THE BELL SYSTEM TECHNICAL JOURNAL
Vol. 62, No. 8, October 1983
Printed in U.S.A.

Note on the Properties of a Vector Quantizer for LPC Coefficients

By L. R. RABINER,* M. M. SONDHI,* and S. E. LEVINSON*

(Manuscript received March 15, 1983)

Vector quantization has been used in coding applications for several years. Recently, quantization of linear predictive coding (LPC) vectors has been used for speech coding and recognition. In these latter applications, the only method that has been used for deriving the vector quantizer code book from a set of training vectors is the one described by Linde, Buzo, and Gray. In this paper, we compare this algorithm to several alternative algorithms and also study the properties of the resulting code books. Our conclusion is that the various algorithms that we tried gave essentially identical code books.

I. INTRODUCTION

The technique of vector quantization for LPC voice coding has been in use for several years, and has been shown to be of great utility for LPC analysis/synthesis systems.[1–4] Recently, vector quantization of LPC vectors has been applied to speech-recognition systems both in direct applications[5,6] and in conjunction with work on the application of hidden Markov models (HMMs) to recognition.[7,8]

The main idea of vector quantization is summarized as follows: assume that a training set $\{T\}$ of I LPC vectors is given. It is desired to find a code book of M^* LPC vectors such that the average distance of a vector in $\{T\}$ from the closest code book entry is minimized. Thus we wish to find a set $\{R\}$ of reference vectors that minimizes the

* Bell Laboratories.

average distance $\bar{D}_I(M^*)$ given by

$$\bar{D}_I(M^*) = \min_{\{R\}} \left[\frac{1}{I} \sum_{i=1}^{I} \min_{1 \leq m \leq M^*} [d(T_i, R_m)] \right], \tag{1}$$

where $d(T_i, R_m)$ is the LPC distance between training vector T_i and code book entry R_m.

The optimum code book is generated by a method similar to the K-means algorithm. Starting with an initial guess of M^* entries, each vector of the training set is assigned to the closest entry. The centroids of the M^* subsets (clusters) obtained in this manner are used as new trial entries in the code book, and the iteration is continued until some stopping criterion is satisfied.

For large M^*, the choice of initial guesses can be quite important, and it is unlikely that a randomly chosen initial guess is a good one. For this reason the splitting algorithm was devised in Ref. 1. In this algorithm a code book of $M = 2$ entries is optimized, as described above, starting with a random initial guess. Next, each optimum code book entry for $M = 2$ is split into 2 and used as an initial guess for a code book of size $2M$. This process is used until $M = M^*$. To distinguish this algorithm from others considered later, we call it the *binary*-split algorithm.

To the best of our knowledge, all speech-related applications of vector quantization so far have used this binary-split algorithm. However, a priori, the requirement that every code word be split appears to be too restrictive. For example, after optimizing an $M = 2$ code book, if one cluster contains almost all the training set and the other contains just a few elements, it might be argued that only the larger cluster should be split. Thus it is of interest to consider "single-split" algorithms in which a single cluster is split at each iteration.

For very large M^* (e.g., 1024 or 2048) single-split algorithms might require prohibitive amounts of computation. However, M^* on the order of 64 or 128 can be quite useful in certain applications.[8] In these cases a single-split algorithm is quite feasible. In any case, it is of interest to know whether or not a single-split algorithm yields a better code book than the binary-split algorithm.

There are at least three reasonable ways of implementing the splitting rule of a single-split algorithm for training the vector quantizer. To describe these three splitting rules we need some definitions. Let

$\{\hat{T}_M(m)\}$ = The set of training vectors represented by the mth code book entry (cluster) in a size M vector quantizer

$C_M(m)$ = The number of training vectors in $\hat{T}_M(m)$

$d_M(m)$ = The average distance (distortion) of the $C_M(m)$ vectors from the mth code-book entry

$D_M(m)$ = The total distance (distortion) of the $C_M(m)$ vectors.

We then have the relationships

$$I = \sum_{m=1}^{M} C_M(m) \tag{2}$$

$$d_M(m) = \frac{1}{C_M(m)} \sum_{q=1}^{C_M(m)} d(\hat{T}_M(m)_q, R_m) \tag{3}$$

$$D_M(m) = C_M(m) \cdot d_M(m). \tag{4}$$

Using eqs. (2) through (4) we can write the average distortion of eq. (1) as

$$\bar{D}_I(m) = \min_{\{R\}} \left[\frac{\sum_{m=1}^{M} D_M(m)}{\sum_{m=1}^{M} C_M(m)} \right] \tag{5a}$$

$$= \min_{\{R\}} \left[\frac{\sum_{m=1}^{M} d_M(m) C_M(m)}{\sum_{m=1}^{M} C_M(m)} \right]. \tag{5b}$$

Based on the above definitions, the three splitting rules we have considered are:

Rule 1: Split the cluster, m, with the largest number of vectors, $C_M(m)$. We denote the resulting (vector quantizer) VQ code-word set as R_c.

Rule 2: Split the cluster, m, with the largest average distortion, $d_M(m)$. We denote the resulting VQ code-word set as R_d.

Rule 3: Split the cluster, m, with the largest total distortion, $D_M(m)$. We denote the resulting VQ code-word set as R_D.

The key issue is how do the different splitting rules affect the properties of the resulting vector quantizer—in particular the average distortion [eq. (1)] and the coverage of the LPC space.

We have run a series of experimental evaluations of the single-split and binary-split algorithms for training the VQ. We have found that each of the different splitting criteria leads to a different reference prototype set (VQ code book); however, all the VQ sets had essentially the *same average distortion*. We were also able to show that the coverage of the LPC space for all VQ sets was identical, and that the average distance of any one VQ set from another VQ set was smaller than the average distortion of the training set. Hence, the different implementations of the training algorithm for the VQ lead to equivalent VQ reference sets. Thus for any practical application the simple binary-split algorithm is effective for deriving the VQ code book entries.

The outline of this paper is as follows. In Section II we review the Linde et al.[1] implementation of the binary-split VQ training algorithm and show how we modified it to handle the single-split case. In Section III we discuss the results of several experiments on testing the different implementations of the training algorithm. In Section IV we provide a discussion and summary of the results.

II. IMPLEMENTATION OF THE VQ TRAINING ALGORITHM

The implementation of the VQ training algorithm is essentially the one proposed by Linde et al.[1] A flow diagram of this procedure for the binary-split case is given in Fig. 1a and for the single-split case in Fig. 1b. Given M code words, each vector of the training set T is assigned to the code word closest to it. The average distortion $\bar{D}_I(M)$ is computed for this assignment of the I training vectors to M clusters. M new code words are obtained as centroids (i.e., averaged normalized autocorrelations) of each cluster, and the distortion $\bar{D}_I(M)$ computed again. This process is iterated until it converges, i.e., until the percent change in distortion is less than a preset value ϵ (chosen to be 1 percent in our simulations). Once convergence is achieved, M is doubled by splitting each code word into two. The entire process is repeated until $M = M^*$. The iteration is initialized by choosing two arbitrary code words.

In our implementation, we made one modification to the VQ training algorithm of Fig. 1. We inserted a check after the classification of the training set vectors to see if any cluster is empty (i.e., contains none of the training set vectors). In such a case the "largest" cluster is split into two clusters, and the convergence test is bypassed (to ensure a reclassification in which each cluster is nonempty). However, for the data used in this experiment, an empty cluster never occurred. In subsequent tests with larger M^* we did encounter such cases.

For the single-split algorithm (Fig. 1b), only one modification is required. After convergence, only the "largest cluster" is split. Here largest can refer to the cluster with the largest average distortion, total distortion, or count.

For a convergence criterion of $\epsilon = 1$ percent, typically it takes three to six iterations of the classification loop to obtain a convergent set of clusters and centroids. We also found that the algorithms of Fig. 1a and 1b work extremely reliably over a broad range of types of training

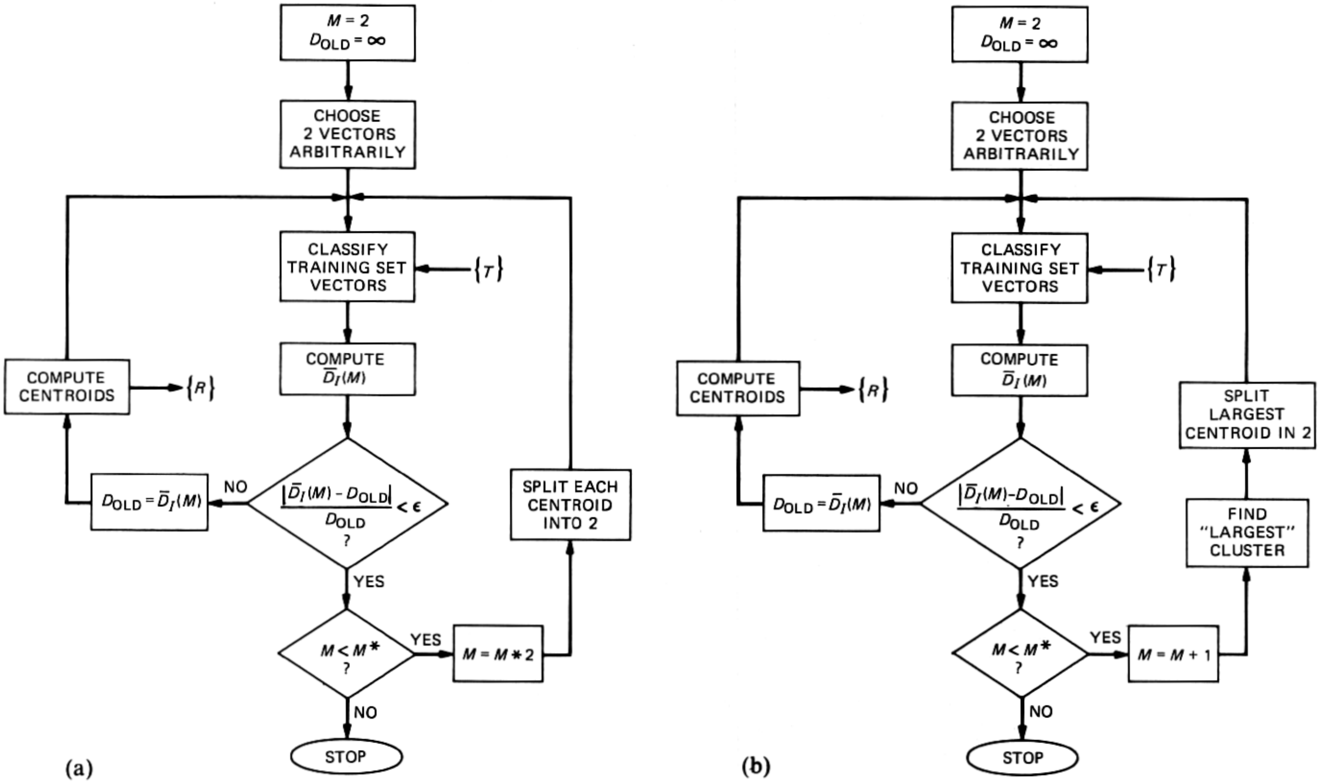

Fig. 1—Flow charts of the vector quantizer training algorithms. (a) The binary-split algorithm. (b) The single-split algorithm.

data (e.g., collected from a single talker, collected from many talkers, collected from a corpus of isolated words, collected from sentence-length material, etc.).

III. COMPARISON OF THE BINARY- AND SINGLE-SPLIT ALGORITHMS

To compare the performances of the binary- and single-split VQ training algorithms of Fig. 1, several tests were run. The database consisted of a set of 39,708 LPC vectors. The LPC analysis used a 6.67-kHz sampling rate and an eigth-order analysis of 300 sample (45 ms) frames of speech. The sample frames had been preemphasized with a simple, first-order digital network (preemphasis factor of 0.95) and windowed by a 300-sample Hamming window. Frames were taken 100 samples apart across the duration of each word of a series of 1000 isolated words (digits) spoken by 100 talkers (50 male, 50 female). All recordings were made over dialed-up telephone lines through a local PBX connection. All silence outside the spoken words was eliminated by a word endpoint detector;[9] hence, all LPC training frames were from within word boundaries.

Several aspects of the binary- and single-split training algorithms were studied. The first question considered was whether the two training procedures yielded identical results (i.e., whether the resulting LPC code words and the clusters from which they were derived were identical). Figure 2 shows plots of the cluster splitting for an $M^* = 8$ solution for the binary-split algorithm (Fig. 2a) and the single-split algorithm based on average distance splitting (Fig. 2b). It can be seen that the resulting eight clusters in the single-split case come from very different splits than those for the binary-split case. For example, in

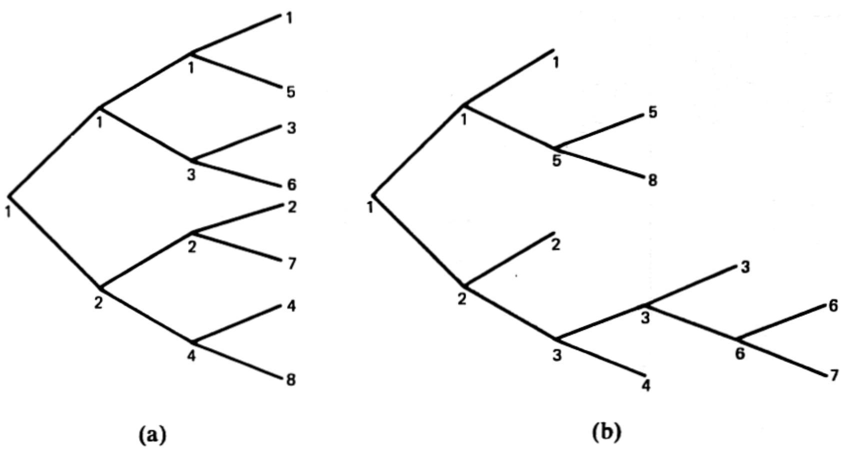

Fig. 2—Splitting charts for an $M^* = 8$ vector quantizer with splits based on average distortion. (a) The binary-split training algorithm. (b) The single-split algorithm.

the single-split case, final clusters 6 and 7 come from four splits of the original cluster 2, whereas final clusters 1 and 2 come from single splits of original clusters 1 and 2. In the binary-split case all final clusters come from two splits of original clusters 1 and 2. Similarly, the actual clusters were grossly different for the three different criteria for the single-split algorithm.

The next question we considered was how the different training procedures differed in performance. Figures 3 through 5 show a series of plots of statistics comparing some of the details of the individual training procedures. For each of these plots, Parts (a) through (d) show results for the binary-split case, the single-split case based on count, the single-split case based on average distortion, and the single-

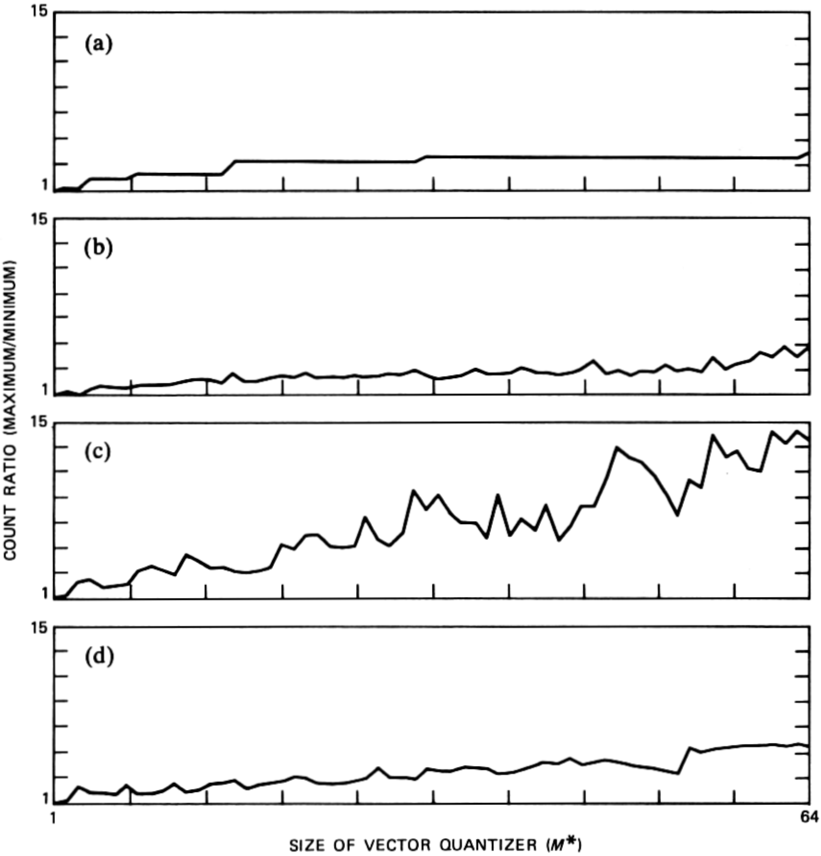

Fig. 3—Plots of count ratio (maximum cluster count divided by minimum cluster count) as a function of the size of the vector quantizer. (a) Binary-split training. (b) Single-split training based on count. (c) Single-split training based on average distortion. (d) Single-split training based on total distortion.

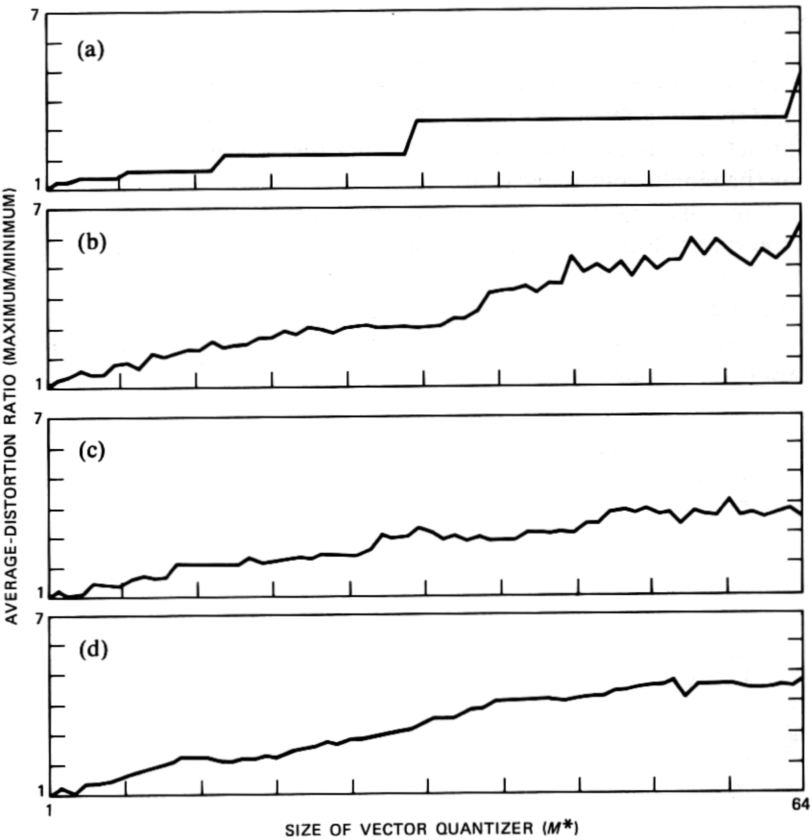

Fig. 4—Plots of average distortion ratio as a function of the size of the vector quantizer. (a) Binary-split training. (b) Single-split training based on count. (c) Single-split training based on average distortion. (d) Single-split training based on total distortion.

split case based on total distortion. The statistics plotted are ratio of maximum to minimum cluster count (Fig. 3), ratio of maximum to minimum average distortion (Fig. 4), and ratio of maximum to minimum total distortion (Fig. 5) versus size of the vector quantizer. These statistics were chosen because each of them should ideally approach 1.0 for clusters that are of equal size according to the corresponding splitting criterion. For example, we would expect the count ratio to approach 1.0 for the split on count criterion but not necessarily for the other splitting criteria.

Examination of Figs. 3 through 5 shows several interesting things. As seen in Fig. 3, the count ratio for the binary-split case for $M^* = 64$ (4.1) is actually smaller than the count ratio for the single split on count case for $M^* = 64$ (4.8). The count ratios for the other two split

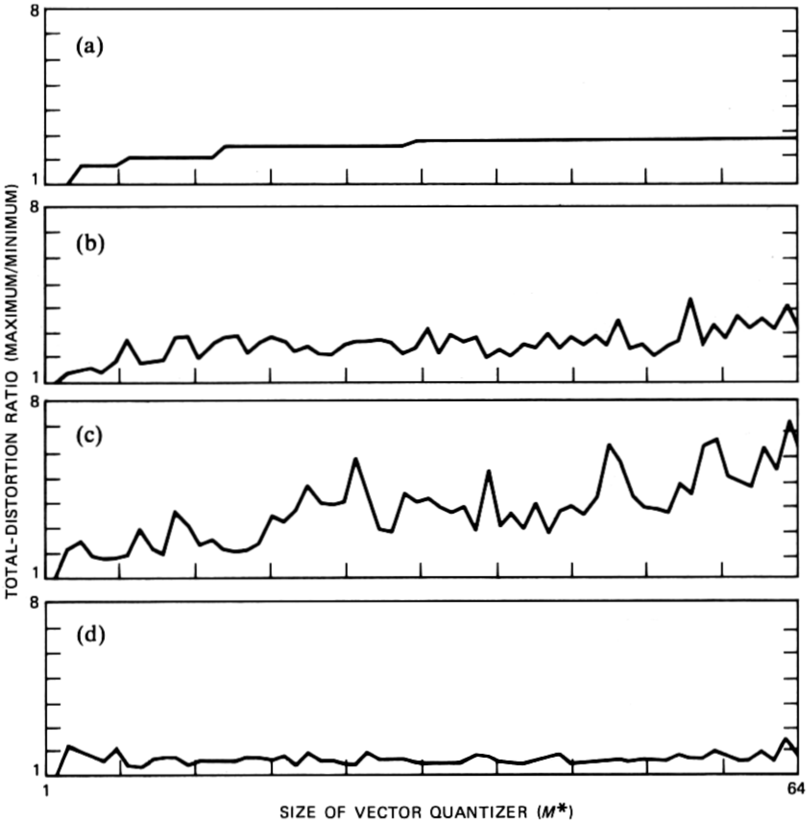

Fig. 5—Plots of total-distortion ratio as a function of the size of the vector quantizer. (a) Binary-split training. (b) Single-split training based on count. (c) Single-split training based on average distortion. (d) Single-split training based on total distortion.

criteria are indeed larger than for the split on count, as expected. Figure 4 shows that the average-distortion ratio is smallest (4.1) at $M^* = 64$ for the single split on average-distortion case; however, the distortion ratios for the binary case (4.4) and the single split on total-distortion (4.7) cases are only slightly larger. Finally, Fig. 5 shows a similar set of results on the total-distortion-ratio statistic in which the results for $M^* = 64$ for the binary-split case (2.7) are only slightly worse than for the single split on total-distortion case (2.6).

The results of Figs. 3 through 5 indicate that the binary-split case seems to yield cluster training statistics that are almost as good as the best statistics for any of the single-split cases in terms of count ratio, average-distortion ratio, and total-distortion ratio. Hence, from the point of view of cluster statistics, the binary-split cases appear to give the best overall performance.

Two gross performance checks were made on the training algorithms. In the first test, the average distance between vector quantizer sets obtained from the different training procedures was calculated as a function of M^*. The results of this test are given in Table I. It can

Table I—Average distance between code book entries of vector quantizers designed on the basis of count (R_c), average distortion (R_d), total distortion (R_D), and binary splitting (R_B)

M^*	$\bar{d}(R_c, R_d)$	$\bar{d}(R_c, R_D)$	$\bar{d}(R_c, R_B)$	$\bar{d}(R_d, R_D)$	$\bar{d}_I(M^*)$†
4	0.384	0.019	0.047	0.270	0.707
8	0.125	0.138	0.157	0.101	0.426
16	0.148	0.143	0.160	0.065	0.326
32	0.191	0.108	0.175	0.132	0.255
64	0.216	0.131	0.148	0.131	0.203

† Average distance between the training vectors and the code words representing them.

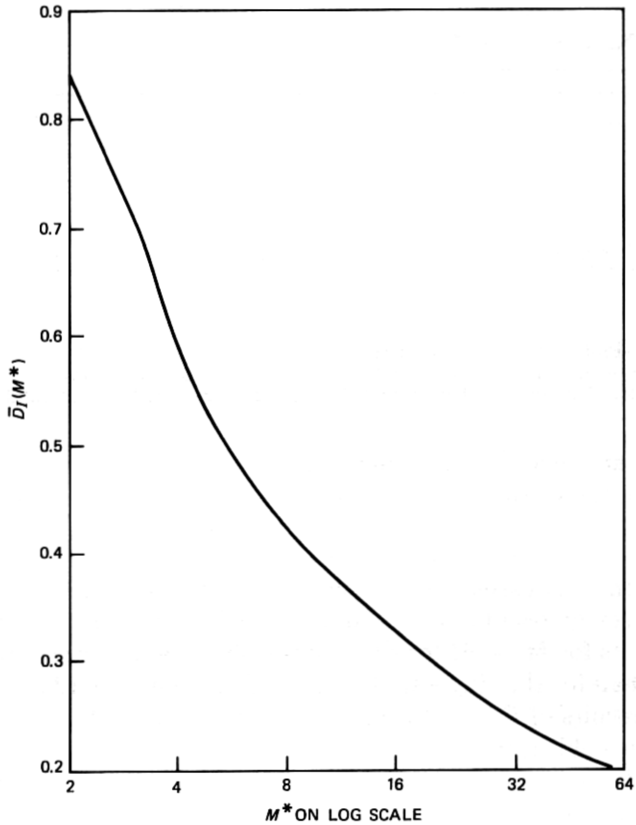

Fig. 6—Plot of average training set distortion $\bar{D}_I(M^*)$ as a function of the size of the vector quantizer.

be seen that the average distance between vector quantizer sets is as small or smaller than the average distance of the training vectors to the code book sets. Hence, the code book sets derived from the different training algorithms are, on average, quite close to each other.

The second test we performed was to measure the average distortion, $\bar{D}_I(M^*)$ versus M^* for the different training algorithms for values of M^* from 2 to 64. The results of this test are plotted in Fig. 6. On the

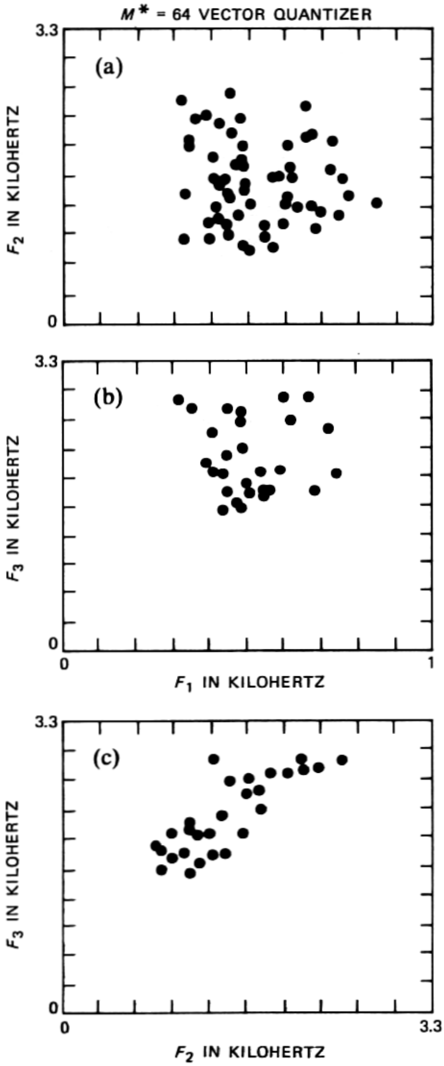

Fig. 7—Plots of code-word coverage in the F_1-F_2, F_1-F_3, and F_2-F_3 planes for an M^* = 64 vector quantizer.

scale of this plot, the differences in average distortion are *indistinguishable* among the different vector quantizers.

The third and final question we considered concerns the coverage of the space of speech sounds by the optimum code books. A good way of displaying this coverage is to look at the code books in the space of formant frequencies. The formant frequencies (and bandwidths) for each entry of the code book are given by the zeroes of the trigonometric polynomial associated with it. Thus each code book may be displayed

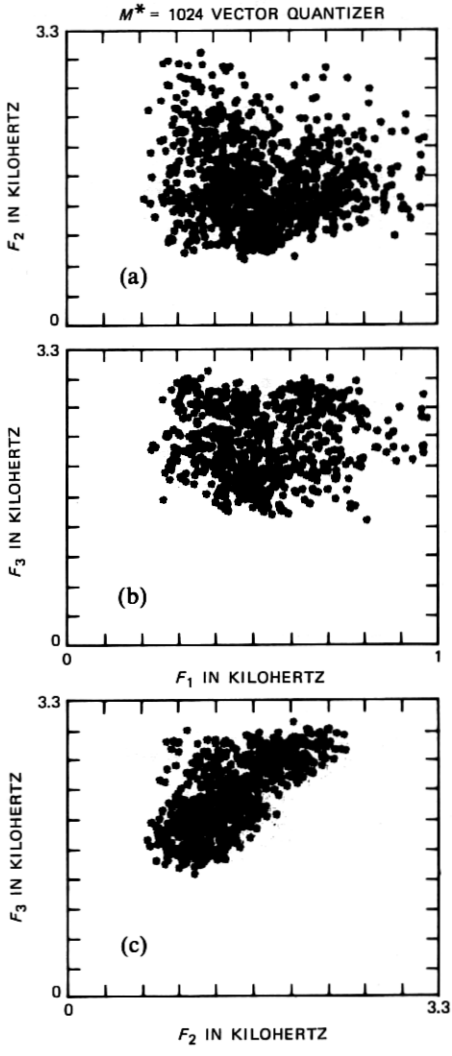

Fig. 8—Plots of code-word coverage in the F_1-F_2, F_1-F_3, and F_2-F_3 planes for an M^* = 1024 vector quantizer.

as a scatter plot in F_1-F_2-F_3 space. Projections of this scatter diagram on the F_1-F_2, F_1-F_3, and F_2-F_3 planes are shown for a typical code book in Figs. 7 and 8 for the code books obtained from the binary-split training algorithm for $M^* = 64$ (Fig. 7) and $M^* = 1024$ (Fig. 8). It is seen that the code words cover the expected regions in the formant frequency planes fairly uniformly. The major difference between the coverage of the $M^* = 1024$ and the $M^* = 64$ code books is the density of coverage of the areas in the respective formant frequency planes. The coverage of the single-split algorithms for $M^* = 64$ was essentially identical to that of the binary-split algorithm.

IV. DISCUSSION

Our overall conclusion from the tests that compared the fine and gross differences in clustering LPC vectors via a VQ training algorithm is that all the variations in the training procedure that we studied (i.e., different splitting procedures, different convergence criteria, etc.) lead to essentially indistinguishable differences in the set of VQ code book entries. Since the binary-split algorithm, as discussed by Linde et al.[1] requires the least amount of computation, it is the best of the algorithms considered.

In this paper we present the results of a series of experiments on a training set of 39,708 vectors. More recently we have experimented with the binary-split VQ training procedure on a number of different training sets whose size varied from 10,000 to 600,000 vectors. We found that the training procedure always rapidly and reliably converged to a set of code book vectors whose properties were similar to those described in this paper. We are currently using the VQ code book sets in work related to speech recognition and speech coding.

V. ACKNOWLEDGMENTS

The authors gratefully acknowledge several fruitful discussions with Fred Juang of Bell Laboratories concerning the characteristic properties of the VQ clusters. Juang's insight into the training procedure and the resulting properties of the code book vectors coincided with the results we found in this study.

REFERENCES

1. Y. Linde, A. Buzo, and R. M. Gray, "An Algorithm for Vector Quantization," IEEE Trans. Commun., *COM-28*, No. 1 (January 1980), pp. 84–95.
2. B. Juang, D. Wong, and A. H. Gray, Jr., "Distortion Performance of Vector Quantization for LPC Voice Coding," IEEE Trans. on Acoust., Speech, and Signal Proc., *ASSP-30*, No. 2 (April 1982), pp. 294–303.
3. A. Buzo, et al., "Speech Coding Based Upon Vector Quantization," IEEE Trans. on Acoust., Speech, and Signal Proc., *ASSP-28*, No. 5 (October 1980), pp. 562–74.
4. D. Wong, B. Juang, and A. H. Gray, Jr., "An 800 Bit/S Vector Quantization LPC Vocoder," IEEE Trans. on Acoust., Speech, and Signal Proc., *ASSP-30*, No. 5 (October 1982), pp. 770–80.

5. A. Buzo, H. Martinez, and C. Rivera, "Discrete Utterance Recognition Based Upon Source Coding Techniques," Proc. ICASSP-82 (May 1982), pp. 539–42.
6. J. E. Shore and D. Burton, "Discrete Utterance Speech Recognition Without Time Normalization," Proc. ICASSP-82 (May 1982), pp. 907–10.
7. R. Billi, "Vector Quantization and Markov Source Models Applied to Speech Recognition," Proc. ICASSP-82 (May 1982), pp. 574–7.
8. L. R. Rabiner, S. E. Levinson, and M. M. Sondhi, "On the Application of Vector Quantization and Hidden Markov Models to Speaker Independent, Isolated Word Recognition," B.S.T.J., *62*, No. 4 (April 1983), pp. 1075–105.
9. L. F. Lamel, et al. "An Improved Endpoint Detector for Isolated Word Recognition," IEEE Trans. on Acoust., Speech, and Signal Proc., *ASSP-29*, No. 4 (August 1981), pp. 777–85.

AUTHORS

Stephen E. Levinson, B. A. (Engineering Sciences), 1966, Harvard; M.S. and Ph.D. (Electrical Engineering), University of Rhode Island, Kingston, 1972 and 1974, respectively; General Dynamics, 1966–1969; Yale University, 1974–1976; Bell Laboratories, 1976—. From 1966 to 1969, Mr. Levinson was a design engineer at Electric Boat Division of General Dynamics in Groton, Connecticut. From 1974 to 1976, he held a J. Willard Gibbs Instructorship in Computer Science at Yale University. In 1976, he joined the technical staff at Bell Laboratories, where he is pursuing research in the areas of speech recognition and cybernetics. Member, Association for Computing Machinery; Fellow, Acoustical Society of America; Senior Member, IEEE, editorial board of Speech Technology; Associate Editor, IEEE Transactions on Acoustics, Speech and Signal Processing.

Lawrence R. Rabiner, S.B. and S. M., 1964, Ph.D. (Electrical Engineering), The Massachusetts Institute of Technology: Bell Laboratories, 1962—. From 1962 through 1964, Mr. Rabiner participated in the cooperative plan in electrical engineering at Bell Laboratories. He worked on digital circuitry, military communications problems, and problems in binaural hearing. Presently, he is engaged in research on speech communications and digital signal processing techniques. He is coauthor of *Theory and Application of Digital Signal Processing* (Prentice-Hall, 1975), *Digital Processing of Speech Signals* (Prentice-Hall, 1978), and *Multirate Digital Signal Processing* (Prentice-Hall, 1983). Former President, IEEE, ASSP Society; former Associate Editor, ASSP Transactions; former member, Technical Committee on Speech Communication of the Acoustical Society, ASSP Technical Committee on Speech Communication; Member, IEEE Proceedings Editoral Board, Eta Kappa Nu, Sigma Xi, Tau Beta Pi. Fellow, Acoustical Society of America, IEEE.

Man Mohan Sondi, B.Sc. (Physics), Honours degree, 1950, Delhi University, Delhi, India; D.I.I.Sc. (Communications Engineering), 1953, Indian Institute of Science, Bangalore, India; M.S., 1955; Ph.D. (Electrical Engineering), 1957, University of Wisconsin, Madison, Wisconsin; Bell Laboratories, 1962—. Before joining Bell Laboratories, Mr. Sondhi worked for a year at the Central Electronics Engineering Research Institute, Pilani, India and taught for a year at the University of Toronto. At Bell Laboratories his research has included work on speech signal processing, echo cancellation, adaptive filtering, modelling of auditory and visual processes, and acoustical inverse problems. From 1971 to 1972 Mr. Sondhi was a guest scientist at the Royal Institute of Technology, Stockholm, Sweden.