A Two Codebook Format for Robust Quantization of Line Spectral Frequencies

Ravi P. Ramachandran, Man Mohan Sondhi, Nambi Seshadri, *Member, IEEE,* and Bishnu S. Atal, *Fellow, IEEE*

***Abstract*— An important problem in speech coding is the quantization of linear predictive coefficients (LPC) with the smallest possible number of bits while maintaining robustness to a large variety of speech material and transmission media. Since direct quantization of LPC's is known to be unsatisfactory, we consider this problem for an equivalent representation, namely, the line spectral frequencies (LSF). To achieve an acceptable level of distortion a scalar quantizer for LSF's requires a 36 bit codebook. We derive a 30 bit two-quantizer scheme which achieves a performance equivalent to this scalar quantizer. This equivalence is verified by tests on data taken from various types of filtered speech, speech corrupted by noise and by a set of randomly generated LSF's. The two-quantizer format consists of both a vector and a scalar quantizer such that for each input, the better quantizer is used. The vector quantizer is designed from a training set that reflects the joint density (for coding efficiency) and which ensures coverage (for robustness). The scalar quantizer plays a pivotal role in dealing better with regions of the space that are sparsely covered by its vector quantizer counterpart. A further reduction of 1 bit is obtained by formulating a new adaptation algorithm for the vector quantizer and doing a dynamic programming search for both quantizers. The method of adaptation takes advantage of the ordering of the LSF's and imposes no overhead in memory requirements. The dynamic programming search is feasible due to the ordering property. Subjective tests in a speech coder reveal that the 29 bit scheme produces equivalent perceptual quality to that when the parameters are unquantized.**

I. Introduction

A LINEAR predictive (LPC) analysis [1] of a speech signal, based on the model that a speech sample is a weighted linear combination of p previous samples, results in the set of weights $a(i)$. These weights correspond to the direct form coefficients of a nonrecursive filter $A(z) = 1 + \Sigma_1^p\, a(i)z^{-i}$. Passing the speech signal through the filter $A(z)$ results in the removal of the near-sample correlations and produces an LPC residual. In addition, the magnitude spectrum of $1/A(z)$ describes the spectral envelope of the speech being analyzed. Therefore, transforming the speech into the LPC residual also has the frequency domain interpretation that the spectral envelope is removed.

In predictive speech coders, the LPC residual and LPC parameters $a(i)$ are quantized and coded for transmission. In this paper, we investigate the quantization of only the LPC parameters. The goal of LPC parameter quantization is to minimize the spectral distortion (SD) or the root mean-square deviation between the true log magnitude spectrum of $1/A(z)$ and the spectrum that results after quantization. Quantization of the parameters $a(i)$ is known not to be efficient and can lead to an unstable filter $1/A(z)$. We therefore transform the LPC coefficients to an equivalent set of parameters, namely, the line spectral frequencies (LSF) $f(i)$ [2]–[5]. Line spectral frequencies have several properties that make them more suitable for quantization. They are approximately related to the formant frequencies and bandwidths and show a localized spectral sensitivity. Also, it is possible to define a tractable distortion measure in terms of LSF's, which is closely related to the SD. Finally, it is possible to quantize LSF's without destroying an ordering property which guarantees stability of $1/A(z)$. We have chosen the order of the LPC analysis to be $p = 10$, which gives us ten LSF's to quantize.

Several interrelated issues affect the design of a quantizer: The type of quantizer involved (scalar or vector), the distortion measure used, the inclusion or exclusion of memory in the quantizer, the search complexity, the codebook design, the desired number of bits, memory requirements for storing the codebook, robustness to a wide class of inputs and robustness to channel errors. A study of all these issues is beyond the scope of this paper. We focus our investigation on the issue of quantizer robustness to various filtering and noise conditions. Our objective is to realize a quantization scheme that results in a prescribed level of distortion, with the lowest possible number of bits, for a wide class of inputs, under the assumption that there are no channel errors.

The outline of the paper is as follows. In Section II, we elaborate on our objective and discuss each of the issues mentioned above. The effect of different filtering conditions on LSF behavior is qualitatively described in Section III. Results for a robust scalar quantizer and vector quantizer are given in Sections IV and V, respectively. Section VI describes a scheme involving two quantizers. In Section VII, a codebook adaptation algorithm is described. Section VIII discusses an optimal search strategy for finding the codevector. The results obtained by subjective tests of our quantizer in a coder are reported in Section IX. The search complexity is discussed in Section X. Section XI summarizes the conclusions.

II. Issues in Quantizer Design

The quantizer performance is evaluated by the spectral distortion (SD) which, in dB, is defined as

Manuscript received March 24, 1994; November 12, 1994. The associate editor coordinating the review of this paper and approving it for publication was Dr. Spiros Dimolitsas.

R. P. Ramachandran is with the CAIP Center, Department of Electrical Engineering, Rutgers University, Piscataway, NJ USA 08858. This work was done when the author was at AT&T Bell Labs on a fellowship from the Natural Sciences and Engineering Research Council of Canada.

M. M. Sondhi, N. Seshadri, and B. S. Atal are with AT&T Bell Laboratories, Murray Hill, NJ 07974 USA.

IEEE Log Number 9410230.

$$\mathrm{SD} = \sqrt{\frac{1}{B}\int_R [10\log(|A_q(e^{j2\pi f})|^2/|A(e^{j2\pi f})|^2)]^2\, df} \quad (1)$$

where A and A_q refer to the filters resulting from the original and quantized parameters, respectively. Throughout our study, the speech was sampled at 8 kHz, the region R taken to be the frequency band from 125 to 3400 Hz and B taken to be the bandwidth represented by R. It is generally accepted that a quantizer may be considered transparent if its average spectral distortion (AVSD) is about 1 dB, the percentage of Type 1 outliers (SD from 2–4 dB) is less than 2% and there are no Type 2 outliers (SD greater than 4 dB) [6]. For the codebook design, we use a computationally more tractable measure than the SD. The measure is the weighted squared Euclidean distortion given by

$$d(\boldsymbol{f}, \hat{\boldsymbol{f}}) = \sum_{i=1}^{p} w(i)[f(i) - \hat{f}(i)]^2, \quad (2)$$

where $f(i)$ is the ith component of the original LSF vector $\boldsymbol{f}$ and $\hat{f}(i)$ is the corresponding component of the quantized vector $\hat{\boldsymbol{f}}$. The weights are chosen to be [7]

$$w(i) = \frac{1}{f(i) - f(i-1)} + \frac{1}{f(i+1) - f(i)}. \quad (3)$$

These weights add some emphasis to the formant frequencies thus making the measure d correspond better to the SD than the unweighted squared Euclidean distortion. The codebook design method is based on the Linde-Buzo-Gray (LBG) algorithm with binary splitting [8] in which an input set of training data is used to determine the codewords such that the expected distortion is minimized. Note that in comparison to the SD, the determination of the Voronoi cells and the centroid computation is much simpler for the weighted squared Euclidean distortion thereby making it computationally feasible. Although the weighting is input dependent, the centroid is computed as the average of the training vectors in the Voronoi region so as to preserve the ordering property.

Since the focus of our study is on robustness to various conditions, we examine the relationship of robustness to the type of quantizer and number of bits used. The advantage of vector quantizers over scalar quantizers is that they require fewer bits to achieve a given level of distortion by exploiting the multidimensional probability density and allowing for Voronoi regions of arbitrary convex shape [9]. Since the true probability density of the LSF's derived from all possible speech material is not known, a vector quantizer design is based on a large set of empirical training data. Due to the sampling variability of the training set, there is an incomplete description of the multidimensional probability density. Therefore, the vector quantizer is sensitive to the distribution of the data on which it is tested thereby diminishing its robustness. In contrast, a scalar quantizer is configured by designing codebooks for each dimension separately. It can adequately cover a designated multidimensional space. Therefore, it performs well even for data with a distribution different from that of the training set. However, to achieve this robustness, the scalar quantizer requires more bits than a vector quantizer.

Our goal is to design a quantization scheme that compromises between the conflicting requirements of using few bits and robustness. A scalar quantizer that results in about a 1 dB AVSD for various test data serves as a benchmark. Then, we develop a quantizer with as few bits as possible and with a performance no worse than that of the scalar quantizer. The performance is evaluated in terms of AVSD and number of Type 1 and Type 2 outliers. Note that we attach considerable importance to reducing the number of outliers especially if the AVSD is already about 1 dB or less.

Our quantization scheme is memoryless, i.e., each vector is coded independently of past or future actions of the encoder or decoder [10]. Finally, search complexity and memory requirements are important practical considerations. Although we do not focus on these, neither will be made prohibitively large.

III. Parameter Variations

Since our aim is to get a robust quantizer, we first study the effects of various practically reasonable filtering conditions on the behavior of LSF's. This will give us information about the dynamic range of each LSF and the boundaries of the 10-dimensional space that have to be considered for the design. The LPC analysis is performed every 20 ms by the modified covariance method [11] with error weighting [12] by a 25 ms Hamming window, a high frequency compensation factor of 0.05 and a bandwidth expansion of 10 Hz. The speech is taken from the TIMIT database and the data comes from 518 different sentences spoken by 209 speakers.

The TIMIT database consists of speech sampled at 16 kHz. We study the effects of applying 3 different lowpass filters (with cutoff frequencies at 3600, 3400, and 3200 Hz, respectively) to the speech, followed by downsampling by 2. It is observed that as the cutoff frequency decreases, $f(9)$ and $f(10)$ shift downward and the rest remain essentially the same. After lowpass filtering to 3400 Hz and downsampling by 2, we study the effect of applying 3 additional filter weightings on LSF behavior. Applying a 300 Hz highpass filter is equivalent to examining the limitation of telephone bandwidth. In this case, the major influence is on $f(1)$ which shifts upward. Also $f(2)$ and $f(3)$ shift upward but to a lesser extent than $f(1)$. The other LSF's do not change. Processing speech by a bandpass filter that conforms with the Intermediate Reference Mask (IRS), which is a CCITT standard, has a different impact. The upward shift of the first 3 LSF's is more than with the highpass filter. The frequencies $f(4), f(5), f(6)$, and $f(7)$ shift upward but not as much as the first 3 LSF's. The last two LSF's shift downward while $f(8)$ remains the same. We do a z to $-z$ transformation on the IRS filter to get another bandpass filter to observe the effect of different spectral tilts on the LSF's. Compared to the standard IRS filter, this other bandpass filter causes a larger upward shift of $f(1)$ and $f(2)$, about the same shift for $f(3)$ and a smaller upward shift of $f(4)$. There are practically no shifts for $f(5)$ through $f(8)$. The

downward shift of $f(9)$ is about the same and the downward shift in $f(10)$ is less than for the IRS filter.

The above observations suggest the following:

1) An acceptable vector quantizer designed on training data from just one filtering condition (e.g., the 3400 Hz lowpass filter) may not work well for inputs from another condition (e.g., the IRS filter) because of the shifting behavior of many LSF's.
2) Even two bandpass filters related by a simple frequency transformation but having different spectral tilts influence the joint LSF distribution in different ways.
3) Since the dynamic ranges of some individual LSF's are affected by highpass and bandpass filter weighting, the overall 10-dimensional space that has to be considered for codebook design is larger than the space described for LSF's derived from 3400 Hz lowpass filtered speech only. This indicates that more bits are needed to accommodate the filter weightings.

Differential line spectral frequencies (DLSF) are defined as follows: $\Delta(1) = f(1)$ and $\Delta(i+1) = f(i+1) - f(i)$ for $i = 1$ to $p-1$. These parameters (except $\Delta(1)$) have the empirically observed property that the filtering conditions described above do not change their dynamic ranges. Also, the standard deviations of the DLSF's are generally lower than those of the LSF's. In view of this, the DLSF's are more suited to quantization. However, DLSF's have potential advantages only for scalar quantization. Vector quantization of DLSF's yields no advantage over that of LSF's as discussed later.

IV. Description of Scalar Quantizer

In order to assess the savings in bits offered by our approach, we first design a benchmark scalar quantizer for LSF's by the LBG method. The training data consists of 333 355 LSF vectors collected from the TIMIT database. They correspond to equal contributions from 5 conditions, namely, 1) the lowpass filter at 3400 Hz, 2) the lowpass filter at 3200 Hz, 3) the 300 Hz highpass filter, 4) the IRS filter, and 5) the transformed IRS filter. Two lowpass filters are used to provide some emphasis to the relatively lower values of the first 3 LSF's. The weightings (3)–(5) are applied after lowpass filtering to 3400 Hz and downsampling by 2. Let $f_{\min}(i)$ and $f_{\max}(i)$ denote the smallest and largest values of the ith LSF determined from the 333 355 data points. The overall space S is described by the cross-product of the intervals $[f_{\min}(i), f_{\max}(i)]$ under the usual constraint of ordering. The training set consisting of the 333 355 data points leads to a scalar quantizer that accomplishes a coverage of the space S. The codebook for each $f(i)$ is designed independently of the others.

The performance of the quantizer is evaluated by five sets of test data as follows:

1) 14 780 vectors from the 5 filtering conditions but different from the training set.
2) 14 780 vectors from LPC analysis of speech corrupted by additive white Gaussian noise with an SNR of 15 dB.
3) 14 780 vectors from LPC analysis of speech corrupted by car noise (at a speed of 120 km/h) with an SNR of 15 dB.

TABLE I
Performance Results for 36 Bit Scalar Quantizer

Condition	AVSD (dB)	Type 1 Outliers (%)	Type 2 Outliers (%)
Different filters	0.99	2.17	0.03
White noise	0.80	0.13	0
Car noise	0.92	1.10	0.01
Babble noise	0.96	1.62	0.02
Random LSFs	1.16	6.51	0.13

4) 14 780 vectors from LPC analysis of speech corrupted by multitalker babble noise with an SNR of 30 dB.
5) 10 000 randomly generated LSF vectors.

For the first four items, 18 sentences of speech from 18 speakers (1 sentence for each speaker) from the TIMIT database are used. They are different from the ones used to derive the LSF's that train the quantizer. The noise is added to the original 16 kHz sampled speech and filtered in the same way as the speech such that the resulting SNR (as mentioned above) is obtained just prior to LPC analysis.

To generate the test data for item 5, the procedure is as follows. We first transform the training data vectors to log-area ratio vectors, and compute the mean and variance of each component. To get one random log-area ratio vector, we use a Gaussian random number generator for each component with the computed mean and variance. Finally, we transform this log-area ratio vector into the corresponding random LSF vector.

A 36 bit scalar quantizer achieves about a 1 dB AVSD for the test conditions. The bit allocation is (4, 4, 4, 4, 4, 4, 3, 3, 3, 3). Table I shows the performance results. These results were obtained by doing a sequential search for the best codebook entry for each LSF separately such that ordering is preserved. Better search strategies are discussed later.

As seen from Table I, the scalar quantizer works reasonably well on a variety of input vectors. The outliers are mainly due to clipping of one or more LSF components. For speech degraded by noise, a somewhat better performance is achieved than for LSF's obtained from noiseless speech. This is because additive noise generally moves each $f(i)$ towards a region in $[f_{\min}(i), f_{\max}(i)]$ that is more densely populated by the training data, and at the same time reduces its variance. The case of lowest variance and best performance is for speech with white noise. The poorest performance is, as expected, for random LSF's. However, even for these, the resulting AVSD is only slightly higher than for the LSF's obtained from the various filtering conditions.

A 34 bit DLSF scalar quantizer achieves about the same performance as the 36 bit LSF quantizer for all the testing conditions. The bit allocation is (4, 4, 4, 4, 3, 3, 3, 3, 3, 3). The sequential search strategy is as outlined in [4]. The first DLSF is $\Delta(1) = f(1)$. It is quantized to $\hat{\Delta}(1) = \hat{f}(1)$. Then, for $i = 1$ to $p-1$, the value

$$\Delta_r(i+1) = f(i+1) - \hat{f}(i) \tag{4}$$

is quantized to $\hat{\Delta}(i+1)$. The reconstructed LSF's are given by

$$\hat{f}(i+1) = \hat{f}(i) + \hat{\Delta}(i+1) \tag{5}$$

for $i = 1$ to $p-1$. The design of a globally optimum scalar quantizer based on dynamic programming (as opposed to a

locally optimum solution provided by the LBG method) is dealt with in [13], [14].

V. Use of Vector Quantization

In the previous section, we saw that scalar quantization of DLSF's saves about 2 bits compared to scalar quantization of LSF's, for the same performance. Also, as pointed out earlier, the various filtering conditions do not appear to alter the distribution of the DLSF's. It appears to be natural, therefore, to consider a vector quantizer for DLSF's. However, unless the DLSF's are quantized sequentially, (see (4) and (5)), quantization errors propagate to the higher LSF components and yield a large spectral distortion. Inspection of this sequential quantization procedure shows that, except for a renaming of the components, vector quantization of DLSF's reduces precisely to the vector quantization of LSF's. We show this equivalence in detail in the Appendix, for codebooks constructed with the LBG algorithm, based on the distortion measure d (see (2)). When the sequential procedure is used as part of the LBG algorithm, the codebook for the DLSF's is shown to be equivalent to that for the LSF's. Thus, for vector quantization there is no advantage in going to DLSF's.

It is clear from our discussion in Sections II and III that robustness is not guaranteed for a vector quantizer even if the same training data is used as for a scalar quantizer. The reason is that the LBG algorithm (or any other reasonable algorithm for that matter) allocates large numbers of codevectors to regions of S that are densely populated by the training data, and very few to the sparse regions. This gives rise to a large spectral distortion whenever a *test* vector occurs in one of the sparse regions. This problem can be alleviated by configuring a training set with two components. The first component is the set of LSF vectors corresponding to the 5 filtering conditions discussed in Section IV. This data provides the empirical estimate of the probability density. The second component is a set of uniformly distributed vectors in S (which we will call a "uniform sheet"). This set increases robustness by covering the sparse regions of the first component. The relative proportion of vectors from these two components is determined experimentally to get the best performance.

It is, of course, not feasible to design a single codebook of more than 12 bits because of the prohibitively large memory and computational requirements. Two suboptimum approaches are the multistage vector quantizer [15], [16] and the split vector quantizer [6]. For both these methods, the codebooks at each stage have fewer bits than when using a single codebook. In the multistage technique, a given test vector is quantized with the first codebook. For the remaining stages, the quantization error from the previous stage is quantized with its particular codebook. The final quantized version of the test vector is obtained by summing the codevectors of all the stages. Note, however, that the final quantized vector is not guaranteed to satisfy the ordering property. Also, it is not feasible to generate uniform sheets to adequately cover 10-dimensional regions. For LSF vectors, therefore, we are forced to use a special case of a multistage vector quantizer, called a split vector quantizer. Here, the codebook at each stage quantizes only a subvector. We use a (3,3,4) split for our 10-dimensional vectors. (i.e., the first stage quantizes (f_1, f_2, f_3), the second stage quantizes (f_4, f_5, f_6) and the last stage quantizes (f_7, f_8, f_9, f_{10})). Clearly with this scheme, it is possible to enforce ordering of the quantized vectors. The ordering property also allows us to use codebook adaptation which is discussed later. Finally, the reduced number of dimensions for each codebook makes it feasible to generate the uniform sheets mentioned in the previous paragraph.

TABLE II
Performance Results for 32 Bit Vector Quantizer Using Random LSF's

Fraction of uniform sheet	AVSD (dB)	Type 1 Outliers (%)	Type 2 Outliers (%)
0.0	1.18	9.27	0.16
0.1	1.16	5.28	0.01
0.2	1.18	3.99	0.0
0.3	1.18	3.12	0.0
0.4	1.20	3.07	0.01
0.5	1.22	2.55	0.01
0.6	1.23	2.49	0.0
0.7	1.26	2.12	0.0
1.0	1.49	6.36	0.0

TABLE III
Performance of Vector Quantizers for LSF's Obtained from Speech

Condition	Lowest bit rate	Fraction of uniform sheet	AVSD (dB)	Type 1 Outliers (%)	Type 2 Outliers (%)
Different filters	27	0.0	1.02	1.31	0.0
White noise	28	0.0	0.82	0.07	0.0
Car noise	30	0.1	0.95	0.51	0.0
Babble noise	28	0.0	0.98	0.77	0.0

We evaluate the performance of the different vector quantizers designed by using different proportions of the two training data components. A sequential search is used to find the best codebook entry for each subvector subject to the ordering constraint. By far the most stringent test condition is that of the random LSF's. Table II shows the results of experimentation with different proportions of the uniform sheet for a 32 bit quantizer for this test condition. The bit allocation for the three subvectors was chosen to be (11, 10, 11). From Table II, it is seen that a fraction of 0.6 for the uniform sheet results in the fewest number of outliers. In comparison to the 36 bit scalar quantizer, a matching AVSD and significantly fewer outliers are obtained for fractions of 0.2 and 0.3. For this condition, we clearly cannot go less than 32 bits.

Table III summarizes the results of similar experimentation with the other testing conditions. When tested on LSF's obtained from the 5 filtering conditions, additive white noise and babble noise, it turns out that the best performance is obtained when *no* uniform sheet is used. The criteria for transparent quantization (see Section II) are met and the performance is no worse than that of the scalar quantizer. For the filtering conditions alone, we need only 27 bits. An additional bit is needed for white noise and babble noise. For the case of car noise, the best performance is obtained if 10% of the training data is from the uniform sheet. A 30 bit codebook is required.

To achieve the performance of the 36 bit scalar quantizer, we need 32 bits if we design the codebook in the manner outlined above. Of course, the procedure of introducing the uniform sheet may not be the best way of providing coverage of sparse regions of the training data. Indeed, as we show in

TABLE IV
PERFORMANCE RESULTS FOR 30 BIT TWO-QUANTIZER FORMAT USING RANDOM LSF'S

Fraction of uniform sheet	AVSD (dB)	Type 1 Outliers (%)	Type 2 Outliers (%)	Vector quantizer selected (%)
0.0	1.15	4.56	0.01	54
0.1	1.16	4.07	0.01	50
0.2	1.18	3.92	0.0	46
0.3	1.18	3.55	0.0	45
0.4	1.20	3.81	0.0	41
0.5	1.21	3.94	0.01	38
0.6	1.22	4.03	0.0	36
0.7	1.23	3.83	0.0	33
1.0	1.28	5.53	0.0	19

TABLE V
PERFORMANCE OF 30 BIT TWO-QUANTIZER FORMAT FOR LSF'S OBTAINED FROM SPEECH. THE UNIFORM SHEET CONSTITUTES 20 OR 30% OF THE TRAINING DATA FOR THE VECTOR QUANTIZER

Condition	Fraction of uniform sheet	AVSD (dB)	Type 1 Outliers (%)	Type 2 Outliers (%)	Vector quantizer selected (%)
Different filters	0.2	1.01	0.64	0.0	67
Different filters	0.3	1.03	0.76	0.0	62
White noise	0.2	0.86	0.01	0.0	73
White noise	0.3	0.89	0.05	0.0	68
Car noise	0.2	0.99	0.77	0.0	61
Car noise	0.3	1.00	0.38	0.0	58
Babble noise	0.2	0.99	0.45	0.0	65
Babble noise	0.3	1.01	0.54	0.0	60

the next section, a scheme using two quantizers accomplishes the coverage more efficiently.

VI. TWO-QUANTIZER FORMAT

Another way to improve coverage of sparse regions of the training data is to use *both* a vector quantizer *and* a scalar quantizer. The number of bits for each quantizer is the same and one additional bit specifies which quantizer is used. The encoding algorithm for each input vector is as follows:

1) Find the codeword for the vector quantizer that minimizes $d(\boldsymbol{f}, \hat{\boldsymbol{f}})$.
2) Find the codeword for the scalar quantizer that minimizes $d(\boldsymbol{f}, \hat{\boldsymbol{f}})$.
3) Select the codeword that results in the lowest SD.

The training data for the vector quantizer is as before. The 333 355 vectors obtained from the 5 filtering conditions are used to train the DLSF scalar quantizer.

We discuss next the performance of a 30 bit scheme (29 bits for each quantizer +1 bit to specify the selected quantizer). For the vector quantizer, the bit allocation is (10, 9, 10). In the case of the scalar quantizer, the bit allocation is (3, 3, 3, 3, 3, 3, 3, 3, 3, 2). By examining the quantized vectors, some benefits of this scheme are understood. The two quantizers can complement each other. The vector quantizer deals better with vectors whose components are clipped by the scalar quantizer. The scalar quantizer can be better for inputs in regions of the space that are otherwise sparsely covered by the vector quantizer codebook. Using two quantizers has the added advantage that the final selection can be based directly on the SD. Generally, this yields a slight reduction in the SD and can reduce the number of Type 1 outliers by 0.5%.

Table IV shows the performance of this scheme on the random LSF's as the testing condition. The 30 bit two-quantizer scheme outperforms the 36 bit scalar quantizer even when no uniform sheet is used. However, using a sheet (fraction of 0.3) leads to the fewest number of outliers and about the same AVSD as the 36 bit scalar quantizer. The next best situation is when the fraction of the sheet is 0.2. When the fraction of the uniform sheet is 0.2 or 0.3, the 30 bit scheme is comparable to a 32 bit vector quantizer. It is interesting to note from the last column of Table IV that the scalar quantizer is chosen for quite a large percentage of the test data.

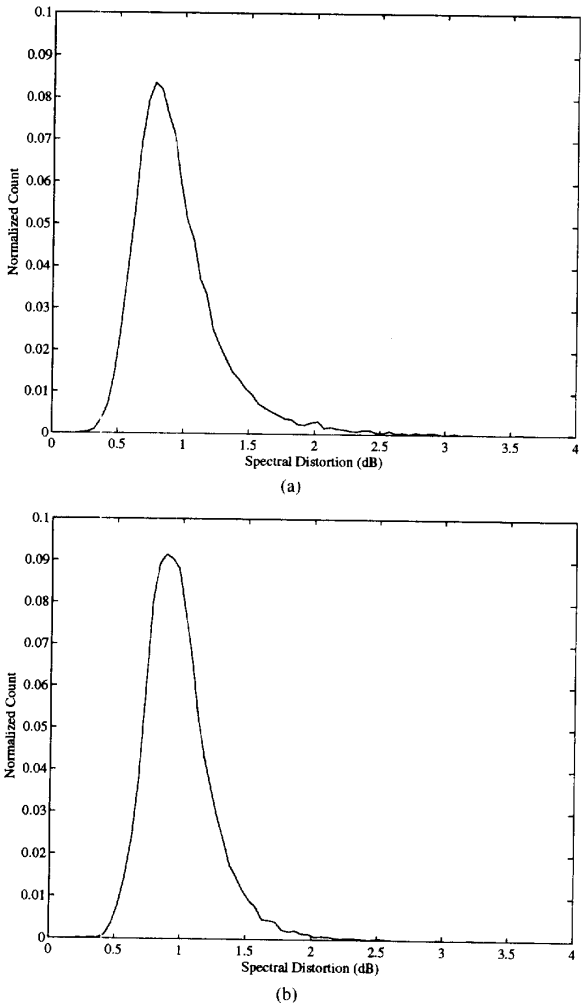


Fig. 1. Probability density of spectral distortion for the case of babble noise. In our scheme, a 20% sheet is used.

Table V shows the performance for the other testing conditions when the uniform sheet comprises 20% and 30% of the training data. It is seen that the two conditions are almost equivalent. The fraction 0.2 is better for most of the conditions. For both the fractions, the criteria for transparent quantization are met. Compared to the 36 bit scalar quantizer, the number of outliers are much less at the expense of a higher AVSD. However, the latter difference is not as significant since the AVSD is already 1 dB or less. Fig. 1 shows the probability density of the spectral distortion for a 36 bit scalar quantizer and our 30 bit scheme (20% sheet) when babble noise is the testing condition.

VII. Codebook Adaptation

The investigation continues by attempting to further lower the bit rate without sacrificing performance. Therefore, we no longer do a comparison with the 36 bit scalar quantizer. Rather, the results in Tables IV and V serve as the benchmark. The (3, 3, 4) split vector quantization described above does not take advantage of any dependence among the subvectors. As far as the design of the codebooks is concerned, it is not easy to take advantage of this dependence. The codebook for each subvector has to be designed independently of the others because a joint design is not feasible. However, in the quantization of a given vector it is possible to capitalize on the dependence. The basic idea is as follows. Suppose the ith subvector is quantized to $\hat{\boldsymbol{f}}_{\boldsymbol{i}}$. Then the ordering property of LSF's restricts the region in which the $(i+1)$st quantized vector $\hat{\boldsymbol{f}}_{\boldsymbol{i+1}}$ may lie. Rather than restricting the search of the $(i+1)$st codebook to the allowed region, the entire codebook may be mapped to that region, thus providing a finer sampling of it. Note that if the mapping is completely specified by the quantized vector $\hat{\boldsymbol{f}}_{\boldsymbol{i}}$, there is no cost in terms of bits. Also, since the mapping starts with one fixed codebook, there is no overhead in terms of memory requirements. This idea, called codebook adaptation, was first proposed for a scalar quantizer in [14]. Our method is a multidimensional generalization of this approach. The complete procedure is outlined below.

Recall that the ith LSF is in the interval $[f_{\min}(i), f_{\max}(i)]$ (see Section IV). Suppose there are N subvectors $\boldsymbol{f}_i$ of dimension d_i to be quantized by N corresponding codebooks C_i. Then, the quantized vector is $\hat{\boldsymbol{f}} = [\hat{\boldsymbol{f}}_{\boldsymbol{1}}|\hat{\boldsymbol{f}}_{\boldsymbol{2}}|\cdots|\hat{\boldsymbol{f}}_{\boldsymbol{N}}]$. The subvector $\boldsymbol{f}_{\boldsymbol{1}}$ is quantized to $\hat{\boldsymbol{f}}_{\boldsymbol{1}} = [\hat{f}(1)\hat{f}(2)\cdots\hat{f}(d_1)]$ without any adaptation. Let the jth entry of C_2 be $\hat{\boldsymbol{f}}_{\boldsymbol{j,2}} = [\hat{f}_j(d_1+1)\hat{f}_j(d_1+2)\cdots\hat{f}_j(d_1+d_2)]$. For each j, it must be ensured that $\hat{f}_j(d_1+1) > \hat{f}(d_1)$. If $\hat{f}(d_1) < f_{\min}(d_1+1)$, this naturally holds and no adaptation is performed. Otherwise, we transform codebook C_2 to C_2^t such that C_2^t covers a smaller space that is, in general, not similar in shape to the space covered by C_2. Our transformation maps each component of the jth entry separately. The mapped first component $\hat{f}_j^t(d_1+1)$ is given by

$$\hat{f}_j^t(d_1+1) = \frac{[(\hat{f}_j(d_1+1) - f_{\min}(d_1+1)][f_{\max}(d_1+1) - \hat{f}(d_1)]}{f_{\max}(d_1+1) - f_{\min}(d_1+1)} + \hat{f}(d_1). \tag{6}$$

Note that this is the mapping used in [14] for the scalar quantizer. Successive components $\hat{f}_j(d_1+r)$ for $r=2$ to d_2 are sequentially mapped as outlined below. Regarding the original codebook C_2, let

$$f_m(d_1+r) = \begin{cases} f_{\min}(d_1+r) \\ \quad \text{if } f_{\min}(d_1+r-1) \le \hat{f}_j(d_1+r-1) \\ \quad\quad \le f_{\min}(d_1+r) \\ \hat{f}_j(d_1+r-1) \\ \quad \text{if } \hat{f}_j(d_1+r-1) > f_{\min}(d_1+r). \end{cases} \tag{7}$$

For the transformed codebook C_2^t, let

$$f_m^t(d_1+r) = \begin{cases} f_{\min}(d_1+r) \\ \quad \text{if } f_{\min}(d_1+r-1) \le \hat{f}_j^t(d_1+r-1) \\ \quad\quad \le f_{\min}(d_1+r) \\ \hat{f}_j^t(d_1+r-1) \\ \quad \text{if } \hat{f}_j^t(d_1+r-1) > f_{\min}(d_1+r). \end{cases} \tag{8}$$

Then, the mapping is given by (9) shown at the bottom of the page.

Note the dependence on the previous component of the original codeword and the mapped codeword. This is needed to maintain the ordering property of the entries of C_2^t. The next subvector $\boldsymbol{f}_{\boldsymbol{2}}$ is quantized to $\hat{\boldsymbol{f}}_{\boldsymbol{2}}$ using C_2^t. Then, C_3 is mapped in a similar fashion as outlined above for the quantization of $\boldsymbol{f}_{\boldsymbol{3}}$. The process continues until all the N subvectors are quantized.

In implementing the codebook transformation, we have considered two options. Option 1 is to map every codeword of C_i $(i=2$ to $N)$.This is based on the heuristic claim that the conditional probability density $p[f(\Sigma_{k=1}^{i-1} d_k + 1), \cdots, f(\Sigma_{k=1}^{i} d_k) | \hat{f}(\Sigma_{k=1}^{i-1} d_k)]$ is similar in shape to the probability density $p[f(\Sigma_{k=1}^{i-1} d_k + 1), \cdots, f(\Sigma_{k=1}^{i} d_k)]$. Option 2 is to keep the codewords of C_i that maintain the ordering property intact and only map the other codewords. This guarantees a lower distortion for any input vector if $N=2$. Experiments show that for our vector quantizers in which $N=3$, the performance is much better if Option 2 is invoked.

Consider the case of a 30 bit vector quantizer alone in which the fraction of the uniform sheet is either 0.2 or 0.3. For LSF's taken from speech, the AVSD consistently decreases by about 0.02 dB. Depending on the quantizer and the testing condition, the ratio of the number of outliers with and without adaptation ranges from 0.33–0.91 (average value is 0.57). In the case of random LSF's, the AVSD decreases by 0.09 dB and there are 0.58 as many outliers. Table VI gives the results for a 30 bit two-quantizer system with adaptation. The consequences of adaptation are more apparent in terms of reducing outliers than in diminishing the AVSD. Note that although adaptation improves performance, the number of bits is not lowered.

$$\hat{f}_j^t(d_1+r) = \frac{[(\hat{f}_j(d_1+r) - f_m(d_1+r)][f_{\max}(d_1+r) - f_m^t(d_1+r)]}{f_{\max}(d_1+r) - f_m(d_1+r)} + f_m^t(d_1+r) \tag{9}$$

TABLE VI
PERFORMANCE OF 30 BIT TWO-QUANTIZER FORMAT WITH ADAPTATION. THE UNIFORM SHEET CONSTITUTES 20 OR 30% OF THE TRAINING DATA FOR THE VECTOR QUANTIZER

Condition	Fraction of uniform sheet	AVSD (dB)	Type 1 Outliers (%)	Type 2 Outliers (%)	Vector quantizer selected (%)
Different filters	0.2	0.99	0.39	0.0	70
Different filters	0.3	1.01	0.42	0.0	66
White noise	0.2	0.85	0.01	0.0	77
White noise	0.3	0.87	0.02	0.0	72
Car noise	0.2	0.98	0.64	0.0	64
Car noise	0.3	0.99	0.21	0.0	60
Babble noise	0.2	0.97	0.27	0.0	68
Babble noise	0.3	0.99	0.26	0.0	63
Random LSFs	0.2	1.14	2.70	0.0	53
Random LSFs	0.3	1.14	2.31	0.0	52

TABLE VII
PERFORMANCE OF 29 BIT TWO-QUANTIZER FORMAT WITH ADAPTATION AND HYBRID SEARCH TECHNIQUE. THE UNIFORM SHEET CONSTITUTES 20 OR 30% OF THE TRAINING DATA FOR THE VECTOR QUANTIZER

Condition	Fraction of uniform sheet	AVSD (dB)	Type 1 Outliers (%)	Type 2 Outliers (%)	Vector quantizer selected (%)
Different filters	0.2	1.03	0.33	0.0	61
Different filters	0.3	1.04	0.32	0.0	56
White noise	0.2	0.88	0.01	0.0	71
White noise	0.3	0.91	0.01	0.0	66
Car noise	0.2	1.02	0.51	0.0	55
Car noise	0.3	1.02	0.21	0.0	51
Babble noise	0.2	1.01	0.21	0.0	58
Babble noise	0.3	1.02	0.25	0.0	53
Random LSFs	0.2	1.15	2.35	0.0	41
Random LSFs	0.3	1.15	1.82	0.0	41

VIII. DYNAMIC PROGRAMMING SEARCH

The results generated so far are obtained by a sequential search. For a split vector quantizer, a sequential search yields the optimal codevector if the subvectors are independent. However, for ordered data (like LSF's), where $\hat{\boldsymbol{f}}_{\boldsymbol{i}+1}$ depends on $\hat{\boldsymbol{f}}_{\boldsymbol{i}}$, this is not true. The search is suboptimal even with codebook adaptation, because the transformed codebook C_{i+1}^t is different for each entry of C_i^t. We use a dynamic programming technique called the A^* algorithm [18] to get the optimal codevector for a given split vector quantizer. The A^* algorithm can be used with or without codebook adaptation. We illustrate the technique when codebook adaptation is used.

1) Initialization: We have a split vector quantizer with N codebooks C_i each having a size n_i and dimension d_i. The input vector is $\boldsymbol{f} = [f(1)f(2)\cdots f(p)]$ where $p = \Sigma_i^N\ d_i$. The subvectors of $\boldsymbol{f}$ are denoted by $\boldsymbol{f_i}$. The jth entry of C_i is $\hat{\boldsymbol{f}}_{\boldsymbol{j,i}} = [\hat{f}_j(\Sigma_{r=1}^{i-1}\ d_r + 1)\cdots\hat{f}_j(\Sigma_{r=1}^{i}\ d_r)]$. The corresponding entry of C_i^t is $\hat{\boldsymbol{f}}_{\boldsymbol{j,i}}^{\boldsymbol{t}} = [\hat{f}_j^t(\Sigma_{r=1}^{i-1}\ d_r + 1)\cdots\hat{f}_j^t(\Sigma_{r=1}^{i}\ d_r)]$.
2) Quantize the first subvector $\boldsymbol{f_1}$ with every entry of C_1. Arrange the results in a stack. The jth element of the stack corresponds to the jth entry of C_1. The path length is $p(j) = 1$, the accumulated distortion is $d_q(j) = \Sigma_{i=1}^{d_1}\ w(i)[f(i) - \hat{f}_j(i)]^2$ and the index matrix entry is $I(j,1) = j$. The number of stack elements is $n_{\text{tot}} = n_1$.
3) Given n_{tot} stack elements, find the stack element k corresponding to the smallest accumulated distortion subject to a positive path length.
4) If $p(k) = N$, the algorithm terminates as the encoding is complete.
5) Now, $p(k) = m < N$. Depending on the quantized subvector from the transformed codebook C_m^t, adapt C_{m+1} to C_{m+1}^t. Quantize $\boldsymbol{f_{m+1}}$ using each entry of C_{m+1}^t. Accumulate the stack with n_{m+1} entries. For these entries ($l = n_{\text{tot}}+1$ to $n_{\text{tot}}+n_{m+1}$ corresponding to the indices $j = 1$ to n_{m+1}), $p(l) = m+1, I(l,i) = I(k,i)$ for $i = 1$ to $m, I(l,m+1) = j$. The accumulated distortion is $d_q(l) = d_q(k) + \Sigma_{i=u}^{v}\ w(i)[f(i) - \hat{f}_j^t(i)]^2$ where $u = \Sigma_{r=1}^{m}\ d_r + 1$ and $v = \Sigma_{r=1}^{m+1}\ d_r$. Update the value of n_{tot}. Set $p(k) = 0$. Go back to step 3.

The A^* method has been successfully applied to the scalar quantization of DLSF's [19]. However, due to the mismatch between the weighted squared Euclidean distortion and the SD, a hybrid scheme is suggested in [19]. For any input vector, there are two possible codevectors, one from a sequential search and one from an A^* search. The codevector that yields the smallest SD is chosen. We use the hybrid technique for both the vector and scalar quantizers to select the best codevector in terms of SD among four possibilities. The first two are the codewords from the vector quantizer resulting from a sequential and A^* search. The same procedure is repeated for the scalar quantizer to get the other two possibilities. Table VII shows the results for the resulting 29 bit scheme. For the vector quantizer, the bit allocation is (10, 9, 9). In the case of the scalar quantizer, the bit allocation is (3, 3, 3, 3, 3, 3, 3, 3, 2, 2).

Compared to a 30 bit system with no adaptation (see Tables IV and V), the AVSD is about the same and the number of outliers are reduced. The A^* algorithm offers much more of an improvement for the scalar quantizer than for the vector quantizer. This leads to an increased usage of the scalar quantizer for all the testing conditions as seen in Tables VI and VII. Transparent quantization is achieved by a 29 bit system for LSF's obtained from speech. The results are generally about the same whether we use a 20 or 30% uniform sheet to train the vector quantizer. However, note that the 30% sheet is better for random LSF's and can bring the number of outliers below 2%. The combination of codebook adaptation and the A^* algorithm results in a savings of 1 bit. Fig. 2 shows the probability density of the spectral distortion using the two-quantizer scheme (30% sheet) when tested on the filtering conditions. In Fig. 2(a), 30 bits are used with no adaptation and a sequential search. In Fig. 2(b), 29 bits are used with adaptation and a dynamic programming search.

IX. SUBJECTIVE TESTING

The results in Table VII show that our 29 bit system can achieve transparent quantization. We further tested the scheme in a coder. Ideally, we would like to compare synthesized speech signals that differ only due to LPC quantization, and demonstrate that there is no perceptible difference. However, in existing predictive coders operating at rates below 8 kb/s, the coding distortion is not confined to that of LPC quantization. Specifically, consider Code Excited Linear Prediction (CELP) [20]. The excitation signal and the LPC parameters both are subject to quantization distortion. However, the individual distortions are not independent. A coarse quantization

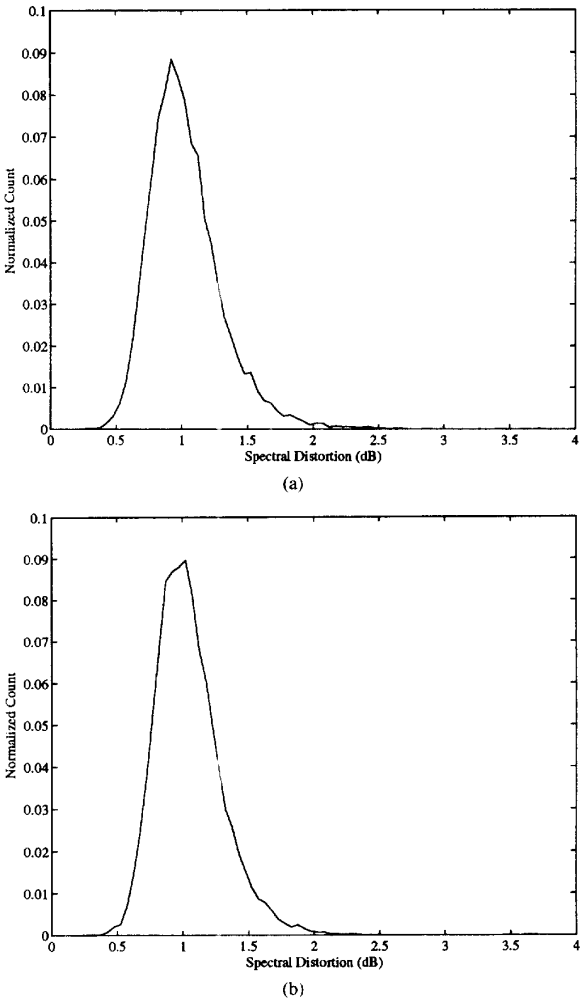


Fig. 2. Probability density of spectral distortion for the case of different filtering conditions. A 30% sheet is used.

of the LPC parameters, for instance, can be compensated for by a fine quantization of the excitation signal. There is no way of rigorously dealing with this problem. We alleviate it by using the CELP coder of [21]. In that coder the excitation is quantized at 4.8 kb/s. That is neither too coarse (which would give poor quality speech) nor too fine (which could lead to the compensation mentioned above). In implementing the adaptive codebook, we determined the best delay by an exhaustive search and used that value without doing any delay smoothing. The stochastic codebook contribution is adaptively controlled [22] to enhance the periodicity of the synthesized speech.

To study the effect of quantization, we generate two synthetic speech signals. Both use the coder of [21] with the LPC vector updated every 20 ms. The difference is that in one case, the LPC parameters are not quantized and in the other case, our 29 bit scheme (including adaptation and dynamic programming search) is used. In the latter case, the CELP coder operates at a bit rate of 6.25 kb/s. We compare the quantizers resulting from both the 20 and 30% sheets to the unquantized case. Sixteen sentences from the TIMIT database encompassing 8 male and 8 female speakers were processed. Eight different conditions were represented such that for each condition, one male and one female speaker were used. The five filtering conditions were included. The other three corresponded to noisy speech. The additive white Gaussian noise was filtered by the 3400 Hz lowpass filter only (SNR of 15 dB). The car and babble noise (SNR of 15 and 30 dB, respectively) were filtered by the IRS filter. Ten listeners participated in a pair comparison test. The two sets of decoded speech for each sentence were played in both orders. The ordering was randomized for each sentence. The participants had the option of indicating a preference or declaring no preference.

Consider the scheme in which the vector quantizer is designed with a 20% sheet. The mean preference for the CELP coder with quantized LPC parameters is 48.0%. The standard deviation is 5.3. The 95% confidence interval as calculated using the t distribution ranges from 44.0 to 52.0. When a 30% sheet is used, the mean preference for the coder with quantized parameters is 47.0%. The standard deviation is 4.5 and the 95% confidence interval ranges from 43.6–50.4. This shows that the coders are indistinguishable and the quantization does not contribute to audible distortion.

X. Search Complexity

We define the search complexity to be the total number of comparisons N_C between the input vector and the codevectors to obtain the quantized vector. For a b bit quantizer, an exhaustive search requires $N_C = 2^b$ comparisons, which for a 30 bit quantizer is clearly unfeasible. The use of a sequential search and a split vector quantizer alleviates the complexity although the optimal codevector is not necessarily obtained. Consider our 30 bit system. When there is no adaptation, the 29 bit (10, 9, 10) split vector quantizer involves a maximum of $2^{10} + 2^9 + 2^{10} = 2560$ comparisons. When adaptation is used, the number of comparisons always equals this maximum. For the 29 bit scalar quantizer, 76 comparisons are performed. An extra comparison dictates the type of quantizer used. Therefore, $N_C = 2637$ assuming adaptation is performed.

Although the 29 bit system gives a savings of 1 bit, a significantly higher complexity is needed to maintain the same performance. For the vector quantizer, a sequential search with adaptation involves 2048 comparisons. The complexity of the A^* algorithm is input dependent and hence, a random variable. Since the best codevector in terms of weighted squared Euclidean distortion is found, we want to show that the complexity is significantly lower than for an exhaustive search. In our implementation, if the number of comparisons exceeds 10^5, we revert to a sequential search. Fig. 3 shows the probability density of the number of comparisons for the vector quantizer (20% sheet) for two testing conditions. The first is for LSF's obtained due to different filters. The second is for random LSF's. For the filtering conditions, the average number of comparisons is 13 430 and the standard deviation is 14 977. The number of comparisons exceeds 10^5 about

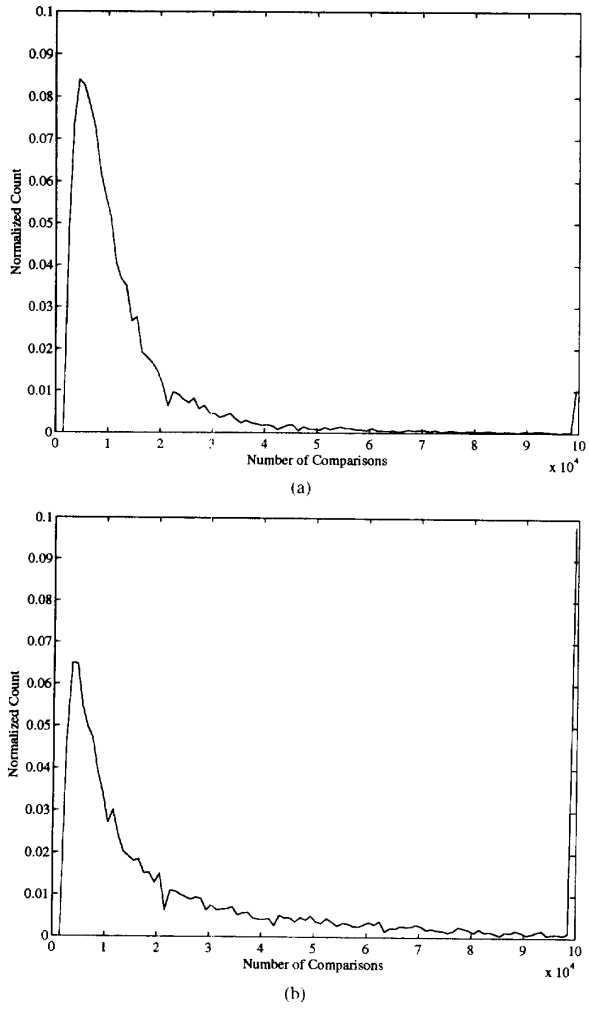


Fig. 3. Probability density of the number of comparisons for a 28 bit vector quantizer (20% sheet).

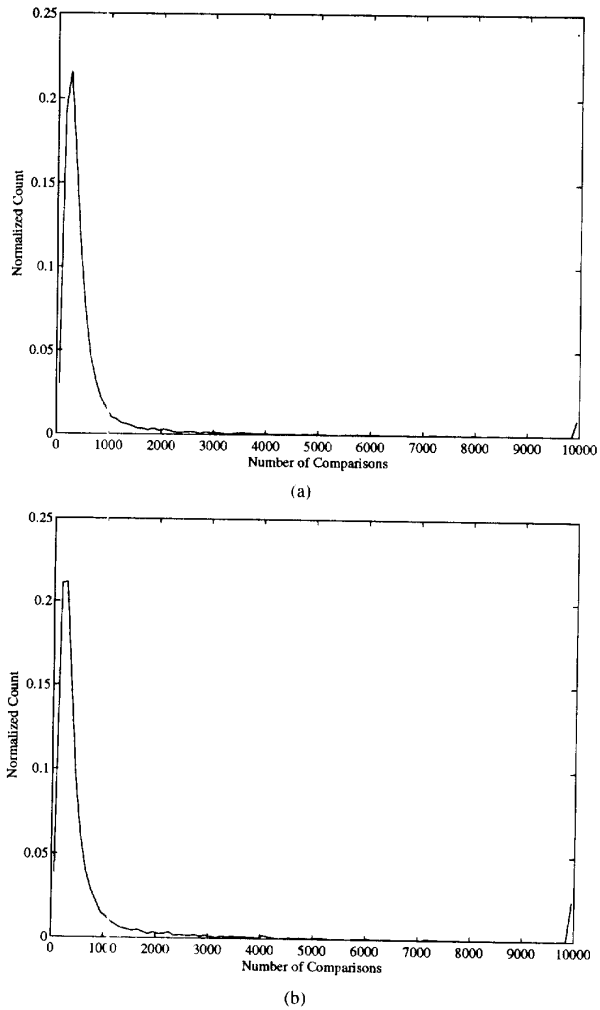


Fig. 4. Probability density of the number of comparisons for a 28 bit scalar quantizer.

1.02% of the time. In the case of random LSF's, the average number of comparisons is 28 138 and the standard deviation is 30 853. The number of comparisons exceeds 10^5 about 9.68% of the time. For both conditions, the search complexity is significantly less than for an exhaustive search. Since the training data includes the filtering conditions, the optimal path is more distinguishable from the competing paths when the testing also includes the same filtering conditions. This explains the higher search complexity for the random LSF's. Since the maximum number of comparisons is a significant issue, Fig. 3 suggests that an alternative implementation can lower the threshold below 10^5 for reverting to a sequential search. Further studies should examine the tradeoff between this threshold and performance.

For the scalar quantizer, a sequential search involves just 72 comparisons. In our implementation of the A^* algorithm, the number of comparisons again does not exceed 10^5. Fig. 4 shows the probability density of the number of comparisons for the scalar quantizer. For the filtering conditions, the average number of comparisons is 796 and the standard deviation is 3682. The number of comparisons exceeds 10^5 about 0.05% of the time. In the case of random LSF's, the average number of comparisons is 1573 and the standard deviation is 8386. The number of comparisons exceeds 10^5 about 0.47% of the time. Again, note the higher search complexity for the random LSF's. Also, the complexity is less than for a vector quantizer. Given that 4 comparisons are needed to determine the type of quantizer used, we add the average value for the A^* search to get $N_C = 16\,350$ (different filters) and $N_C = 31\,385$ (random LSF's).

The number of comparisons is directly related to a more precise quantity, namely, the number of arithmetic operations. The arithmetic complexity depends on the calculation of the distortion measure, the adaptation algorithm and the computation of the spectral distortion. Also, one comparison for the scalar quantizer involves less arithmetic complexity than for a vector quantizer thereby confirming the lower complexity the scalar quantizer offers. Consider a 30 bit

system with adaptation and a sequential search and a 29 bit system with adaptation and a dynamic programming search. The maximum number of comparisons for the 30 bit system is less than the average value for the 29 bit system when individually examining the vector and scalar quantizers. Also, the adaptation algorithm has to be usually applied more than once when an A^* search is used. Two more spectral distortion computations are needed for the 29 bit system. The 29 bit system is more complex in terms of arithmetic complexity. However, it is much better than an exhaustive search which would involve 2^{28} comparisons for each quantizer.

XI. Summary and Conclusions

In this paper, we have proposed a 30 bit quantization scheme that is robust to a wide variety of inputs. The performance is at least equal to that of a 36 bit scalar quantizer for various test conditions that include a set of randomly generated LSF's. The scheme is based on a two-quantizer format involving both a vector and a scalar quantizer. For each input, the quantizer that achieves the lower distortion is used.

The vector quantizer is designed with training data that has two components. The first component, consisting of LSF's derived from speech passed through various filters in common use, provides coding efficiency. The second component, consisting of a set of vectors uniformly distributed over the space S described by the LSF's in the first component, provides robustness. The scalar quantizer is designed with training data from the first component only. The two key factors in achieving robustness are our configuration of the training set for the vector quantizer, and the option of using the scalar quantizer. The scalar quantizer plays an important role by dealing better with regions of S that are sparsely covered by the vector quantizer codebook.

A further savings of 1 bit is obtained by codebook adaptation and a dynamic programming search. We present a new codebook adaptation scheme for the split vector quantizer that is a generalization of a method for scalar quantizers. The adaptation is based on transforming the codebook for a particular subvector in a manner that depends on the previously quantized subvector and is such that ordering is preserved. This reduces the SD without an increase in bits and without additional memory requirements. A dynamic programming search using the A^* algorithm alleviates the suboptimality of a sequential search. Although more complex than the sequential search, the A^* method is much more efficient than a prohibitive exhaustive search.

To evaluate our 29 bit quantizer subjectively, we used a CELP coder to code several sentences with quantized as well as unquantized LPC parameters. The experiments showed that the output speech signals obtained with and without quantization are indistinguishable to the ear.

Appendix A

We compare the codebooks that are derived by using the LBG algorithm (with binary splitting) for LSF's and DLSF's and show that the codebooks are equivalent. We start with a training set of LSF vectors $\boldsymbol{t}$ with components $t(i)$ for $i = 1$ to p. The equivalent set of DLSF vectors have components $\Delta t(1) = t(1)$ and $\Delta t(i+1) = t(i+1) - t(i)$ for $i = 1$ to $p-1$. Consider the vector quantization of LSF's using the distortion measure d as given by (2). For a 0 bit system, there is one Voronoi cell R_1 and one codeword $\hat{\boldsymbol{f}}_1$ with components[1]

$$\hat{f}_1(i) = \frac{1}{N(R_1)} \sum t(i) \tag{A.1}$$

for $i = 1$ to p where $N(R_1)$ is the number of training vectors in R_1. The procedure continues by splitting the centroid $\hat{\boldsymbol{f}}_1$ into two codewords $\hat{\boldsymbol{f}}_1$ and $\hat{\boldsymbol{f}}_2$ as a starting point for a 1 bit system. The Voronoi cells and centroids are iteratively updated until the average distortion decreases only slightly. The Voronoi cells are defined as

$$R_i = [\boldsymbol{t} : d(\boldsymbol{t}, \hat{\boldsymbol{f}}_i) \le d(\boldsymbol{t}, \hat{\boldsymbol{f}}_j) \qquad \forall j \ne i] \tag{A.2}$$

for $i = 1$ and 2. The codewords are expressed as

$$\hat{\boldsymbol{f}}_i = \frac{1}{N(R_i)} \sum_{\boldsymbol{t} \in R_i} \boldsymbol{t} \tag{A.3}$$

for $i = 1$ and 2. This process continues until a vector quantizer with the desired number of bits is designed.

For DLSF's, we start with the distortion measure

$$d_\Delta = \sum_{i=1}^{p} w(i)[\Delta_v(i) - \hat{\Delta}(i)]^2 \tag{A.4}$$

where the weights $w(i)$ are as defined in (3). Note that we are using the variable $\Delta_v(i)$ to correspond to the sequential search (see (4) and (5)). It can be shown that $d_\Delta = d$ thereby giving the same distortion measure as for LSF's. Consider the LBG algorithm applied on DLSF's with the distortion measure d_Δ. For a 0 bit system, there is one Voronoi cell S_1 and one DLSF codeword. It can be shown that this codeword has components[2]

$$\hat{\Delta}_1(i) = \frac{1}{N(S_1)} \sum \Delta t(i) \tag{A.5}$$

for $i = 1$ to p where $N(S_1)$ is the number of training vectors in S_1. Since $R_1 = S_1$, the LSF and DLSF codewords are related as

$$\hat{\Delta}_1(1) = \hat{f}_1(1) \tag{A.6}$$

$$\hat{\Delta}_1(i+1) = \hat{f}_1(i+1) - \hat{f}_1(i) \tag{A.7}$$

for $i = 1$ to $p-1$. To proceed to a 1 bit system, the DLSF centroid is split into two to provide an initial codebook. Assume that the initial codebook for DLSF's is related to that of LSF's as in (A.6) and (A.7). Since $d_\Delta = d$, the Voronoi cells are identical in that $S_i = R_i$ for $i = 1$ and 2. Therefore, the codebook for DLSF's is again related to the codebook for LSF's as in (A.6) and (A.7). This is true for every iteration. By induction, as the number of codewords is increased to get the desired number of bits, the codebooks for LSF's and DLSF's continue to be related.

[1] Note that although the weights are input dependent, we use the above expression for the components of the centroid that would result if the weights are constant. This is done to ensure that the ordering property is satisfied.

[2] Again, we compute the centroid that would result if the weights are constant.

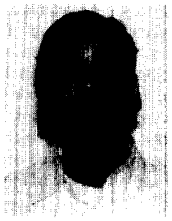
The conclusion of this derivation is as follows. The established relationship between the codebooks for LSF's and DLSF's as given by (A.6) and (A.7) shows a form of equivalence in that the same number of bits are needed in either domain to get a certain performance. However, in applying the LBG algorithm in either domain, this equivalence is based on the assumption that the initial codebooks derived from binary splitting depict this same relationship.

ACKNOWLEDGMENT

The authors thank P. Kroon for supplying the software to run the CELP coder and for furnishing the quantizer tables for the adaptive and stochastic codebook gains.

REFERENCES

[1] L. R. Rabiner and R. W. Schafer, *Digital Processing of Speech Signals*. Englewood Cliffs, NJ: Prentice-Hall, 1978.
[2] F. Itakura, "Line spectrum representation of linear predictor coefficients of speech signals," *J. Acoust. Soc. Amer.*, vol. 57, pp. S35(A), 1975.
[3] H. Wakita, "Linear prediction voice synthesizers: Line spectrum pairs (LSP) is the newest of several techniques," *Speech Technology*, Fall 1981.
[4] F. K. Soong and B.-H. Juang, "Line spectrum pair (LSP) and speech data compression," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing*, San Diego, CA, Mar. 1984, pp. 1.10.1–1.10.4.
[5] G. S. Kang and L. J. Fransen, "Application of line spectrum pairs to low bit rate speech encoders," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing*, Tampa, FL, Apr. 1985, pp. 7.3.1–7.3.4.
[6] K. K. Paliwal and B. S. Atal, "Efficient vector quantization of LPC parameters at 24 bits/frame," *IEEE Trans. Speech, Audio Processing*, vol. 1, pp. 3–14, Jan. 1993.
[7] R. Laroia, N. Phamdo, and N. Farvardin, "Robust and efficient quantization of speech LSP parameters using structured vector quantizers," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing*, Toronto, Canada, May 1991, pp. 641–644.
[8] Y. Linde, A. Buzo, and R. M. Gray, "An algorithm for vector quantizer design," *IEEE Trans. Commun.*, vol. COM-28, pp. 84–95, Jan. 1980.
[9] J. Makhoul, S. Roucos, and H. Gish, "Vector quantization in speech coding," *Proc. IEEE*, vol. 73, pp. 1551–1558, Nov. 1985.
[10] A. Gersho and R. M. Gray, *Vector Quantization and Signal Compression*. Norwood, MA: Kluwer Academic Publishers, 1992.
[11] B. S. Atal and M. R. Schroeder, "Predictive coding of speech signals and subjective error criteria," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-27, pp. 247–254, June 1979.
[12] S. Singhal and B. S. Atal, "Improving performance of multi-pulse LPC coders at low bit rates," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing*, San Diego, CA, Mar. 1984, pp. 1.3.1–1.3.4.
[13] F. K. Soong and B.-H. Juang, "Optimal quantization of LSP parameters," *IEEE Trans. Speech, Audio Processing*, vol. 1, pp. 15–24, Jan. 1993.
[14] N. Sugamara and N. Farvardin, "Quantizer design in speech analysis-synthesis," *IEEE J. Select. Areas Commun.*, vol. 6, pp. 432–440, Feb. 1988.
[15] W.-Y. Chan, S. Gupta, and A. Gersho, "Enhanced multistage vector quantization by joint codebook design," *IEEE Trans. Commun.*, vol. 40, pp. 1693–1697, Nov. 1992.
[16] W. P. LeBlanc, V. Cuperman, B. Bhattacharya, and S. A. Mahmoud, "Efficient search and design procedures for robust multi-stage VQ of LPC parameters for 4 kb/s speech coding," *IEEE Trans. Speech, Audio Processing*, vol. 1, pp. 373–385, Oct. 1993.
[17] E. Paksoy, W.-Y. Chan, and A. Gersho, "Vector quantization of speech LSF parameters with generalized product codes," in *Proc. Int. Conf. Spoken Lang. Proc.*, Banff, Canada, Oct. 1992, pp. 33–36.
[18] N. J. Nilsson, *Problem-Solving Methods in Artificial Intelligence*. New York: McGraw-Hill, 1971.
[19] F. K. Soong and B.-H. Juang, "Optimal quantization of LSP parameters using delayed decisions," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing*, Albuquerque, NM, Apr. 1990, pp. 185–188.
[20] M. R. Schroeder and B. S. Atal, "Code-excited linear prediction (CELP): High-quality speech at low bit rates," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing*, Tampa, FL, Mar. 1985, pp. 25.1.1–25.1.4.
[21] P. Kroon and K. Swaminathan, "A high-quality multirate real-time CELP coder," *IEEE J. Select. Areas Commun.*, vol. 10, pp. 850–857, June 1992.
[22] Y. Shoham, "Constrained-stochastic excitation coding of speech at 4.8 kb/s," in *Advances in Speech Coding*, B. S. Atal, V. Cuperman, and A. Gersho Eds. Norwood, MA: Kluwer Academic Publishers, 1991, pp. 339–348.



Ravi P. Ramachandran was born in Bangalore, India, on July 12, 1963. He received the B.Eng. degree (with great distinction) form Concordia University, Montreal, P.Q., Canada, in 1984 and the M.Eng. and Ph.D. degrees from McGill University, Montreal, P.Q., Canada, in 1986 and 1990, respectively.

From January to June 1988, he was a Visiting Postgraduate Researcher at the University of California, Santa Barbara. From October 1990 to December 1992, he worked in the Speech Research Department at AT&T Bell Laboratories, Murray Hill, NJ. Since January 1993, he has been a Research Assistant Professor at the Caip Center, Department of Electrical Engineering, Rutgers University, Piscataway, NJ. His main research interests are in speech processing, data communications and digital signal processing.

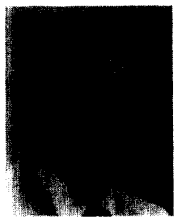
Man Mohan Sondhi received the B.Sc. degree (Honours) in physics from Delhi University, Delhi, India, in 1950, the D.I.I.Sc. degree in communications engineering from the Indian Institute of Science, Bangalore, India, in 1953, and the M.S. degree in electrical engineering and the Ph.D. degree from the University of Wisconsin, Madison, in 1955 and 1957, respectively.

He has been with AT&T Bell Laboratories since 1962. Before joining Bell Laboratories, he worked for one and a half years at the Avionics division of John Oster Mfg. Co., Racine, WI; for one year at the Central Electronics Research Institute, Pilani, India; and taught for one year at Toronto University, Toronto, Canada. He is a supervisor and Distinguished Member of Technical Staff, in the Acoustics Research Department at AT&T Bell Labs. He holds nine patents and has authored or co-authored over 90 published articles. He has co-edited *Advances in Speech Signal Processing* with S. Furui of NTT, Japan. His research has included work on i) speech signal processing, ii) echo cancellation, iii) adaptive filtering, iv) modeling of auditory, speech and visual processing by human beings, v) acoustical inverse problems, vi) speech recognition, and vii) analysis and synthesis of speech using articulatory models. He was a guest scientist for one year (1971–1972) at the Royal Institute of Technology, Stockholm, Sweden. He was also guest scientist for short periods at CNET, Lannion, France (1982) and NTT Human Interface Lab, Musashino, Japan (1990).

Dr. Sondhi was associate editor of the IEEE TRANSACTIONS ON ACOUSTICS, SPEECH, AND SIGNAL PROCESSING for several years, and a distinguished lecturer of the ASSP society for two years.

Nambi Seshadri (S'81–M'87) received the Bachelor's degree in electronics and communications from University of Madras, India, in 1982 and the Masters and Ph.D degrees in computer and systems engineering from Rensselaer Polytechnic Institute, Troy, NY, in 1984 and 1986, respectively.

Since 1986, he has been a Member of Technical Staff at AT&T Bell Laboratories in the Signal Processing Research Department. He has broad research interests in signal processing and communication techniques for wireless, multimedia and storage applications.



Bishnu S. Atal (M'76–SM'78–F'82) was born in Kanpur, India, and he came to the United States in 1961 to join the Acoustics Research Department of AT&T Bell Laboratories. He received the Ph.D. degree in electrical engineering from Polytechnic Institute of Brooklyn, NY, in 1968.

His research at Bell Laboratories has covered a wide range of topics in acoustics and speech. His research work is documented in over 90 technical papers and he holds 16 U.S. and many international patents. He became head of Acoustics Research Department in 1985, and head of Speech Research Department in 1990. He is internationally recognized for his many contributions to speech analysis, synthesis, and coding. His pioneering work in linear predictive coding of speech established linear prediction as one of the most important speech analysis techniques leading to many applications in coding, recognition and synthesis of speech. He invented multipulse linear predictive coding and code-excited linear prediction (CELP) which have found widespread applications for efficient transmission of speech in telephone networks. The CELP voice coders have been adopted as standards for digital transmission of voice on cellular radio systems in North America, Europe, Japan, and elsewhere in the world to meet the increasing demand for cellular phones.

Dr. Atal is a Fellow of the Acoustical Society of America. He was elected to the National Academy of Engineering in 1987, and to the National Academy of Sciences in 1993. He received the IEEE Signal Processing Society Award in 1993 for contributions to linear prediction of speech, multipulse and code-excited source coding and the 1975 IEEE ASSP Society Technical Achievement Award for fundamental contributions to linear predictive coding of speech signals. In 1980 he received, jointly with M. R. Schroeder, the IEEE ASSP Senior Award. He is the recipient of the IEEE Centennial Medal in 1984 and the IEEE Morris N. Liebmann Memorial Field Award in 1986 for his pioneering contributions to linear predictive coding for speech processing. In 1994 he was named a Bell Labs Fellow, and in 1995 he was awarded the Thomas A. Edison Patent Award.