointly Gaussian, Huang and Schultheiss [65] and then Segall [120] showed that the optimal rotation $\boldsymbol{A}$ is given by a matrix $\boldsymbol{S}$ whose rows are the normalized eigenvectors of $\boldsymbol{\Gamma}_x$, the covariance matrix of the vector $\boldsymbol{x}$ (see (13)), and $\boldsymbol{B}$ is the inverse of $\boldsymbol{A}$

$$\boldsymbol{A} = \boldsymbol{S} \qquad \boldsymbol{B} = \boldsymbol{S}^{-1} \tag{101}$$

with

$$\boldsymbol{S}^{-1} = \boldsymbol{S}^T \tag{102}$$

and

$$\boldsymbol{S}\boldsymbol{\Gamma}_x\boldsymbol{S}^T = \boldsymbol{\lambda} = \operatorname{diag}[\lambda_1\lambda_2 \cdots \lambda_N] \tag{103}$$

where $\boldsymbol{\lambda}$ is a diagonal matrix whose elements are the eigenvalues of the covariance matrix $\boldsymbol{\Gamma}_x$. Equation (103) is a well-known property of a symmetric matrix [13]. The eigenmatrix $\boldsymbol{S}$ is known to be an orthogonal matrix. The rotated vector $\boldsymbol{u}$

$$\boldsymbol{u} = \boldsymbol{A}\boldsymbol{x} = \boldsymbol{S}\boldsymbol{x} \tag{104}$$

will then have a covariance matrix

$$\begin{aligned}\boldsymbol{\Gamma}_u &= \mathscr{E}\left[(\boldsymbol{u} - \bar{\boldsymbol{u}})(\boldsymbol{u} - \bar{\boldsymbol{u}})^T\right]\\ &= \boldsymbol{S}\boldsymbol{\Gamma}_x\boldsymbol{S}^T = \boldsymbol{\lambda}.\end{aligned} \tag{105}$$

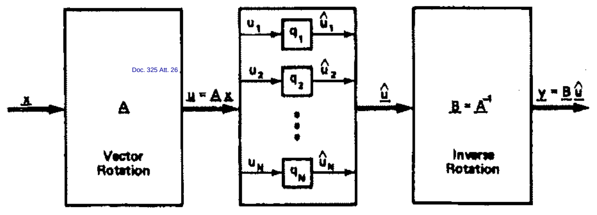

Fig. 16. Scalar quantization of a vector source. The eigenvector rotation $\boldsymbol{A}$ converts the input $\boldsymbol{x}$ into a vector $\boldsymbol{u}$ with uncorrelated components. An inverse rotation after the scalar quantizers gives $\boldsymbol{y}$, the quantized value of $\boldsymbol{x}$.

Therefore, the covariance of $\boldsymbol{u}$ is a diagonal matrix whose elements are the eigenvalues of the covariance $\boldsymbol{\Gamma}_x$. Since $\boldsymbol{\Gamma}_u$ is diagonal, the vector $\boldsymbol{u}$ has uncorrelated components, and since $\boldsymbol{x}$ is Gaussian, $\boldsymbol{u}$ will be Gaussian with independent components. The variance of each component u_k is then equal to the corresponding eigenvalue λ_k.

Using property (102) of an orthogonal matrix, one can show that the distortion at the receiver is equal to the distortion in quantizing the vector $\boldsymbol{u}$. Since

$$\boldsymbol{y} = \boldsymbol{S}^{-1}\,\hat{\boldsymbol{u}} \tag{106}$$

we have from (104), (106), and (102)

$$D = \frac{1}{N}\mathscr{E}\left[(\boldsymbol{x} - \boldsymbol{y})^T(\boldsymbol{x} - \boldsymbol{y})\right] = \frac{1}{N}\mathscr{E}\left[(\boldsymbol{u} - \hat{\boldsymbol{u}})^T(\boldsymbol{u} - \hat{\boldsymbol{u}})\right]. \tag{107}$$

Therefore, minimum distortion is obtained by designing quantizers that will minimize the average distortion in quantizing $\boldsymbol{u}$. The optimal scalar quantization scheme is then to quantize the components u_k using the bit-allocation scheme described in Section IV-A.

For the special case of high bit rates, the minimum distortion and the bit allocation are given by (95) and (96), respectively, with λ_k substituted for σ_k^2. The ratio of the minimum distortion after rotation $D'_{\min}$ to the equal bit allocation distortion D_0 is then given by

$$\frac{D'_{\min}}{D_0} = \frac{\left[\prod_{k=1}^{N}\lambda_k\right]^{1/N}}{\frac{1}{N}\sum_{k=1}^{N}\sigma_k^2}. \tag{108}$$

Note that, since [74]

$$\mathrm{GM}\{\lambda_k\} \leqslant \mathrm{GM}\{\sigma_k^2\} \tag{109}$$

we have from (98) and (108)

$$D'_{\min} \leqslant D_{\min} \tag{110}$$

with equality iff the components of the Gaussian vector $\boldsymbol{x}$ are uncorrelated and, hence, the variances are equal to the eigenvalues. In fact, the eigenvector rotation is the transform that minimizes the geometric mean of the resulting component variances. The same transformation, therefore, minimizes the quantization distortion. From (104) and (102), we can also write

$$\begin{aligned}\frac{1}{N}\sum_{k=1}^{N}\sigma_k^2 &= \frac{1}{N}\mathscr{E}\left[(\boldsymbol{x} - \bar{\boldsymbol{x}})^T(\boldsymbol{x} - \bar{\boldsymbol{x}})\right]\\ &= \frac{1}{N}\mathscr{E}\left[(\boldsymbol{u} - \bar{\boldsymbol{u}})^T(\boldsymbol{u} - \bar{\boldsymbol{u}})\right]\\ &= \frac{1}{N}\sum_{k=1}^{N}\lambda_k.\end{aligned} \tag{111}$$

Substituting in (108), we have

$$\frac{D'_{\min}}{D_0} = \frac{\mathrm{GM}\{\lambda_k\}}{\mathrm{AM}\{\lambda_k\}} \leqslant 1. \tag{112}$$

We note that there are many possible decorrelating rotations that can be applied to $\boldsymbol{x}$ to obtain a vector $\boldsymbol{u}$ whose components are uncorrelated. The eigenmatrix $\boldsymbol{S}$ is the one decorrelating matrix that results in the minimum distortion.

Exhibit Y Page 478

Optimal Scalar Quantization: Therefore, for a Gaussian vector process, the scalar quantization procedure that minimizes the mse for a given bit rate is

Step 1: Using the eigenmatrix $\boldsymbol{S}$ derived from the covariance of the vector process, transform (rotate) the input vector $\boldsymbol{x}$ to form another vector $\boldsymbol{u}$.

Step 2: Quantize the components of $\boldsymbol{u}$ using some optimal scalar quantizer and the bit-allocation procedure in Section IV-A. The result is the quantized vector $\hat{\boldsymbol{u}}$.

Step 3: Perform an inverse transform (rotation) on $\hat{\boldsymbol{u}}$ with the matrix $\boldsymbol{S}^T$, resulting in the output vector $\boldsymbol{y}$.

The above procedure is known to be optimal for Gaussian sources, but experience shows that the procedure works well for other distributions. In Example 1, we transformed the vector $\boldsymbol{x}$ in Fig. 5 into the vector $\boldsymbol{u}$ in Fig. 6. One can show that the rotation used is in fact an eigenvector rotation. Note how the marginal densities for the vector components change shape as we go from $\boldsymbol{x}$ to $\boldsymbol{u}$, and the components u_1 and u_2 in this case become independent. With some computation, one can show the interesting result in this example that the reduction in distortion achieved by the eigenvector rotation and the bit allocation is far greater than that predicted by (112). The reason, we believe, is the change in the shape of the marginal densities effected by the rotation.

Transform Coding: The transformation introduced by the eigenvector rotation is sometimes referred to as the *Karhunen–Loève Transform* (KLT) (see [74]), or simply the *eigenvector transform*. The transform involves the computation of the eigenvectors and eigenvalues of the covariance matrix. For a relatively small number of dimensions N, the computation of the eigenvectors can be accomplished via any of a number of standard routines [67]. The eigenvector transform has been used, for example, in quantizing the outputs of a filter bank in a speech vocoder [79] and in quantizing LPC log-area-ratio parameters [45], [112], [116] (see Section IV-C). However, as N increases to values in the hundreds, the amount of computation may make computing the eigenvectors prohibitively expensive. In such cases, one often resorts to simpler orthogonal transforms, such as the discrete cosine transform [3], which, even if not optimal, has been shown to produce very good results for speech and other data [74]. Adaptive transform coding (ATC) [16], [130], [142] is one of the well-known methods for the coding of speech waveforms in the range 8–16 kbits/s.

Distortion Weighting: In Section II-B we mentioned the possibility of using a weighting matrix $\boldsymbol{W}$ to weight the errors in specific components differently. The reason for utilizing such weighting is often guided by perceptual considerations. Minimizing the weighted mse in quantizing $\boldsymbol{x}$ is equivalent to minimizing the mse in quantizing the transformed vector $\tilde{\boldsymbol{x}}$ given in (16). The quantization procedure then follows Fig. 16 with $\tilde{\boldsymbol{x}}$ instead of $\boldsymbol{x}$, and $\boldsymbol{A}$ is the eigenmatrix corresponding to the covariance of $\tilde{\boldsymbol{x}}$, which can be shown to be

$$\boldsymbol{\Gamma}_{\tilde{x}} = \boldsymbol{P}\,\boldsymbol{\Gamma}_x\,\boldsymbol{P}^T \qquad (113)$$

where $\boldsymbol{P}$ is given by (15).

C. Comparisons with Vector Quantization

The performance of VQ in any specific application can be compared to scalar quantization to help assess the benefits accrued by VQ. Such comparisons are particularly important because of the substantial storage and computational costs typically associated with VQ. In this section we present an example to help illustrate the concepts presented and to see the specific benefits in an important application in speech coding.

EXAMPLE 4

In very-low-rate speech coding employing LPC analysis, one generally transmits for each frame a set of LPC coefficients, a gain term, a voiced/unvoiced decision, and a pitch value if the sound is voiced. At an analysis rate of 40 frames/s (i.e., every 25 ms) and 20 bits/frame, for example, the total bit rate will be 800 bits/s. In this paper we shall focus only on the quantization of spectral (LPC) parameters, since that is where vector quantization is mostly used. Typically, about 10–13 bits/frame are used to code the spectrum. Coincidentally, computational and storage costs tend to become unacceptable for applications requiring significantly more bits per frame to code the spectrum.

Fig. 17 shows the relative performance of scalar and vector quantizers for LPC parameters. Speech, low-pass filtered at 5 kHz and sampled at 10 kHz, was recorded from ten male speakers, 1 min of speech from each speaker. LPC analysis with 14 coefficients was performed at a rate of 100 frames/s. The resulting 60000 frames of data were used to train the scalar and vector quantizers. The LPC coefficients were represented by log-area-ratios (LARs) G_k defined in (22) and the distortion measure was the mse. The mse in Fig. 17 is plotted relative to the distortion at zero bits, which is equal to the average squared value of the LARs for all the data. Fig. 17 shows three plots of mse as a function of bit rate. Plot (a) was obtained using a separate Lloyd–Max

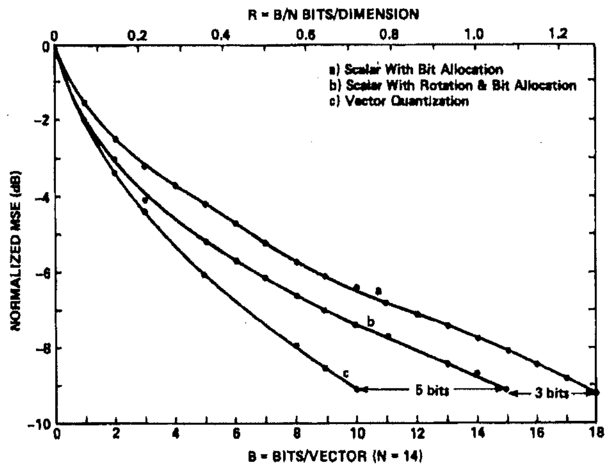

Fig. 17. Normalized mse in quantizing log-area-ratios (LARs) using three methods: (a) scalar quantization with bit allocation; (b) scalar quantization with bit allocation, preceded by eigenvector rotation; and (c) vector quantization. The 3-bit reduction from (a) to (b) takes advantage of linear dependencies (correlation), and the additional 5-bit reduction from (b) to (c) takes advantage largely of nonlinear dependencies.

Exhibit Y Page 479

scalar quantizer for each LAR with the integer bit-allocation scheme described in Section IV-A. Plot (*b*) was obtained by first performing a fixed eigenvector rotation, as described in Section IV-B, followed by bit allocation and Lloyd–Max scalar quantization. Plot (*c*) shows the performance of a vector quantizer whose codebook was designed using the *K*-means algorithm described in Section II-C. For the same mse obtained for the vector quantizer with 10 bits, the scalar quantizer with rotation requires 15 bits, while the scalar quantizer without rotation requires 18 bits. The benefit of 5–8 bits with vector quantization at $B = 10$ bits would be expected to increase as the number of bits is increased, until the slope of the vector quantization curve reaches −0.43 dB/bit (6.02/14 ≈ 0.43). However, in terms of percentage of total bit rate, the relative benefit of vector over scalar quantization decreases as the bit rate increases.

The 3-bit reduction in bit rate with eigenvector rotation shows the benefit of taking advantage of linear dependency. Much of the additional 5-bit reduction effected by vector quantization takes advantage of nonlinear dependencies among LARs. The existence of such significant nonlinear dependencies in the space of LARs (or other spectral parameters) for speech is what makes vector quantization truly attractive in this application.

The 10-bit vector quantization system and the 15-bit scalar quantization system with rotation were also compared using subjective listening tests. The analysis/synthesis employed variable-frame-rate transmission (see Section VI-A) at an average of 30 frames/s. Pitch, gain, and duration were unquantized. Upon informal listening, no clear difference could be detected between the two systems.

The bit allocation for the scalar quantizers utilized the pdfs of the different components. Fig. 18 shows the estimated (unnormalized) pdfs for G_1, G_2, and G_{14}. The pdfs for G_1 and G_2 are quite asymmetric and their variances are significantly larger than the other LARs. All pdfs for G_4 to G_{14} tend to be Gaussian in shape, as in the pdf of G_{14} shown in the figure. Table 2 shows the bit allocation before and after eigenvector rotation for the first 15 bits. The table shows the parameter index to which each additional bit is allocated. The parameters are the LARs for the case with no rotation and the transformed parameters after rotation (the index in the latter case is that of the corresponding eigenvector with the eigenvalues ordered in decreasing value). For example, the seventh bit was allocated to the sixth LAR in one case and to the parameter corresponding to the fourth eigenvector in the other. Table 3 shows the total number of bits allocated to each parameter for the 15-bit case. Generally, the parameters with larger variance are allocated more bits. Note in both cases how not all parame-

Table 2 Bit Allocation Before and After the Eigenvector Rotation of LARs. The table shows the parameter index to which each bit is allocated.

	Parameter Index (k)	
Bit Number	No Rotation	With Rotation
1	1	1
2	2	2
3	1	1
4	2	2
5	4	3
6	3	1
7	6	4
8	1	5
9	5	6
10	3	3
11	8	7
12	4	8
13	2	4
14	10	2
15	6	9

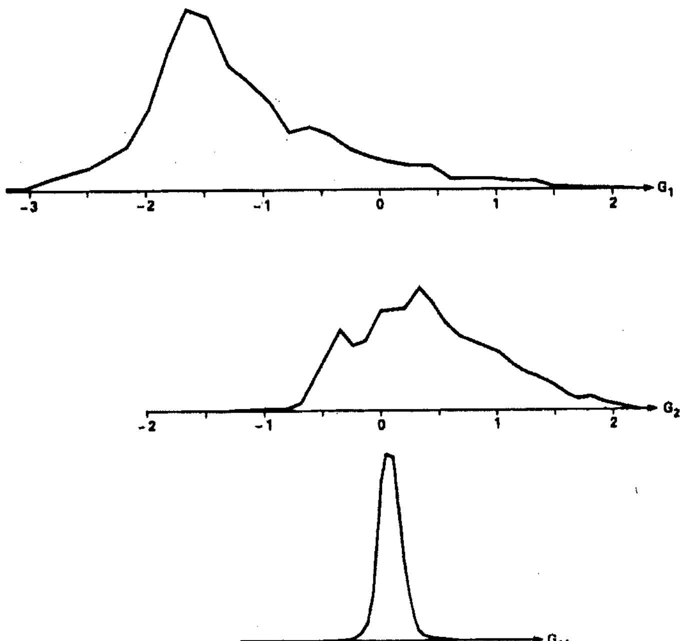

Fig. 18. Histograms of LARs G_1, G_2, and G_{14}. Histograms for G_4 through G_{14} can be modeled well by Gaussian pdfs.

Exhibit Y Page 480

Table 3 Total Number of Bits Allocated to Each Parameter for the 15-Bit Case

Parameter Index (k)	Number of Bits: No Rotation	Number of Bits: With Rotation
1	3	3
2	3	3
3	2	2
4	2	2
5	1	1
6	2	1
7	0	1
8	1	1
9	0	1
10	1	0
11–14	0	0
	15 bits	15 bits

ters are allocated bits. Parameters whose variances are smaller than the expected distortion are allocated zero bits.

Distortion Measure: In the example above, the mse was used in measuring the distortion of quantizing LARs. The same distortion measure was used for the vector and the scalar quantizers. Using the *same* distortion measure to assess the relative performance of quantizers is very important; otherwise, one can obtain misleading results. However, one freedom we have in vector quantization is to choose distortion measures that are not usually defined for the scalar case, such as the Itakura–Saito distortion in (23). It would be unfair, for example, to use the Itakura–Saito distortion to compare the performance of vector and scalar quantizers unless they were both designed using that distortion measure. Unfortunately, it is not simple to design a scalar quantizer using the Itakura–Saito distortion.[11] One, of course, could always compare the performance of any two quantizers fairly by subjective listening tests. We have designed LPC vector quantizers using the Itakura–Saito distortion as well as the mse on LARs and found no clear difference in subjective speech quality between the two methods.

Dimensionality and Bit Allocation: It was clear in the example above that if sufficient bits are not available, then certain vector components may not be allocated any bits with scalar quantization. For cases where vector components are uncorrelated and of equal variance, and $R < 1$ bit, the bit-allocation procedure will have to assign 1 bit to each of several components in some arbitrary fashion and assign zero bits to the other components, such that the average bit rate is R. The major limitation we are witnessing here is that a scalar quantizer cannot assign a fraction of a bit to a component; it must assign either 1 bit or none corresponding to either a two-level or a one-level quantizer (a fractional number of levels is not possible). (The only exception is if entropy coding is used, in which case fractional bits are possible.) A major advantage of the dimensionality of vectors is that fractional bits come in a very natural way, and all vector components are allocated bits in an equitable fashion. It is this aspect of dimensionality, in addition to the ability to specify arbitrary cell shapes, that makes VQ especially important at low rates. These properties of dimensionality are what makes it possible to code the prediction residual at $R < 1$ bit/sample and maintain good speech quality (see Section VII-B).

Vector Sources: The sequence of LPC-parameter vectors in Example 4 constitutes what one might consider as a "naturally occurring" vector source. Each vector is not merely a collection of scalars from a scalar source, but somehow represents a single entity, namely, the short-term spectrum. The VQ of this vector source as described above takes advantage of the dependencies among vector parameters *in parameter space*. VQ theory allows us, however, to form longer vectors from sequences of LPC vectors and perform VQ on the longer vectors, thereby taking advantage of dependencies *in time* between adjacent LPC vectors, and benefiting from the higher dimensionality. (This is precisely what is done in segment quantization, as described in Section VI.) However, the basic VQ process and its properties remain the same whether it is performed on an inherently scalar or vector source.

V. Codebook Design

It should be clear now that VQ can offer substantial performance advantages over scalar quantization at very low rates, especially for sources that exhibit significant nonlinear dependencies. Unfortunately, these advantages are obtained at considerable computational and storage costs, as we saw in Section II-D. For full-search coding, the costs are exponential in the number of bits per vector. Computational and storage costs double for each increase of 1 bit in the rate. In Example 4, to perform 10-bit quantization of vectors of 14 LARs, it takes $14 \times 2^{10} = 14\,336$ multiply-adds to quantize *each* input vector. The codebook requires an equal number of storage locations. This number doubles if 11-bit quantization is desired. Very quickly, computational and storage costs become prohibitively expensive as the number of bits/vector increases.

A number of fast-search algorithms have been proposed in the pattern-recognition literature [44], [121], [140] and more recently in VQ [20], [48], which are designed to reduce the computations in a full search. Most of the algorithms are based on geometrical notions in Euclidean spaces, they require preprocessing of the codebook, and tend to trade off multiplications with comparisons and with increased storage requirements. The number of multiplications can be reduced by as much as an order of magnitude.

In this section we present variations on the basic VQ scheme which are intended to reduce computational costs in a very significant manner (linear rather than exponential growth with the number of bits per vector), but at some reduction in performance. Other methods that are designed to reduce storage costs result in relatively greater reduction in performance. The section ends with a discussion of training and testing issues and codebook robustness.

A. *Binary Search*

With the K-means algorithm (with $K = L$ levels), a *full search* of the L code vectors is required to quantize each

[11]Strictly speaking, one cannot design an optimal scalar quantizer using the Itakura–Saito distortion, because scalar quantization implies the independent quantization of vector parameters and the Itakura–Saito distortion is not separable into distortions on the individual parameters. However, one could design a product code instead (see Section V-C).

Exhibit Y Page 481

input vector. *Binary search* [18], [112], known in the pattern-recognition literature as hierarchical clustering [5], [64], is a method for partitioning space in such a way that the search for the minimum-distortion code vector is proportional to $\log_2 L$ rather than L. Specifically, N-dimensional space is first divided into two regions (using the K-means algorithm with $K = 2$), then each of the two regions is divided further into two subregions, and so on, until the space is divided into L regions or cells. Here, L is restricted to be a power of 2, $L = 2^B$, where B is an integral number of bits. (A relaxation of this condition will be discussed below.) Associated with each region at each binary division is its centroid. Fig. 19 is a schematic of binary division of space into $L = 8$ cells. At the first binary

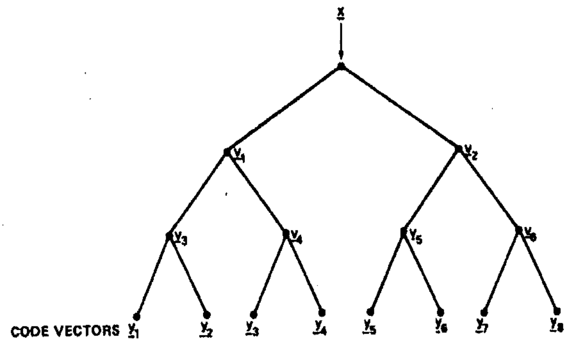

Fig. 19. Uniform tree for a binary-search vector quantizer. The vectors v_i are intermediate code vectors that are compared with the input vector x. The code vectors are the vectors y_i. The codebook size L is restricted to be a power of 2 in a uniform binary search.

division, v_1 and v_2 are the region centroids. At the second binary division, there are four regions with centroids v_3 through v_6. The centroids of the regions after the third binary division are the code vectors y_i. An input vector x is quantized by traversing (searching) the tree in Fig. 19 along a path that gives the minimum distortion at each node in the path. Thus x is compared to v_1 and v_2. If $d(x, v_2) < d(x, v_1)$, for example, then the path leading to v_2 is taken. Next x is compared to v_5 and v_6. If, for example, $d(x, v_5) < d(x, v_6)$, then the path to v_5 is taken and x is finally compared to y_5 and y_6. If $d(x, y_6) < d(x, y_5)$, then y_6 is chosen as the quantized value of x. Clearly, the total number of distortion computations is, in general, equal to $2 \log_2 L$. Again assuming N multiply-adds for each distortion computation, we have a total computational cost of

$$\mathscr{C} = 2N \log_2 L = 2NB \qquad \text{(binary search)} \qquad (114)$$

which is only *linear* with the number of bits. In the example of 10-bit quantization of 14 LARs, the number of computations is now only 280, as compared to 14336 with full search. (The 280 figure is close to the cost of scalar quantization when rotation is used.) Thus a vast reduction in computation has been effected. The storage cost, however, has increased. Note in Fig. 19 that, in addition to storing the code vectors y_i, one must also store all the intermediate vectors v. The total storage cost can be shown to have approximately doubled:

$$\mathscr{M} = 2N(L - 2) \qquad \text{(binary search)}. \qquad (115)$$

Also, we shall see below that an additional price is that a certain loss in performance takes place.

The cost requirements in (114) and (115) can be cut in half if the mse is used as the distortion measure. In that case, instead of comparing x to two vectors, one can test where x lies relative to the hyperplane that separates the two regions. Such a test involves the computation of a single scalar dot product of two vectors.

We call the quantization tree depicted in Fig. 19 *uniform*, in that *all* regions at any stage are each divided into two subregions. When performing the actual training to obtain the quantization tree, there may result one or more clusters at certain points in the subdivision process which contain very few training samples, perhaps even one sample. Such clusters would be essentially wasting bits because any further subdivision of those clusters would not reduce the distortion. To ensure a lower average distortion and maximize the utilization of bits, we do not branch the tree uniformly. Specifically, in the training phase, at each point in the subdivision process, the total distortion contributed by each cluster is examined. The cluster that contributes the largest amount of distortion is subdivided next, and the process is repeated. The result is typically a *nonuniform* tree, as depicted in Fig. 20. Note that, in this case, the

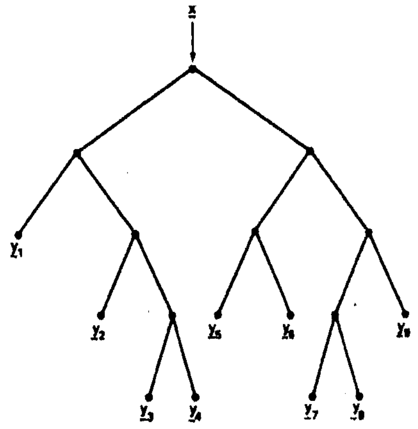

Fig. 20. Nonuniform tree for a binary-search vector quantizer. The codebook size in this case can be any integer.

number of levels can be any desired integer; it is not restricted to be an integral power of 2. A nonuniform tree with $L = 9$ is shown in Fig. 20.

To evaluate the relative performance of binary search (uniform and nonuniform) compared to the full search we give a specific example from very-low-rate speech coding. Fig. 21 shows the average distortion as a function of the number of bits per vector for the quantization of LPC frames of 14 LARs each. As expected, for a given distortion, full search has the lowest rate, followed by nonuniform binary, followed by uniform binary. For the highest rates shown in Fig. 21, the difference between full search and nonuniform binary is approximately 0.5 bits only. For many applications, this small difference in performance is well worth the savings in computation effected by binary search. The plots in Fig. 21 were obtained from the speech of 15 male speakers. For a single speaker, the difference between nonuniform binary and full search is typically larger (about 0.7 bits at $B = 8$ bits).

Binary search is a special case of a class of VQ methods known as *tree-searched* VQ, with the binary search being the simplest. In general, one could divide the space at each node in the tree into more than two subregions (using

Exhibit Y Page 482

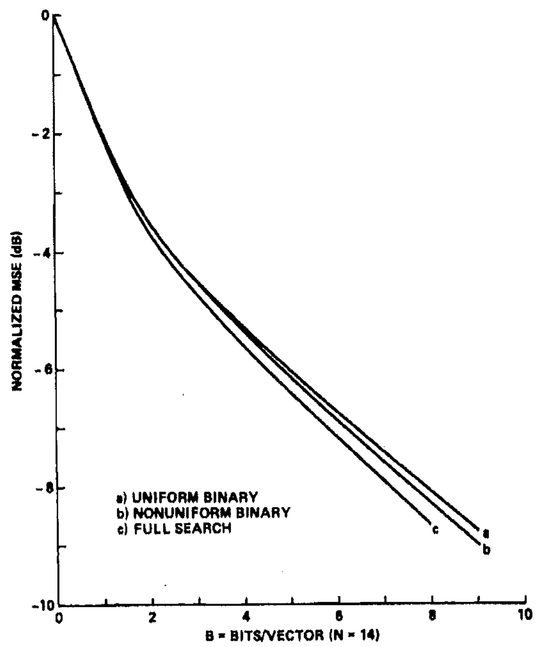

Fig. 21. Comparison of mse when quantizing LARs with three types of vector quantization: (*a*) uniform binary search; (*b*) nonuniform binary search; and (*c*) full search.

K-means with $K > 2$). Such a method would result in an increase in computation over binary search, with some increase in performance. However, because the performance of binary search is usually quite close to full search for many applications, it is typically most cost effective to use binary search with a nonuniform tree.

B. Cascaded Quantization

The major advantage of binary search was the substantial decrease in computational cost relative to full search, accompanied by a relatively small decrease in performance. However, the storage cost was not reduced; in fact, for non-mse distortions the memory cost doubled relative to full search. *Cascaded VQ* is a method intended to reduce storage as well as computational cost [76], [112]. As the name implies, cascaded (also known as multistage) VQ consists of a sequence of VQ stages, each operating on the "residual" of the previous stage. A two-stage cascade is depicted in Fig. 22. The input vector $\boldsymbol{x}$ is first quantized using a B_1-bit (L_1-level) vector quantizer (the quantizer can use full search or binary search, as desired). The residual or "error" $\boldsymbol{e}$ between $\boldsymbol{x}$ and its quantized value $\boldsymbol{z}_i$ is then used as the input to a B_2-bit (L_2-level) second VQ stage with output $\boldsymbol{w}_j$. (The matrix $\boldsymbol{A}_i$ plays a useful role that is explained below; here, we assume that $\boldsymbol{A}_i = \boldsymbol{I}$ and $\boldsymbol{u} = \boldsymbol{e}$.) The final quantized value of $\boldsymbol{x}$ is then simply the sum of the two vectors $\boldsymbol{z}_i$ and $\boldsymbol{w}_j$

$$q(\boldsymbol{x}) = \boldsymbol{y} = \boldsymbol{z}_i + \boldsymbol{w}_j \qquad (\boldsymbol{A}_i = \boldsymbol{I}). \qquad (116)$$

During the training phase, the residuals $\boldsymbol{e}$ from the first stage are pooled together and treated as a new random vector that is to be quantized. The process can be repeated for as many stages as desired.

There can be confusion between tree-searched VQ and cascaded VQ. While both processes take place in stages, what one does at each stage is different in the two methods. In tree-searched VQ, one is trying to find the desired

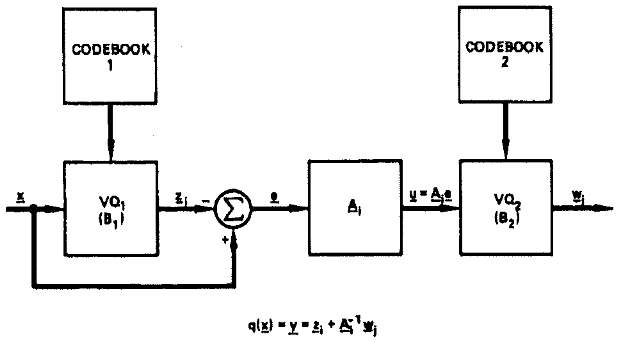

Fig. 22. A two-stage cascaded vector quantizer. In the first stage, a B_1-bit vector quantizer quantizes $\boldsymbol{x}$ into a particular vector $\boldsymbol{z}_i$. The "residual" vector $\boldsymbol{e} = \boldsymbol{x} - \boldsymbol{z}_i$ is then quantized using a B_2-bit quantizer. (The rotation $\boldsymbol{A}_i$ is used to "realign" all the residual vectors to reduce the distortion.) The quantized value of $\boldsymbol{x}$ is then given as the sum of the two code vectors, as shown in the figure.

code vector by searching the space in a systematic manner; at each step, one gets closer and closer to the desired code vector. The role played by the intermediate vectors $\boldsymbol{v}$ in Fig. 19 is simply to guide the search; the desired code vector is found at the end of the search. In cascaded VQ, each stage is a complete VQ search of the space which yields a separate code vector; the final quantized value of the input $\boldsymbol{x}$ is then the *sum* of the code vectors found at each of the stages. The input to each stage is the difference (residual) between the chosen code vector and the input for the previous stage.

The computational and storage costs for a two-stage cascaded vector quantizer (assuming *K*-means for each stage) are simply

$$\mathscr{C} = N(L_1 + L_2) \qquad \text{(cascade)} \qquad (117)$$

$$\mathscr{M} = N(L_1 + L_2) \qquad (118)$$

instead of NL_1L_2 for a single-stage full-search quantization. For the example of 10-bit quantization of 14 LARs, the cost of two-stage cascaded VQ with $B_1 = B_2 = 5$ bits is $14(2^5 + 2^5) = 896$ multiply-adds as compared to 14336 multiply-adds for single-stage VQ. Also, storage cost is reduced in the same proportion. What we are sacrificing for this significant reduction in cost is a loss in performance, as we shall see below.

We use as our example 14-LAR quantization in speech coding. Fig. 23 shows the mse distortion as a function of bit rate for several VQ schemes. Plot (*d*) shows the performance of a nonuniform binary quantizer. Plot (*a*) shows the performance of a cascaded vector quantizer with each stage quantizing to only 1 bit. Clearly, this constitutes the most extreme form of cascaded VQ, with a maximum reduction in cost. As can be seen from plots (*a*) and (*d*), the performance of this cascaded quantizer is significantly worse than the single-stage binary vector quantizer for each bit rate.

What causes this vast reduction in performance for the cascaded quantizer? Recall that during the training phase, after each stage, the residuals are pooled together to form the input to the next stage. If all the clusters in the first stage possess the same relative pdf structure within the cluster, the sets of residuals from the different clusters will have the same pdfs and pooling them together should not reduce performance. However, in general, the residual pdfs from the different clusters will be different and pooling them together will result in a single pdf that will lose many

Exhibit Y Page 483

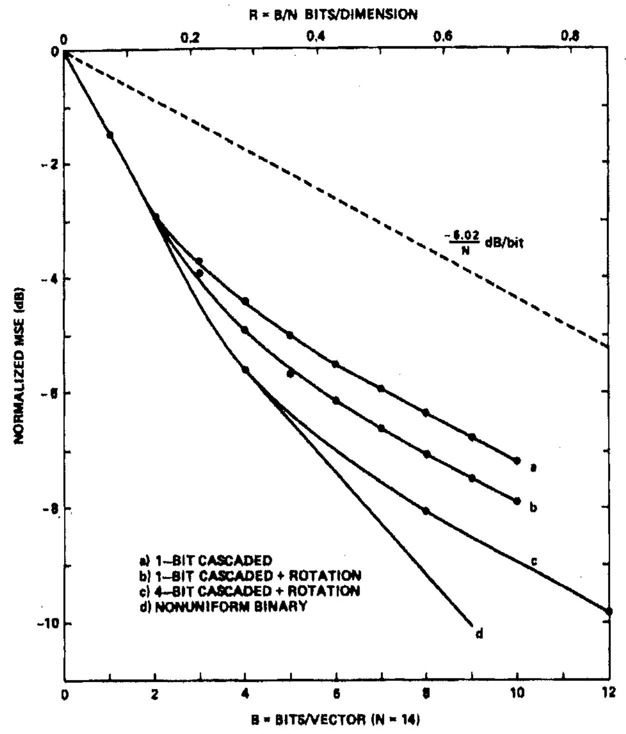

Fig. 23. A comparison of the mse in quantizing LARs using four vector quantizers: (*a*) multistage (10 stages shown) 1-bit cascaded vector quantization with no rotation between stages; (*b*) same as (*a*) but with rotation between stages; (*c*) 3-stage, 4-bit cascaded vector quantization with rotation (each stage used a nonuniform binary search); and (*d*) one-stage vector quantization with a nonuniform binary search.

of the dependencies that existed in the initial clusters. As a result, the distortion-rate curve will reach its asymptotic behavior of $-6.02/N$ decibels per bit at a higher distortion level. We see from plot (*a*) that it reaches its asymptotic behavior at 7 bits and a distortion of -6 dB.

In an effort to forestall the loss of dependence among residual components, we have inserted a vector rotation $\boldsymbol{A}_i$ after each stage [112], as shown in Fig. 22. The idea is to attempt to at least preserve some of the linear dependencies that exist among residual components before they are pooled together. The matrix $\boldsymbol{A}_i$ is set equal to the eigenmatrix corresponding to the covariance matrix of the residuals from the ith cluster. In this manner, the residuals from the different clusters are rotated so that their covariance matrices will have the same structure. Therefore, the covariance of $\boldsymbol{u} = \boldsymbol{A}_i\boldsymbol{e}$ resulting from each cluster will have the same structure. So, pooling the vectors $\boldsymbol{u}$ from the different clusters together will keep the same covariance structure. Fig. 23, plot (*b*), shows the performance of a 1-bit cascaded quantizer with vector rotation. Note how the asymptotic behavior begins at a lower distortion level. Plot (*b*) shows about a 1.8-bit advantage over plot (*a*) for the same distortion at the higher rates. This advantage with using rotation would be expected to be smaller if each stage in the cascade employed a larger number of bits.

Plot (*c*) in Fig. 23 shows the performance of a 4-bit, 3-stage, cascaded vector quantizer with eigenvector rotation after each stage. The 4-bit VQ in each stage utilized a binary tree search. (That is why plots (*c*) and (*d*) coincide up till $B = 4$ bits.) The asymptotic behavior is reached after the second stage at a distortion level of about -8 dB. The sharp departure of plot (*c*) from plot (*d*) at 4 bits is a clear indicator of the devastating effect that pooling residuals has, even if eigenvector rotation is utilized. The nonlinear dependencies among LARs are so important and they are virtually destroyed by the pooling of residuals. In fact, the performance for $B > 4$ bits in plot (*c*) is comparable to the performance of a scalar quantizer at that point. So, there may not be any need to go through the expense of additional vector quantizers after the first stage; a scalar quantizer (with rotation) could effectively replace all stages beyond the first.

The improvement in performance afforded by the vector rotation between the two stages of the cascade is obtained at a cost of additional computation and storage. The additional cost can be seen to be

$$\mathscr{C} = 2N^2 \quad \text{(rotation)} \tag{119}$$

$$\mathscr{M} = N^2 L_1. \tag{120}$$

The storage cost is especially large relative to that for the cascade in (118). Because of these additional costs, the inclusion of the rotation becomes less attractive as a method for improving performance.

It appears, therefore, that the maximum that one can effectively utilize in practice is about two stages of cascaded quantization. The first stage would employ VQ and should be allocated the maximum number of bits that one can afford in terms of computations and storage. The second stage would then perform scalar quantization on the residuals from the first stage. However, if the bit rate for the second stage is less than 1 bit/parameter ($R < 1$), then VQ would be advisable for the second stage as well. Also, if the distortion measure is not conveniently defined in the scalar case (such as the Itakura–Saito distortion), then VQ would have to be used for all stages.

Storage Reduction: Cascaded quantization was designed chiefly for the purpose of reducing storage (though computations are reduced as well), for in tree-searched VQ we already have a very effective method for reducing computations with relatively little loss in performance and a modest increase in storage. In fact, with binary search, for example, one could claim with some justification that, for many applications, computational cost is no longer a major limitation in VQ. If storage cannot be reduced without a concomitant major reduction in performance, then one would have to conclude that *storage cost is ultimately the major limitation in VQ*, for it would prevent us from taking advantage of the power of very large codebooks. With binary search, many would not hesitate to use a 30-bit codebook, but the required storage would tax the addressing capabilities of most of today's computers. Furthermore, the amount of training data needed to design such a codebook can be a serious limitation as well. While much research has been devoted to reducing computational costs, relatively little effort has been expended specifically in reducing storage costs. And little or no work appears to have been done on reducing storage at the expense of increased computation instead of decreased performance. Below, we present another approach to reducing storage costs, namely, product codes. This approach also reduces computations and results in reduced performance, but it could serve as a more attractive solution for many applications.

Exhibit Y Page 484

C. *Product Codes*

Assume that we structure our parameter space such that two or more component vectors, each with its own codebook, jointly represent all points in the space. The Cartesian product of the component codebooks is known as a *product code*. If, for example, we structure parameter space in terms of two component vectors of dimensions N_1 and N_2, and let the corresponding codebooks be of sizes L_1 and L_2, respectively, then the product code of dimension N and size L, where

$$N = N_1 + N_2 \qquad L = L_1 L_2 \tag{121}$$

will need memory storage equal to the sum of the storage requirements of the two component codebooks

$$\mathcal{M} = N_1 L_1 + N_2 L_2 \qquad \text{(product code)}. \tag{122}$$

Compared to a storage cost of NL for an N-dimensional codebook of size L, we see that the storage costs for a product code can be substantially lower.[12]

While storage costs are reduced through the use of a product code, computational costs are reduced only by using certain types of distortion measures or by making simplifying assumptions and approximations. To illustrate the problem, assume we want to use a product code for LARs, i.e., have a separate codebook for each LAR (see Example 4), but instead of the mse use the Itakura–Saito distortion[13] to find the nearest code vector. Then the only way to quantize an input LAR vector would be to perform a *full search* of the product code for the nearest code vector, because unlike the mse, the Itakura–Saito distortion is not separable into distortions on the individual LARs. The result is that there is no reduction in computation relative to the full-search codebook designed by K-means.

There are at least two conditions under which computational costs for quantization with a product code can be reduced in a comparable manner to the storage cost reduction in (122): *independent* quantization and *sequential* quantization. Under independent quantization, the component vectors are quantized independently using their respective codebooks. If the distortion measure is *separable* into independent distortions on the individual vectors, then independent quantization will, in fact, give the nearest product code vector. The mse is one example of a distortion measure that is separable in this fashion.

Under sequential quantization, one of the component vectors is quantized first, then the quantized value of that vector is used in the quantization of a second component vector, and so on. (Clearly, sequential quantization includes independent quantization as a special case.) The LPC "gain-shape" product code [18], [114], using a different definition of the Itakura–Saito distortion, is one example where the nearest product code vector can be obtained by quantizing the LPC parameters (shape) first, then using the quantized LPC parameters to quantize the gain such that the overall distortion is minimized.

In practice, sequential quantization is often used as an approximate suboptimal solution so as to minimize computations. One example is in a residual-excited speech coder, where the product code is the product of the spectrum codebook and the residual codebook. In one of the known implementations, the spectral envelope is quantized first, then the residual is computed using the values of the quantized spectrum, and finally the residual is quantized. This sequential quantization procedure does not minimize any known overall distortion on the reconstructed speech waveform. (See Section VII-B for a different example of a residual-based product code.)

Thus far we have considered only the quantization process in using a product code. Another important consideration is the design of the component codebooks in the first place. If an optimal (minimum-distortion) product code is desired, then, in general, the component codebooks cannot be designed independently unless the component vectors are statistically independent *and* the distortion measure is separable. In practice, independent or sequential quantization is employed in the design process to reduce computations even if the final product code is suboptimal.

In structuring our parameter space into two or more component vectors, a question arises as how best to define the component vectors and allocate the bits among the respective codebooks. Ideally, one would want to attempt to *structure parameter space such that the component vectors are statistically independent*, as much as possible. For if the component vectors were independent, then the only reduction in performance would be owing to the reduced dimensionality of the vectors. Otherwise, we would expect an additional reduction in performance in proportion to the dependence between the component vectors. While it is possible to remove linear dependencies by proper vector rotations, as we saw in Section IV, removing nonlinear dependencies is a difficult task, in general.

To maximize performance, it should be clear that, as a general design criterion, the number of component codebooks should be as low as possible and the size of each codebook as large as can be afforded in storage. The problem of how to choose the sizes of the component codebooks is a bit-allocation problem similar to the one discussed in Section IV-A.

In comparing a two-vector product code and two-stage cascaded quantization, we note that if L_1 and L_2 in (122) are equal to L_1 and L_2 in (118), then the cascade storage would be greater than the product code storage. But a fair comparison between the performance of the two methods would be to design the component codebooks in each case such that the memory costs are equal, then compare the resulting distortion. From (118) and (122), that would mean that the sizes of the component codebooks in each case would have to be different. Which of the two methods would be expected to perform better in practice depends on the particular problem and the structure of the statistical dependencies.

Below we mention briefly two specific methods of using product codes for coding LPC spectra, which can be used as alternatives to cascaded quantization. In Section VII-B we present an effective method for utilizing product codes in speech waveform coding.

[12]It is possible to structure parameter space such that $N_1 + N_2$ is larger than the dimension of the original space. (One example is structuring a speech waveform into two vectors: one representing the spectral envelope and another representing the residual.) In that case one has to compare (122) with $N'L$, where N' is the dimension of the original space.

[13]Computing the Itakura–Saito distortion between two LAR vectors would necessitate converting the LAR vectors to predictor coefficients first and then using (23).

Exhibit Y Page 485

Split-Vector Codes: In this method, a vector (such as a LAR vector) is split into two or more subvectors and each subvector is coded independently using a codebook designed for that part of the vector. Consider again the coding of LAR vectors in Example 4 above using a total of 15 bits/vector for 14 LARs, and assume we are interested in comparing a product code with a two-stage cascade for a given storage cost. In particular, assume that the storage cost is the minimum possible for the cascade case, which, from (118) and (121), can be seen to occur for $L_1 = L_2 = 2^{7.5}$, i.e., 7.5 bits for each of the two codebooks, so that $\mathcal{M} = 5068$ for a 14-dimensional vector. A product code can be designed using the bit-allocation table in Table 3 so that the storage is similar. A good solution would be to split the 14-LAR vector into two subvectors and use two codebooks: a 4-dimensional, 10-bit codebook corresponding to parameters 1–4, and a 10-dimensional, 5-bit codebook corresponding to parameters 5–14. The resulting storage cost from (122) is 4416. One would then compare the distortions for the two separate cases. (We note from Table 3 that a similar product code could be designed for the rotated parameters, which may perform better than a product code on the LARs.) To our knowledge, the suggestion to use split-vector codes for LPC parameters is novel. As yet no comparisons have been made with cascaded quantization.

Split-Band Codes: In this method, the spectral envelope is split into two or more bands and each band is vector quantized separately. Copperi and Sereno [24] designed a residual-excited coder in which the signal was split into two equal frequency bands, and separate LPC model parameters for each band were vector quantized separately. Using the split-band approach allows the designer to choose lower order spectral models and to make use of the fact that human perception is generally less sensitive to high-frequency distortions than to low-frequency distortions. This method is reminiscent of sub-band coding [74] except that here we attempt to keep the number of bands to a minimum (preferably not more than two) and the spectrum in each band is modeled explicitly.

D. Random Codebooks

While it is important to reduce the costs associated with the vector *quantization* process, there are times when reducing the costs in the *training* process is of interest. One simple method to design a codebook with essentially no computational training cost is to choose the code vectors at random from a given set of training data. We call the resulting codebook a *random codebook*.

At first glance it would appear that a random codebook would not perform well for quantization purposes. However, results on the asymptotic (large L and N) optimality of the expected performance of random quantizers [115] and the use of random codebooks in proving the basic rate-distortion theorems is a strong motivation for the use of such codebooks. While the use of a random codebook is a reasonable choice for large L and N, it is nevertheless used in practice when these conditions are not satisfied. Random codebook selection and performance under these non-asymptotic conditions is a research area of current interest.

In Fig. 24 we show the relative performance of random and nonuniform binary codebooks for the quantization of 14-LAR vectors using the mse distortion measure. The training data consisted of 15 min of speech (1 min from each of 15 speakers), which comprised approximately 90 000 training vectors The random codebook was obtained by selecting a set of vectors from the training data at equal intervals. Thus if a 100-level random codebook was desired, we chose every 900th vector from the training data. The test data consisted of sentences from five male speakers not used in the training. Note how at low bit rates the random codebook performs significantly worse than the binary codebook. However, the two curves begin to converge as the bit rate increases. We would expect the curves to get even closer at higher rates, but asymptotically they would become parallel to each other with some separation in bits. The difference in performance of about 1.3 bits at 10-bit quantization is smaller than one might expect.

We must bear in mind that, although a random codebook is simple to design, it is a full-search codebook and so

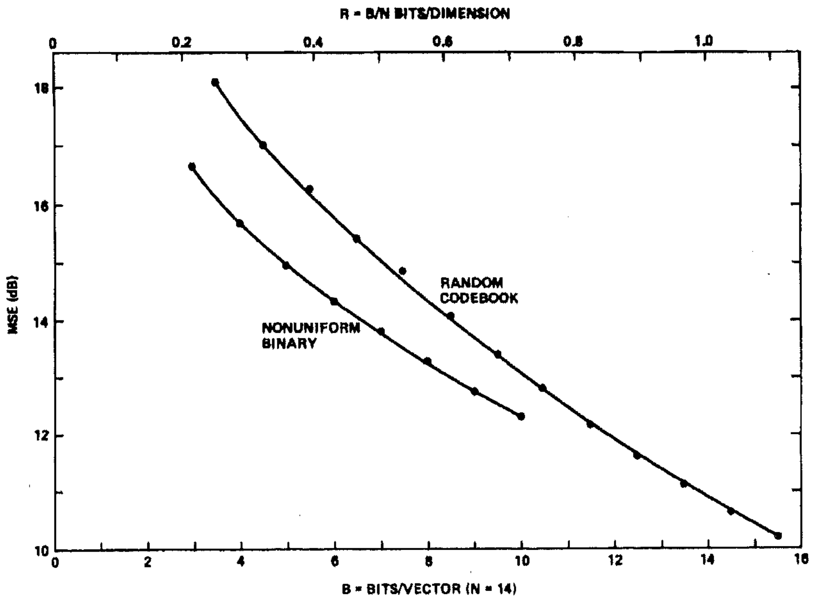

Fig. 24. The mse in quantizing LARs using a random codebook, compared with a nonuniform binary-structured codebook.

requires the same computation and storage. We find random codebooks to be useful in exploring how the VQ distortion decreases as a function of bit rate for higher data rates. In Section VI we mention another useful application of random codebooks.

E. Training and Testing

An important aspect of the design of any codebook is the training procedure used to populate the codebook, i.e., to fill its entries. In the K-means algorithm, the training procedure was given in Section II-C, except for a crucial initialization step where one chooses an initial set of code vectors. Below we describe initialization procedures that we have found useful for the VQ of LPC parameters in speech coding. The testing of the resulting codebook and its robustness are discussed next.

1) Training:

Binary clustering: In binary search we need at each stage a method to initialize the partitioning of space into two regions. Ideally, what we would like to do is to draw a hyperplane through the mean of the training data and orthogonal to the largest eigenvector (we assume a mse distortion). Such a hyperplane would divide space into two initial regions from which to start the iterative training procedure. For data where one dimension has a larger variance than the other dimensions, we approximate the eigenvector direction by the dimension with the largest variance. The centroids of the two clusters separated by the orthogonal hyperplane are used as the initial two code vectors. The K-means algorithm is then continued with $K = 2$ to determine the final code vectors. (We have found 5 to 10 iterations sufficient to obtain convergence.) After the first partition is completed, each of the subregions is divided into two using the same procedure, etc.

K-means initialization: Because the K-means is not guaranteed to result in a codebook that is globally optimum, it is often suggested that one repeat the algorithm with a number of different initial sets of code vectors. The codebook that results in the minimum distortion is then chosen for actual use. The pattern-recognition and VQ literatures contain a number of initialization methods for K-means [30], [56], [64], [99].

Because the performance of binary search is close to the optimal K-means (see Fig. 21), we have found the codebook generated by nonuniform binary search as a good initial codebook to start the K-means iteration. This initialization method has produced better results than using a random initial codebook.

2) Testing: After a codebook is designed with a given set of training data, it is important to test the performance of the codebook on independent data that were not used in the training. Testing only on the training data always presents an overly optimistic view of how the codebook will perform on operational data.

Fig. 25 shows the mse distortion for a 6-bit codebook ($L = 64$ levels) of 14-LAR vectors. The mse is plotted against the number of vectors in the training data. The training data were taken from the speech of 15 male speakers. The resulting codebook for each set of training data was tested on the same training data and on an independent set of test data taken from 5 male speakers who were not used in the training. With increased training, the convergence of the two plots increases. However, the convergence is very slow beyond 20 training vectors per codebook level (i.e., per code vector). As a rule of thumb, using 50 training data per code vector is considered sufficient for most applications. If sufficient data are not available or if the computation or storage becomes excessive, then using as few as 10 training data per code vector may be adequate. The remaining difference between the two plots in Fig. 25 is due to the fact that the test speakers were different from the training speakers. This issue relates to codebook *robustness*, which is discussed below.

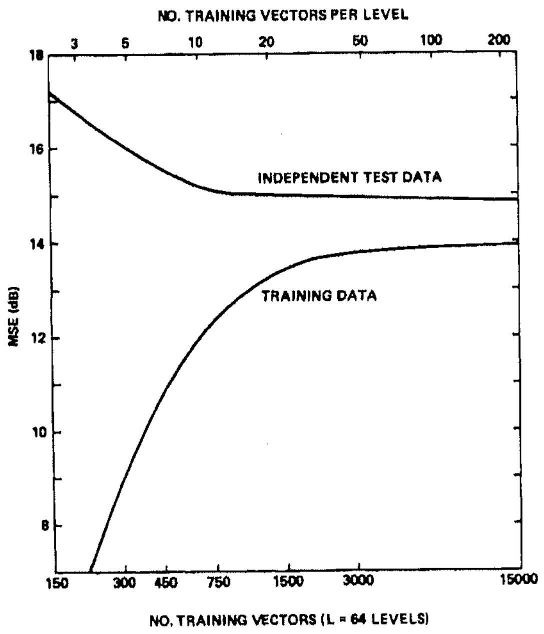

Fig. 25. The mse in quantizing LARs for a 6-bit codebook ($L = 64$ levels) as a function of the number of training data vectors, when tested on the training data and on an independent data set. The smaller the gap between the two curves, the more robust is the codebook.

In general, convergence of training and test performance curves with increased training set size is not guaranteed for arbitrary sources. One theoretical result of interest here is that, for reasonable distortion measures, such convergence is guaranteed for a class of sources known as *asymptotically mean stationary* [59], in which the stationarity condition is relaxed. Speech from a single speaker under a fixed set of environmental conditions, for example, exhibits short-term and long-term stationary properties and is well modeled as an asymptotically mean stationary source. However, the model may not apply as well under more variable, nonstationary conditions.

3) Codebook Robustness: Codebook robustness refers to the resistance of a codebook to degraded performance when tested on data whose distribution is different from that of the training data. Under operational conditions, one cannot usually predict all of the situations under which a quantizer will be used, and so the distribution of the operational data will in general be different from that of the training data. (Fischer and Dicharry [36] have studied the effects of designing codebooks for particular memoryless sources and testing on others with different pdfs.) We differentiate two major types of variabilities that impact the design and operational performance of a codebook: input signal variability and digital transmission channel errors. We discuss each of these below and compare VQ to scalar quantization.

For speech, signal variability can be classified further into

Exhibit 4 Page 487

speaker variability and environmental variability. Intraspeaker variability is that due to changes in each speaker's voice: normal everyday changes, changes due to different health conditions, and changes brought about by various emotional states. Interspeaker variability refers to voice differences across speakers. Environmental variability refers to the level and type of background noise that surrounds the speaker (e.g., office environment, outdoors, helicopter, etc.) and signal pickup characteristics, including the types of microphones and transmission facilities (e.g., telephone, radio). The performance of any codebook would be expected to deteriorate if used with types of signals for which it had not been designed. Thus if a codebook is designed (trained) for the voice of one speaker, it would not be expected to perform as well for other speakers. Likewise, a codebook trained in an office environment would not be expected to perform as well in a helicopter environment, for example. It is clear that, for optimal performance, one should train the codebook using data that are representative of what the VQ system will meet in actual operation.

The two plots of Fig. 25 would have converged even more had the test data been taken from the same speakers used in the training. The remaining difference of about 1 dB between training and test data in Fig. 25 is significant. Increasing the amount of training data from the same 15 male speakers will not reduce the difference appreciably, it will simply render the codebook better at quantizing the data from those same 15 speakers. If what is desired is the best possible speaker-independent performance, then the gap between training and test shown in Fig. 25 can be bridged further only by increasing the number of speakers in the training rather than by increasing the amount of speech from each speaker. If the final system is to be used with female speech as well, then including female speakers in the training becomes important.

For a given bit rate, a speaker-independent codebook cannot possibly perform as well as a speaker-dependent codebook. (We have seen differences in performance between 1 to 2 bits per vector for LAR quantization around 10 bits.) One interesting possibility for maximizing performance of a VQ system is to design a speaker-independent codebook initially. Then, as the system is used, it adapts to the speech of the new speaker. Such a system would have the extra advantage of also automatically adapting to the acoustic environment of the speaker. Codebook adaptation means that the code vectors will change in time, which necessitates transmitting the new code vectors to the receiver as well. Two such systems were designed by Paul [100] for real-time speech coding at 800 bits/s. In one system, a full-search codebook minimizes the maximum distortion between the input vectors and the code vectors. When the quantization distortion exceeds a certain threshold, the input vector is taken as a new code vector and the least used code vector up to that point is discarded. The new code vector is then transmitted to the receiver.

Scalar quantizers are generally more robust to signal variability than vector quantizers. For the same number of bits per vector, a scalar quantizer usually has fewer output levels than the vector quantizer. (For example, a 10-bit vector quantizer has 1024 levels, while 10 1-bit scalar quantizers have a total of only 20 output levels.) Therefore, given the same amount of training data, the scalar quantizer will have more training data per level and, hence, the resulting quantizer will be more robust. Another more important reason for the greater robustness of the scalar quantizer is exactly the same reason that renders its performance lower than the vector quantizer. The scalar quantizer, by "averaging" or "projecting" the data on one dimension at a time, is less able to model the detailed multidimensional dependencies in the data and, therefore, will be less susceptible to changes in those detailed dependencies. In other words, the one-dimensional pdfs are expected to change less between training and test than the corresponding multidimensional pdfs. Because vector quantizers are not in general as robust as scalar quantizers, it is very important to test vector quantizers extensively on independent data. Also, whenever changes in the algorithm are made, a new set of test data should be used. Otherwise, there might be a tendency to unconsciously change the algorithm to improve performance on the same test set, effectively rendering the test set part of the training data.

Transmission channel errors present a different type of problem to system robustness. Channel errors translate directly into distortions in the output; the higher the error rate, the greater the distortion. Again, in general, VQ systems tend to be less robust to random channel errors than scalar quantizers. (Burst errors may affect both types of quantizers equally.) A simple example will illustrate the difference. Let us take the example of the 10-bit vector quantizer versus the ten 1-bit scalar quantizers, and assume a channel error rate of 1 percent. Then, in 100 bits there will be 1 bit that is in error, on the average. In the scalar quantizer, that 1-bit error causes one value in one dimension to be wrong, while in the vector quantizer, the same 1-bit error causes a whole 10-bit vector to be wrong, which would result in larger distortions on the average.

By using extra transmission bits, standard error-correcting techniques can be employed to help correct for channel errors [103]. The use of such techniques is strongly recommended for high error rates (> 0.1-percent errors). In an unstructured codebook, all bits in the code vector would have to be protected in some fashion, while in structured codebooks (binary-structured or scalar), one can choose to protect only those bits that are most important to the fidelity of the output.

Channel errors can play havoc with any system that employs variable-length coding (such as entropy coding). A single bit error affects not only the one value in which it occurs but can also affect many succeeding values. The error causes the receiver to lose synchronization with the input and succeeding bits may be erroneously decoded. The use of "self-synchronizing codes" [105] reduces the magnitude of the problem significantly. However, independent synchronization schemes, involving the transmission of additional synchronization bits, would still be needed to avoid loss of synchronization.

It should be clear by now that simply optimizing system performance over a set of training data is only one aspect of choosing the best system for some application. Overall robustness of the system to the operational environment must be included in system performance assessment.

VI. Time-Dependent Vector Quantization

Thus far, we have assumed that each input vector is quantized independently of any other input vector. While we have considered dependencies among vector components (in short-term spectral space, for example), we have

Exhibit Y Page 488

not yet taken advantage of longer term vector dependencies in time. Certainly, any further dependencies we can uncover should allow us to transmit the data at even lower rates for a given average distortion. Speech, of course, is a time-varying process that is rich in time-related dependencies. The statistical structures of the spectral sequences for each sound and of the sequences of sounds in each utterance are fertile sources of dependency. In this section, we survey briefly several methods in which time dependencies (beyond a single frame) are exploited to reduce the bit rate. We begin first with frame transmission schemes that select the frames to be transmitted judiciously, followed by time-dependent VQ methods. Most of the methods described below incorporate additional delays.

A. *Selective Frame Transmission*

Perhaps the simplest, and yet effective, method for selective frame transmission is known as *frame repeat* or *frame fill* [93], [101], [102]. In this method, only every other frame is transmitted; for the intervening frames, a 1-bit code is transmitted which specifies whether the missing frame should be replaced by the previous frame or the following frame. Which frame to repeat is determined by a nearest neighbor rule. A variation on this method is to send 1 or 2 bits which specify values for the missing frame which are a linear interpolation of the transmitted frames [119].

Another simple and very effective method for selective frame transmission is *variable-frame-rate* transmission [100], [119], [132]. In this method, information is transmitted only when the properties of the speech signal have changed sufficiently (in some predetermined sense) since the preceding transmission. The parameters of the untransmitted frames are regenerated at the receiver through linear interpolation between the parameters of the two adjacent transmitted frames. Thus parameter transmissions occur more frequently when speech characteristics are changing rapidly, as in transitions between sounds, and less frequently when speech characteristics are relatively constant, as in steady-state sounds.

B. *Segment Quantization*

Variable-frame-rate VQ of spectral parameters is a useful procedure for coding at rates as low as 400 bits/s. For data rates of 300 bits/s or lower, the linear interpolation assumption breaks down and the method is no longer adequate in maintaining reasonable intelligibility. At these lower rates, the method of choice is to perform VQ on a sequence of frames comprising a larger segment of speech. The method is known as *segment quantization* [111], [137], in contrast with frame quantization where each frame is quantized separately. VQ is now employed over a larger vector $\boldsymbol{X} = [\boldsymbol{x}(1)^T\boldsymbol{x}(2)^T \cdots \boldsymbol{x}(J)^T]^T$ comprising the elements of J consecutive frame vectors. In this manner, the dependence between consecutive frames is included implicitly in the larger vectors.[14] The duration J of the segment can be either fixed or variable. Clearly, a fixed J results in a simpler quantization problem. However, utilizing variable-duration segments has led to better speech quality.

Segment quantization is a good example where a random codebook has been especially useful. Because of the relatively high-dimensional space ($10 \times 14 = 140$ if $J = 10$), one can argue that a random codebook should give good results. The overriding reason for using a random codebook, however, comes from perceptual considerations. We have found that the clustering and centroid computations in the K-means algorithm produce code vectors (segments) that sound muffled upon listening. So, even if a random codebook produces a larger mse distortion than the corresponding K-means codebook, the synthesized speech from the random codebook yields higher quality and intelligibility. This is yet another instance where an objective measure of distortion is not a good measure of perceptual distance. In the experiments performed thus far, a 13-bit codebook is used ($L = 8192$ segments). Because a random codebook is not structured, quantization of the speech input requires a substantial amount of computation.

Segment quantization is truly in the spirit of VQ theory. For, in principle, one should be able to approach the performance of the optimal coding system by simply using longer blocks for quantization. Unfortunately, if signal fidelity is to be maintained, exponentially larger codebooks will need to be used, which very quickly becomes impractical. An alternative solution is to include time dependence explicitly in the modeling (as opposed to implicitly, as in segment quantization). In this manner, the complexity of the vector quantizer can remain manageable. We have already described two such methods above, frame repeat and variable-frame-rate transmission. Below, we describe other models that adapt to the changing properties of the signal.

C. *Adaptive VQ*

Adaptive VQ implies a change in the codebook as a function of time. There are two types of adaptive vector quantizers: forward-adaptive and backward-adaptive. In forward-adaptive schemes, the system examines *future* data and decides whether a change in the codebook is necessary or not, and transmits to the receiver the desired changes using additional bits. Typically, such schemes incorporate some delay to allow for the inspection of upcoming data. Backward-adaptive schemes examine only *past* transmitted data and change the codebook accordingly. Since past information is available at both the transmitter and the receiver, the necessary codebook updating computation can be performed at both ends simultaneously, thus obviating the need for transmitting additional bits. Backward-adaptive systems have the added advantage of having no extra delay, but are quite susceptible to degraded performance under channel errors. If in forward-adaptive systems the additional bits are not counted, then such systems will result in lower quantization distortion than backward-adaptive systems. However, for an overall fixed data rate, which of the two types of systems performs better depends on the relative number of additional bits to the overall data rate. The principles of forward and backward adaptation are similar to those used in scalar quantization [74].

We should mention at this point that VQ, by its very nature, implies some delay to allow for the input vectors to be specified. Even a backward-adaptive vector quantizer

[14]Segment quantization has also been termed *matrix quantization* [137]. We prefer to use the former term because matrix quantization could be taken to imply another method beyond scalar and vector quantization, while in reality it is simply VQ applied over a longer vector that comprises a larger segment of speech.

Exhibit Y Page 489

will have this inherent delay. A forward-adaptive vector quantizer may have additional delays above and beyond the delays inherent in specifying the vectors in the first place.

An example of a forward-adaptive system is Paul's adaptive VQ system [100] described above. Most adaptive VQ schemes to date, however, are backward-adaptive and most such schemes are *feedback* VQ systems [56]. Two major classes of feedback VQ systems are *vector predictive* systems and *finite-state VQ* systems. Vector predictive systems [25] are a generalization of scalar predictive systems to the vector case and, thus far, they have been used for medium-rate coding of speech waveforms.

In finite-state VQ, we have a finite-state network with transitions among states and we associate with each state a codebook. If one is in a particular state, then the codebook associated with that state is used to quantize the input vector at that time. Depending on which code vector was used, a transition is made to another state (which could also be the same state). The codebook at that state is now used to quantize the next input vector, and so on. The union of all codebooks from all states is usually very large. By subsetting the larger codebook into smaller codebooks at each state, a smaller number of bits is used to quantize each input vector, so that the average bit rate is reduced from the case where the large codebook is used for every input vector. If the time-dependent structure of the data is extensive, substantial reduction in bit rate without loss in fidelity can be accomplished. Finite-state VQ schemes have been employed with segment quantization [113], frame quantization [32], [110], and waveform coding [42], [62].

VII. Speech Waveform Coding

Our understanding of the vector quantization of speech *spectral parameters* has advanced and its usefulness and practicability have been demonstrated to the point where real-time hardware implementation of very-low-rate speech coders is becoming a reality. The application of VQ to speech *waveform* coding is a more recent but intense research activity, and lends itself well to the development of a large number of techniques and variations. We believe it will be some time before the relative merits of the various techniques are worked out and fully appreciated. While some waveform VQ techniques are becoming practical now, especially for coding at about 16 kbits/s, other techniques for low-rate coding require significant computational resources that are not widely available. With additional research effort, the role of waveform VQ in speech coding will become clearer and the possibilities of achieving toll quality at low data rates (below 8 kbits/s) may become a reality.

Below, we give the reader a view of the difficulties inherent in waveform VQ and present a framework that we hope will help sharpen our understanding of the waveform VQ process. It is instructive first to take a brief look at the state of the art in the scalar quantization of speech waveforms, especially those methods that will have a bearing on our VQ discussion below.

A. *Scalar Waveform Quantization*

It is generally accepted that at 16 kbits/s, adaptive predictive coding (APC) and adaptive transform coding (ATC) result in speech that has approximately toll quality [74] (i.e., quality equal to a high-grade telephone line). In a particular configuration for both methods, one computes and transmits: 1) a representation of the short-term spectrum (typically in the form of LPC coefficients); 2) the short-term gain; 3) the pitch and parameters of a pitch filter; and 4) the prediction residual waveform.[15] The pitch filter is usually an all-pole filter with a maximum of three parameters; it is of the form $1/C(z)$, where

$$C(z) = 1 + c(\tau - 1)z^{-(\tau-1)} + c(\tau)z^{-\tau} + c(\tau + 1)z^{-(\tau+1)} \tag{123}$$

and τ is the estimated pitch period in samples. Pitch periods for adults vary from less than 3 ms for high-pitched females to over 12 ms for low-pitched males. (In certain system implementations, the inclusion of a pitch filter is found to be unnecessary or undesirable; see, for example [89], [133].) Parameter sets 1–3 above are computed and transmitted every frame as *side information* with the residual. At the receiver, the residual excites a filter of the form

$$H(z) = \frac{G}{A(z)C(z)} \tag{124}$$

to result in the output speech. The speech signal is usually sampled at either 6.67 or 8 kHz. At 6.67-kHz sampling, the residual is coded with 2 bits/sample, which leaves about 2.7 kbits/s to transmit the side information. At 8-kHz sampling, a 3-level quantizer is used to code the residual (or about 1.6 bits/sample), which leaves over 3 kbits/s to code the side information.

At 16 kbits/s, the short-term SNR achievable by APC and ATC is approximately equal to the gains predicted by rate-distortion theory. The SNR gain over simple quantization of the speech waveform itself is approximately equal to $-10 \log_{10}\gamma$, where γ is the ratio of the geometric mean to the arithmetic mean of the short-term speech spectrum as represented by $H(z)$ (see (76)). The short-term SNR, therefore, changes as the spectrum changes and, in speech, a typical range is 5–20 dB, with higher SNR values associated with vowel sounds and lower values with unvoiced sounds.

As the number of bits used to quantize the residual is reduced to about 1 bit/sample or less, speech quality deteriorates rapidly. A number of methods have been employed to improve the quality at transmission rates in the range 8–9.6 kbits/s. One technique known as *multipulse coding* [9] models the residual as a sequence of a relatively small number of pulses (smaller than the number of residual samples) whose amplitudes and time locations are optimized and transmitted each frame. Another class of coders, with a long history in speech coding, is known as *baseband-excited coders* [89] or *voice-excited coders* [38]. This is a frequency-selective method where only a low-pass band (known as the baseband) of the residual, for example, is transmitted. At the receiver, the high frequencies are obtained from the low frequencies by nonlinear high-frequency regeneration techniques. Though quite effective, these methods also break down as the bit rate goes much

[15]Most ATC systems operate directly on the speech waveform and not on the prediction residual [74]. However, if done appropriately, ATC on the prediction residual can produce similar results [16].

below 1 bit/sample. Vector quantization promises to carry speech coding with high quality into these lower rates.

B. Vector Waveform Quantization

Direct VQ of the speech waveform can be accomplished by simply dividing the waveform into consecutive blocks of N samples each and designing a codebook whose vectors are obtained by one of the clustering methods described earlier. At high data rates, such a method would give good results at relatively small values of N. However, as the bit rate is decreased, it becomes more and more important to increase the block length to minimize the distortion. As N increases, two problems arise. The first problem is the one we already know, and that is the exponential increase of needed resources. This problem can be dealt with partly by proper structuring of the codebook, as we shall see below. The second problem is less obvious; it concerns the concatenation of waveforms for synthesis. As we quantize each block independently to its closest code vector, we minimize the distortion over the whole block so that, for a long block, there is no guarantee that the end of one code vector will match the beginning of the next code vector in time. The result will be a discontinuity in the speech waveform, which may be heard as roughness in the speech. The problem is aggravated at low data rates since the choices for appropriate code vectors are relatively few. One solution to the concatenation problem is to quantize blocks that overlap by a small amount. Then, the quantized code vectors are overlapped and added with a decreasing weighting during the overlap region which goes to zero in both directions, so that the transition from one quantized block to the next is made smoother. This method is used effectively in ATC of the speech waveform to prevent waveform discontinuities from one block to the next [130]. Although the weighted-overlap method just mentioned would be expected to work well, it has the drawback that the overlap actually wastes bits since the bits for the overlapped regions are in effect transmitted twice. Another solution to the waveform discontinuity problem is to quantize the residual instead of the speech waveform itself, as will be described below. ATC of the residual has been used effectively in eliminating the need for overlap [16].

Much of the recent activity in applying VQ to the coding of speech waveforms has concentrated on the medium-rate range of 8–16 kbits/s (see, for example, [1], [24], [25], [34], [37], [42], [49], [50], [56], [63], [83], [109], [124]). In that range, scalar quantization methods already produce approximately toll quality speech at 16 kbits/s and communications quality [39] at 8–9.6 kbits/s. Of course, with the use of VQ, we have the potential of moving the toll quality boundary further down. An exposition of the various waveform VQ techniques under development is beyond the intended scope of this paper. Instead, we have chosen in this brief section to discuss a particular approach to speech coding at low rates (below 8 kbits/s), for it is at those rates that scalar methods tend to break down and VQ offers the potential of achieving toll quality. This approach was chosen because it forms a natural extension to existing well understood scalar speech coding techniques, and appears to have a potential for high-quality low-rate coding of speech.

The main problem in speech waveform coding is how to structure and design our codebook such that signal fidelity is maintained but computational and storage costs are reduced to manageable proportions. One relatively natural manner to structure the codebook is to factor out the short-term spectral and pitch dependencies and code them separately, in effect use a product code. By removing the short-term linear dependencies, one is left with a residual that is relatively white. Fig. 26 shows three waveforms: plot (a) is the speech waveform; plot (b) is the prediction residual after filtering the speech waveform with the spectral inverse filter $A(z)$; and plot (c) is the residual after filtering the speech waveform with the combined spectral and pitch inverse filter $A(z)C(z)$. (Note that in Fig. 26, waveform (b)

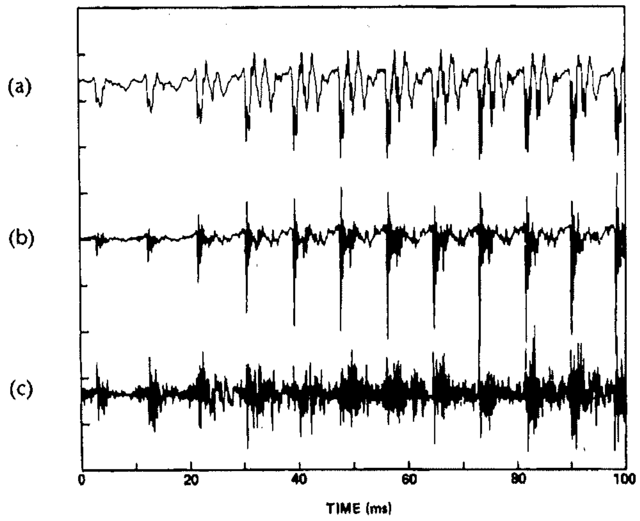

Fig. 26. (a) A speech waveform. (b) The prediction residual after filtering the speech with the all-zero filter $A(z)$. (The plot is amplified 10 dB relative to (a).) (c) The residual after filtering the speech with the combined filter $A(z)C(z)$, thus removing much of the pitch-related structure. (The plot is amplified 20 dB relative to (a).) (From Atal [7].)

was amplified 10 dB and waveform (c) 20 dB relative to the speech waveform.) The residual waveform (c) has most of its short-term linear dependencies removed; it certainly looks much more random than (a) or (b). The statistics of waveform (c) tend to be Gaussian [7]. Our codebook can now be structured in four parts corresponding to the spectral filter $A(z)$, the pitch filter $C(z)$, the gain G, and the residual (shown as waveform (c) in Fig. 26). The first three sets of parameters can be coded and transmitted as side information, exactly as in APC, except that here we can perform VQ on each set of parameters [2], [24] to reduce the bit rate. Also as in APC, the residual is *not quantized separately*, but rather as part of a synthesis procedure. If performed separately, time-domain quantization of the residual will result in no SNR gain over PCM coding of the waveform.

There is a compelling reason why the product code suggested above would be expected to perform well. We mentioned in Section V-C that a product code would be expected to yield good results if the component vectors are chosen to be independent. One could argue with some justification that the short-term spectral envelope, the pitch, and the whitened residual are to some extent independent,

Exhibit Y Page 494

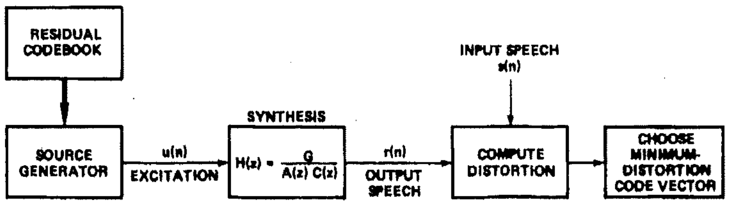

Fig. 27. A block diagram of a system for low-rate speech waveform coding which employs vector quantization. The spectral and pitch filter parameters are quantized and transmitted separately. The codebook shown is used to find the excitation that results in the minimum distortion. As shown in the figure, the excitation is white. If desired, $C(z)$ can be eliminated, in which case the excitation should have some pitch structure, as in the residual waveform (b) in Fig. 26.

so that the use of a product code in this case should reduce computation and storage at only a relatively small reduction in performance over a single large codebook obtained by K-means.

Fig. 27 shows a basic block diagram of the coding system. The first step is to compute and quantize the parameters of $A(z)$, $C(z)$, and G, then substitute the *quantized* values in the synthesis filter $H(z)$ in Fig. 27. Then, the source generator takes one code vector at a time from the residual codebook and sends the samples from the code vector sequentially in time as excitation to the filter $H(z)$. The distortion between the output speech and the input speech in the block to be coded is computed. The process is repeated for all code vectors in the residual codebook, each time comparing the output to the same block of input speech. The code vector that results in the minimum distortion is chosen for transmission. The whole process is then repeated for the next input block. Note that nowhere is there an explicit quantization of the actual prediction residual; the quantization takes place implicitly by searching all vectors in the residual codebook for the vector that minimizes the distortion in the output speech. (The distortion computation may benefit from perceptually based operations such as the use of spectral noise shaping [11], [88].) In product code terminology, the above is a sequential quantization procedure, whereby the side information is quantized first, then the quantized parameter values are used in a full-search of the residual codebook such that an overall distortion criterion is minimized.

The only remaining issues are to choose the block length N and the bit rate R, and to decide how to populate the residual codebook for a given N and R. There is evidence from experiments by Atal [7] that block lengths of about at least 4 ms are needed for good results. This is equivalent to $N = 32$ samples at 8-kHz sampling. At $R = 1$ bit/sample, the codebook contains 32×2^{32} residual samples or about one-half year of speech! At $R = 0.5$ bits/sample, the equivalent amount of speech is just over 4 min. However, the size of the codebook is still $L = 2^{16} = 65\,536$ vectors, which means that the process in Fig. 27 will need to be repeated that many times for each block of 32 samples, if a full search is desired. These illustrative numbers place in perspective the magnitude of the computational problem in VQ. To render the computations more manageable, one will need to either go to lower rates, use shorter blocks, or structure the residual codebook in some fashion.

Because the number of dimensions N here is relatively large, a random codebook would be expected to do well. This is especially so because the residual is relatively white. Therefore, one could populate the codebook from a random sample of residual vectors. Since the residual is relatively Gaussian and white, another possibility would be to simply use a random Gaussian number generator to populate the codebook. Both methods have been used by Atal and Schroeder with good results. In recent simulations on a Cray-1 computer, with $N = 40$ and $R = 0.25$ bits/sample for the residual (excitation), good speech quality was reported with no quantization of the side information [117]. It is important to note that even if the excitation consists of random numbers, each code vector will have a different detailed spectrum and time-domain structure. The quantization process chooses the particular code vector that minimizes the distortion between the original and reconstructed signals.

The use of a pitch filter $1/C(z)$ may not always be desirable. If a pitch filter is not used, then the residual codebook will need to contain wave shapes similar to waveform (b) in Fig. 26. Because of the need for the "pitch pulses," one can no longer use a random Gaussian source to populate the residual codebook. A codebook will have to be designed from a set of residual training data. One could, of course, use any of the other methods we mentioned in Section V. No one, to our knowledge, has attempted to use a more optimized residual codebook for large block lengths.

Tree and trellis coding: Another approach to structuring the codebook to reduce computations and storage is to have code vectors share some of their elements, i.e., some of the elements will have the same values. The two most well-known methods in this class are *tree coding* and *trellis coding* [74], [75], [135]. In tree coding, the vector elements are on a tree, while in trellis coding, the elements are on a trellis. Both types of coding permit the use of longer blocks because of the reduced computational requirements. However, the computational and memory savings are obtained at the cost of reduction in performance, with trellis coding achieving the greatest computational savings but at reduced performance relative to tree coding and to VQ. A number of studies have utilized tree and trellis coding of speech waveforms [6], [7], [34], [49], [54], [56], [74], [128], [136], but there has not been a systematic study comparing these two methods against VQ at lower data rates.

VIII. Conclusions

In the coding of signals at a given rate, it is always possible to reduce the distortion further by using vector quantization instead of scalar quantization. However, because of the substantial computational and memory costs associated with vector quantization, it is most advantageous to use it at low data rates of less than 1 bit/parameter or signal value to be transmitted. In addition, vector quantizers, on the whole, may not be as robust as scalar quantizers when used in an operational environment, especially under conditions of channel errors. However, at low data rates, vector quantization may still be the method of choice when all factors are considered.

Speech presents a fertile signal for the application of vector quantization. Speech parameters are rich with linear and nonlinear dependencies that are utilized effectively by vector quantizers to optimize performance. Much of the success in applying vector quantization to speech coding has been in the coding of spectral parameters for low rate coding of intelligible speech below 800 bits/s. Recent results in the vector quantization of speech waveforms point to exciting possibilities for high-quality speech coding at 8 kbits/s and less.

Acknowledgment

The authors wish to thank the following friends and colleagues who have read and commented on an earlier version of this paper: B. Atal, M. Ostendorf Dunham, A. Gersho, N. Jayant, B-H. Juang, R. Mercer, P. Noll, D. Paul, and Y. Shoham. Their comments have been very valuable in enhancing the accuracy of the presentation. The authors also thank L. Willson for her assistance in the preparation of the manuscript.

References

[1] H. Abut and S. A. Luse, "Vector quantizers for subband coded waveforms," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing* (San Diego, CA, Mar. 1984), paper 10.6.

[2] J-P. Adoul, J-L. Debray, and D. Dalle, "Spectral distance measure applied to the optimum design of DPCM coders with L predictors," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing* (Denver, CO, Apr. 1980), pp. 512–515.

[3] N. Ahmed, T. Natarajan, and K. R. Rao, "Discrete cosine transform," *IEEE Trans. Comput.*, vol. C-23, no. 1, pp. 90–93, Jan. 1974.

[4] V. R. Algazi, "Useful approximations to optimum quantization," *IEEE Trans. Commun. Technol.*, vol. COM-14, pp. 297–301, 1966.

[5] M. R. Anderberg, *Cluster Analysis for Applications.* New York, NY: Academic Press, 1973.

[6] J. B. Anderson and J. B. Bodie, "Tree encoding of speech," *IEEE Trans. Inform. Theory*, vol. IT-21, pp. 379–381, July 1975.

[7] B. S. Atal, "Predictive coding of speech at low bit rates," *IEEE Trans. Commun.*, vol. COM-30, no. 4, pp. 600–614, Apr. 1982.

[8] B. S. Atal and S. L. Hanauer, "Speech analysis and synthesis by linear prediction of the speech wave," *J. Acoust. Soc. Amer.*, vol. 50, no. 2, pp. 637–655, Aug. 1971.

[9] B. S. Atal and J. R. Remde, "A new model of LPC excitation for producing natural-sounding speech at low bit rates," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing* (Paris, France, May 1982), pp. 614–617.

[10] B. S. Atal and M. R. Schroeder, "Adaptive predictive coding of speech signals," *Bell Syst. Tech. J.*, vol. 49, pp. 1973–1986, Oct. 1970.

[11] ——, "Predictive coding a speech signals and subjective error criteria," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-27, no. 3, pp. 247–254, June 1979.

[12] T. P. Barnwell, III, "Correlation analysis of subjective and objective measures for speech quality," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing* (Denver, CO, Apr. 1980), pp. 706–709.

[13] R. E. Bellman, *Introduction to Matrix Analysis.* New York, NY: McGraw-Hill, 1960.

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

[15] T. Berger, "Minimum entropy quantizers and permutation codes," *IEEE Trans. Inform. Theory*, vol. IT-28, no. 2, pp. 149–157, Mar. 1982.

[16] M. Berouti and J. Makhoul, "An adaptive-transform baseband coder," in *Speech Communication Papers: 97th Meeting of the Acoustical Society of America*, J. J. Wolf and D. H. Klatt, Eds. (Cambridge, MA, June 1979), pp. 377–380.

[17] R. E. Blahut, "Computation of channel capacity and rate-distortion functions," *IEEE Trans. Inform. Theory*, vol. IT-18, pp. 460–473, July 1972.

[18] A. Buzo, A. H. Gray Jr., R. M. Gray, and J. D. Markel, "Speech coding based upon vector quantization," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-28, no. 5, pp. 562–574, Oct. 1980.

[19] J. W. S. Cassels, *An Introduction to the Geometry of Numbers.* Berlin, Germany: Springer-Verlag, 1959.

[20] D. Y. Cheng, A. Gersho, B. Ramamurthi, and Y. Shoham, "Fast search algorithms for vector quantization and pattern matching," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing* (San Diego, CA, Mar. 1984), paper 9.11.

[21] J. H. Conway and N. J. A. Sloane, "Voronoi regions of lattices, second moments of polytopes, and quantization," *IEEE Trans. Inform. Theory*, vol. IT-28, no. 2, pp. 211–226, Mar. 1982.

[22] ——, "Fast quantizing and decoding algorithms for lattice quantizers and codes," *IEEE Trans. Inform. Theory*, vol. IT-28, no. 2, pp. 227–232, Mar. 1982.

[23] ——, "A lower bound on the average error of vector quantizers," *IEEE Trans. Inform. Theory*, vol. IT-31, no. 1, pp. 106–109, Jan. 1985.

[24] M. Copperi and D. Sereno, "9.6 kbit/s piecewise LPC residual excited coder using multiple-stage vector quantization," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing* (San Diego, CA, Mar. 1984), paper 10.5.

[25] V. Cuperman and A. Gersho, "Vector predictive coding of speech at 16 kbits/s," *IEEE Trans. Commun.*, vol. COM-33, pp. 685–696, July 1985.

[26] L. D. Davisson and R. M. Gray, Eds., *Data Compression.* Stroudsburg, PA: Dowden Hutchinson & Ross, 1976.

[27] N. R. Dixon and T. B. Martin, Eds., *Automatic Speech and Speaker Recognition.* New York, NY: IEEE PRESS, 1979.

[28] J. L. Doob, *Stochastic Processes.* New York, NY: Wiley, 1953.

[29] E. Dubois, "The sampling and reconstruction of time-varying imagery with application in video systems," *Proc. IEEE*, vol. 73, no. 4, pp. 502–522, Apr. 1985.

[30] R. O. Duda and R. E. Hart, *Pattern Classification and Scene Analysis.* New York, NY: Wiley, 1973.

[31] H. Dudley, "Phonetic pattern recognition vocoder for narrow-band speech transmission," *J. Acoust. Soc. Amer.*, vol. 30, no. 8, pp. 733–739, Aug. 1958.

[32] M. O. Dunham and R. M. Gray, "An algorithm for design of labeled-transition finite-state vector quantizers," *IEEE Trans. Commun.*, vol. COM-33, no. 1, pp. 83–89, Jan. 1985.

[33] N. Farvardin and J. W. Modestino, "Optimum quantizer performance for a class of non-Gaussian memoryless sources," *IEEE Trans. Inform. Theory*, vol. IT-30, no. 3, pp. 485–497, May 1984.

[34] H. G. Fehn and P. Noll, "Multipath search coding of stationary signals with applications to speech," *IEEE Trans. Commun.*, vol. COM-30, no. 4, pp. 687–701, Apr. 1982.

[35] L. Fejes Toth, "Sur la representation d'une population infinie par un nombre fini d'elements," *Acta Math. Acad. Scient. Hung.*, vol. 10, pp. 299–304, 1959.

[36] T. R. Fischer and R. M. Dicharry, "Vector quantizer design for memoryless Gaussian, Gamma, and Laplacian sources," *IEEE Trans. Commun.*, vol. COM-32, no. 9, pp. 1065–1069, Sept. 1984.

[37] T. R. Fischer and K. T. Malone, "Contour vector quantization and waveform coding," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing* (Tampa, FL, Mar. 1985), pp. 1707–1710.

[38] J. L. Flanagan, *Speech Analysis Synthesis and Perception*, 2nd ed. New York: Academic Press, 1972.

[39] J. L. Flanagan, M. R. Schroeder, B. S. Atal, R. E. Crochiere, N. S. Jayant, and J. M. Tribolet, "Speech coding," *IEEE Trans. Commun.*, vol. COM-27, no. 4, pp. 710–737, Apr. 1979.

Exhibit 4 Page 493

[40] P. E. Fleischer, "Sufficient conditions for achieving minimum distortion in a quantizer," in *1964 IEEE Int. Conv. Rec.*, pt. 1, pp. 104–111, 1964.
[41] E. W. Forgy, "Cluster analysis of multivariate data: Efficiency vs. interpretability of classifications," *Biometrics*, vol. 21, p. 768, abstract, 1965.
[42] J. Foster, R. M. Gray, and M. O. Dunham, "Finite-state vector quantization for waveform coding," *IEEE Trans. Inform. Theory*, vol. IT-31, pp. 348–359, May 1985.
[43] B. Fox, "Discrete optimization via marginal analysis," *Manag. Sci.*, vol. 13, no. 3, pp. 210–216, Nov. 1966.
[44] J. H. Friedman, F. Bashett, and L. J. Shustek, "An algorithm for finding nearest neighbors," *IEEE Trans. Comput.*, vol. C-24, pp. 1000–1006, Oct. 1975.
[45] J. W. Fussell, "The Karhunen-Loève transform applied to the log area ratios of a linear predictive speech coder," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing* (Denver, CO, Apr. 1980), pp. 36–39.
[46] R. G. Gallager, *Information Theory and Reliable Communication.* New York, NY: Wiley, 1968.
[47] A. Gersho, "Asymptotically optimal block quantization," *IEEE Trans. Inform. Theory*, vol. IT-25, no. 4, pp. 373–380, July 1979.
[48] _____, "On the structure of vector quantizers," *IEEE Trans. Inform. Theory*, vol. IT-28, pp. 157–166, Mar. 1982.
[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] A. Gersho, T. Ramstad, and I. Versvik, "Fully vector-quantized subband coding with adaptive codebook allocation," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing* (San Diego, CA, Mar. 1984), paper 10.7.
[51] H. Gish and J. N. Pierce, "Asymptotically efficient quantizing," *IEEE Trans. Inform. Theory*, vol. IT-14, no. 5, pp. 676–683, Sept. 1968.
[52] B. Gold, "Digital speech networks," *Proc. IEEE*, vol. 65, no. 12, pp. 1636–1658, Dec. 1977.
[53] _____, "Experiments with a pattern-matching channel vocoder," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing* (Atlanta, GA, Apr. 1981), pp. 32–35.
[54] A. J. Goldberg, "Predictive coding with delayed decision," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing* (Hartford, CT, May 1977), pp. 405–408.
[55] W. Granzow and P. Noll, "Quantization of a memoryless gamma source," submitted to *IEEE Trans. Commun.*
[56] R. M. Gray, "Vector quantization," *IEEE ASSP Mag.*, vol. 1, pp. 4–29, Apr. 1984.
[57] R. M. Gray, A. Buzo, A. H. Gray, Jr., and Y. Matsuyama, "Distortion measures for speech processing," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-28, no. 4, pp. 367–376, Aug. 1980.
[58] R. M. Gray and E. D. Karnin, "Multiple local optima in vector quantizers," *IEEE Trans. Inform. Theory*, vol. IT-28, no. 2, pp. 256–261, Mar. 1982.
[59] R. M. Gray and J. C. Kieffer, "Asymptotically mean stationary measures," *Ann. Probability*, vol. 8, pp. 962–973, Oct. 1980.
[60] R. M. Gray and Y. Linde, "Vector quantizers and predictive quantizers for Gauss-Markov sources," *IEEE Trans. Commun.*, vol. COM-30, no. 2, pp. 381–389, Feb. 1982.
[61] A. H. Gray, Jr. and J. D. Markel, "Distance measures for speech processing," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-24, no. 5, pp. 380–391, Oct. 1976.
[62] A. Haoui and D. G. Messerschmitt, "Predictive vector quantization," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing* (San Diego, CA, Mar. 1984), paper 10.10.
[63] _____, "Embedded coding of speech: A vector quantization approach," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing* (Tampa, FL, Mar. 1985), pp. 1703–1706.
[64] J. A. Hartigan, *Clustering Algorithms.* New York: Wiley, 1975.
[65] J. Huang and P. Schultheiss, "Block quantization of correlated Gaussian random variables," *IEEE Trans. Commun. Syst.*, vol. CS-11, pp. 289–296, Sept. 1963.
[66] D. A. Huffman, "A method for the construction of minimum-redundancy codes," *Proc. IRE*, vol. 40, no. 9, pp. 1098–1101, Sept. 1952.
[67] IMSL Library, International Mathematical and Statistical Libraries, Inc., Houston, TX.
[68] F. Itakura, "Minimum prediction residual principle applied to speech recognition," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-23, no. 1, pp. 67–72, Feb. 1975.
[69] F. Itakura, S. Saito, T. Koike, H. Sawabe, and M. Nishikawa, "An audio response unit based on partial autocorrelation," *IEEE Trans. Commun.*, vol. COM-20, pp. 792–797, Aug. 1972.
[70] F. Itakura and S. Saito, "Analysis synthesis telephony based on the maximum likelihood method," in *Proc. 6th Int. Congr. Acoust.* (Tokyo, Japan, 1968), pp. C17–C20.
[71] _____, "A statistical method for estimation of speech spectral density and formant frequencies," *Electron. Commun. Japan*, vol. 53-A, pp. 36–43, 1970.
[72] _____, "On the optimum quantization of feature parameters in the PARCOR speech synthesizer," in *Conf. Rec., IEEE Conf. Speech Communication and Processing* (Newton, MA, Apr. 1972), pp. 434–437.
[73] N. S. Jayant, Ed., *Waveform Quantization and Coding.* New York, NY: IEEE PRESS, 1976.
[74] N. S. Jayant and P. Noll, *Digital Coding of Waveforms: Principles and Applications to Speech and Video.* Englewood Cliffs, NJ: Prentice-Hall, 1984.
[75] F. Jelinek, "Tree encoding of memoryless time-discrete sources with a fidelity criterion," *IEEE Trans. Inform. Theory*, vol. IT-15, pp. 584–590, Sept. 1969.
[76] B-H. Juang and A. H. Gray, Jr., "Multiple stage vector quantization for speech coding," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing* (Paris, France, May 1982), pp. 597–600.
[77] G. S. Kang and D. C. Coulter, "600 bps voice digitizer," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing* (Philadelphia, PA, Apr. 1976), pp. 91–94.
[78] N. Kitawaki and F. Itakura, "Nonlinear coding of PARCOR coefficients," in *Meeting of the Acoustical Society of Japan* (in Japanese), pp. 449–450, Oct. 1973.
[79] H. P. Kramer and M. V. Mathews, "A linear encoding for transmitting a set of correlated signals," *IRE Trans. Inform. Theory*, vol. IT-2, pp. 41–46, Sept. 1956.
[80] S. E. Levinson, "Structural methods in automatic speech recognition," this issue, pp. 1625–1650.
[81] Y. Linde, A. Buzo, and R. M. Gray, "An algorithm for vector quantizer design," *IEEE Trans. Commun.*, vol. COM-28, no. 1, pp. 84–95, Jan. 1980.
[82] S. P. Lloyd, "Least squares quantization in PCM," *IEEE Trans. Inform. Theory*, vol. IT-28, no. 2, pp. 129–137, Mar. 1982.
[83] Ph. Mabilleau and J-P. Adoul, "Medium band speech coding using a dictionary of waveforms," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing* (Atlanta, GA, Mar. 1981), pp. 804–807.
[84] J. MacQueen, "Some methods for classification and analysis of multivariate observations," in *Proc. 5th Berkeley Symp. on Math., Statist., and Prob.* Berkeley, CA: Univ. of California Press, 1967, pp. 281–297.
[85] P. C. Mahalanobis, "On the generalized distance in statistics," *Proc. Indian Nat. Inst. Sci.* (Calcutta), vol. 2, pp. 49–55, 1936.
[86] J. Makhoul, "Linear prediction: A tutorial review," *Proc. IEEE*, vol. 63, no. 4, pp. 561–580, Apr. 1975.
[87] _____, "Speech coding and processing," in *Modern Signal Processing*, T. Kailath, Ed. New York, NY: Hemisphere/Springer-Verlag, 1985, pp. 211–248.
[88] J. Makhoul and M. Berouti, "Adaptive noise spectral shaping and entropy coding in predictive coding of speech," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-27, no. 1, pp. 63–73, Feb. 1979.
[89] _____, "Predictive and residual encoding of speech," *J. Acoust. Soc. Amer.*, vol. 66, no. 6, pp. 1633–1641, Dec. 1979.
[90] J. D. Markel and A. H. Gray, Jr., *Linear Prediction of Speech.* New York, NY: Springer-Verlag, 1976.
[91] J. Max, "Quantizing for minimum distortion," *IRE Trans. Inform. Theory*, vol. IT-6, no. 2, pp. 7–12, Mar. 1960.
[92] R. J. McEliece, *The Theory of Information and Coding: A Mathematical Framework for Communication.* Reading, MA: Addison-Wesley, 1977.
[93] E. McLarnon, "A method for reducing the transmission rate of a channel vocoder by using frame interpolation," in *Proc.*

Exhibit 4 Page 497

IEEE Int. Conf. Acoust., Speech, Signal Processing (Tulsa, OK, Apr. 1978), pp. 458–461.

[94] P. Mermelstein, "Evaluation of a segmental SNR measure as an indicator of the quality of ADPCM coded speech," *J. Acoust. Soc. Amer.*, vol. 66, no. 6, pp. 1664–1667, Dec. 1979.

[95] A. Nadas, R. L. Mercer, L. R. Bahl, R. Bakis, P. S. Cohen, A. G. Cole, F. Jelinek, and B. L. Lewis, "Continuous speech recognition with automatically selected acoustic prototypes obtained by either bootstrapping or clustering," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing* (Atlanta, GA, Apr. 1981), pp. 1153–1155.

[96] P. Noll, "Adaptive quantizing in speech coding systems," in *Proc. Int. Zurich Seminar on Digital Commun.* (Zurich, Switzerland, Mar. 1974), paper B3.

[97] P. Noll and R. Zelinski, "Bounds on quantizer performance in the low bit-rate region," *IEEE Trans. Commun.*, vol. COM-26, no. 2, pp. 300–304, Feb. 1978.

[98] J. J. O'Donnell, "A system for very low data rate speech communication," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing* (Atlanta, GA, Apr. 1981), pp. 8–11.

[99] E. A. Patrick, *Fundamentals of Pattern Recognition.* Englewood Cliffs, NJ: Prentice-Hall, 1972.

[100] D. B. Paul, "An 800 bps adaptive vector quantization vocoder using a perceptual distance measure," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing* (Boston, MA, Apr. 1983), pp. 73–76.

[101] ——, "The Lincoln low-rate vocoder: A 1200/2400 bps LPC-10 voice terminal," Tech. Rep. 676, Lincoln Lab., Lexington, MA, Mar. 1984, DCC AD-A141291/5.

[102] D. B. Paul and P. E. Blankenship, "Two distance measure-based vocoder quantization algorithms for very-low-data rate applications: Frame-fill and spectral vector quantization," in *Proc. IEEE Int. Conf. Commun.* (June 1982), pp. 1–6.

[103] W. W. Peterson and E. J. Weldon, Jr., *Error-Correcting Codes*, 2nd ed. Cambridge, MA: M.I.T. Press, 1972.

[104] R. Pieraccini and R. Billi, "Experimental comparison among data compression techniques in isolated word recognition," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing* (Boston, MA, Apr. 1983), pp. 1025–1028.

[105] S. U. H. Qureshi and G. D. Forney, Jr., "Adaptive residual coder—An experimental 9.6/16 kb/s speech digitizer," in *EASCON Rec.*, pp. 29A–29E, 1975.

[106] L. Rabiner, S. E. Levinson, and M. M. Sondhi, "On the application of vector quantization and hidden Markov models to speaker-independent isolated word recognition," *Bell Syst. Tech. J.*, vol. 62, pp. 1075–1105, Apr. 1983.

[107] L. R. Rabiner and R. W. Schafer, *Digital Processing of Speech Signals.* Englewood Cliffs, NJ: Prentice-Hall, 1978.

[108] L. R. Rabiner, M. M. Sondhi, and S. E. Levinson, "A vector quantizer incorporating both LPC shape and energy," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing* (San Diego, CA, Mar. 1984), paper 17.1.

[109] H. Reininger and D. Wolf, "Speech and speaker independent codebook design in VQ coding schemes," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing* (Tampa, FL, Mar. 1985), pp. 1700–1702.

[110] S. Roucos, J. Makhoul, and R. Schwartz, "A variable-order Markov chain for coding of speech spectra," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing* (Paris, France, May 1982), pp. 582–585.

[111] S. Roucos, R. Schwartz, and J. Makhoul, "Segment quantization for very-low-rate speech coding," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing* (Paris, France, May 1982), pp. 1565–1569.

[112] ——, "Vector quantization for very-low-rate coding of speech," in *Proc. IEEE Global Telecommunications Conf.* (Miami, FL, Nov. 29–Dec. 2, 1982), pp. 1074–1078.

[113] ——, "A segment vocoder at 150 B/S," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing* (Boston, MA, Apr. 1983), pp. 61–64.

[114] M. J. Sabin and R. M. Gray, "Product code vector quantizers for waveform and voice coding," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-32, no. 3, pp. 474–488, June 1984.

[115] D. J. Sakrison, "A geometric treatment of the source encoding of a Gaussian random variable," *IEEE Trans. Inform. Theory*, vol. IT-14, no. 3, pp. 96–101, May 1968.

[116] M. R. Sambur, "An efficient linear-prediction vocoder," *Bell Syst. Tech. J.*, vol. 54, pp. 1693–1723, Dec. 1975.

[117] M. R. Schroeder and B. S. Atal, "Code-excited linear prediction (CELP): High-quality speech at very low bit rates," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing* (Tampa, FL, Mar. 1985), pp. 937–940.

[118] R. Schwartz, Y. Chow, S. Roucos, M. Krasner, and J. Makhoul, "Improved hidden Markov modeling of phonemes for continuous speech recognition," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing* (San Diego, CA, Mar. 1984), paper 35.6.

[119] R. Schwartz and S. Roucos, "A comparison of methods for 300–400 b/s vocoders," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing* (Boston, MA, Apr. 1983), pp. 69–72.

[120] A. Segall, "Bit allocation and encoding for vector sources," *IEEE Trans. Inform. Theory*, vol. IT-22, no. 2, pp. 162–169, Mar. 1976.

[121] I. K. Sethi, "A fast algorithm for recognizing nearest neighbors," *IEEE Trans. Syst., Man, Cybern.*, vol. SMC-11, no. 3, pp. 245–248, Mar. 1981.

[122] C. E. Shannon, "A mathematical theory of communication," *Bell Syst. Tech. J.*, vol. 27, pp. 379–423, 623–656, 1948.

[123] ——, "Coding theorems for a discrete source with a fidelity criterion," in *IRE Nat. Conv. Rec.* (pt. 4), pp. 142–163, 1959. (Also in *Information and Decision Processes*, R. E. Machol, Ed. New York, NY: McGraw-Hill, 1960, pp. 93–126.)

[124] Y. Shoham and A. Gersho, "Pitch synchronous transform coding of speech at 9.6 kb/s based on vector quantization," in *Proc. IEEE Int. Conf. Commun.* (Amsterdam, The Netherlands, May 1984), pp. 1179–1182.

[125] ——, "Efficient codebook for an arbitrary set of vector quantizers," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing* (Tampa, FL, Mar. 1985), pp. 1696–1699.

[126] J. E. Shore, D. Burton, and J. Buck, "A generalization of isolated word recognition using vector quantization," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing* (Boston, MA, Apr. 1983), pp. 1021–1024.

[127] C. P. Smith, "Perception of vocoder speech processed by pattern matching," *J. Acoust. Soc. Amer.*, vol. 46, no. 6 (pt. 2), pp. 1562–1571, June 1969.

[128] L. C. Stewart, R. M. Gray, and Y. Linde, "The design of trellis waveform coders," *IEEE Trans. Commun.*, vol. COM-30, no. 4, pp. 702–710, Apr. 1982.

[129] J. T. Tou and R. C. Gonzales, *Pattern Recognition Principles.* Reading, MA: Addison-Wesley, 1974.

[130] J. M. Tribolet and R. E. Crochiere, "Frequency domain coding of speech," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-27, no. 5, pp. 512–530, Oct. 1979.

[131] R. Viswanathan and J. Makhoul, "Quantization properties of transmission parameters in linear predictive systems," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-23, no. 3, pp. 309–321, June 1975.

[132] V. Viswanathan, J. Makhoul, R. Schwartz, and A. W. F. Huggins, "Variable-frame-rate transmission: A review of methodology and application to narrow-band LPC speech coding," *IEEE Trans. Commun.*, vol. COM-30, no. 4, pp. 674–686, Apr. 1982.

[133] V. Viswanathan, W. Russell, A. Higgins, M. Berouti, and J. Makhoul, "Speech-quality optimization of 16 kb/s adaptive predictive coders," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing* (Denver, CO, Apr. 1980), pp. 520–525.

[134] V. Viswanathan, W. Russell, and A. W. F. Huggins, "Objective speech quality evaluation of mediumband and narrowband real-time speech coders," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing* (Boston, MA, Apr. 1983), pp. 543–546.

[135] A. J. Viterbi and J. K. Omura, "Trellis encoding of memoryless discrete-time sources with a fidelity criterion," *IEEE Trans. Inform. Theory*, vol. IT-20, pp. 325–332, May 1974.

[136] S. G. Wilson and S. Hussain, "Adaptive tree encoding of rion," *IEEE Trans. Commun.*, vol. COM-27, pp. 165–170, Jan. 1979.

[137] D. Y. Wong, B. H. Juang, and D. Y. Cheng, "Very low data rate speech compression with LPC vector and matrix quanti-

Exhibit 4 Page 495

zation," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing* (Boston, MA, Apr. 1983), pp. 65–68.

[138] D. Y. Wong, B. H. Juang, and A. H. Gray, Jr., "An 800 bit/s vector quantization LPC vocoder," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-30, no. 5, pp. 770–780, Oct. 1982.

[139] Y. Yamada, S. Tazaki, and R. M. Gray, "Asymptotic performance of block quantizers with difference distortion measures," *IEEE Trans. Inform. Theory*, vol. IT-26, no. 1, pp. 6–14, Jan. 1980.

[140] T. P. Yunck, "A technique to identify nearest neighbors," *IEEE Trans. Syst., Man, Cybern.*, vol. SMC-6, pp. 678–683, Oct. 1976.

[141] P. L. Zador, "Asymptotic quantization error of continuous signals and the quantization dimension," *IEEE Trans. Inform. Theory*, vol. IT-28, no. 2, pp. 139–149, Mar. 1982.

[142] R. Zelinski and P. Noll, "Adaptive transform coding of speech signals," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-25, no. 4, pp. 299–309, Aug. 1977.

[143] J. M. Ziman, *Principles of the Theory of Solids*. Cambridge, UK: Cambridge Univ. Press, 1972.

Exhibit Y Page 496