SRC TR 89-74

Adaptive Block Transform Coding of Speech Based on LPC Vector Quantization

By

Y. Hussain and N. Farvardin

Adaptive Block Transform Coding of Speech Based on LPC Vector Quantization[†]

Y. Hussain and N. Farvardin
Electrical Engineering Department
Institute for Advanced Computer Studies
and
Systems Research Center
University of Maryland
College Park, MD 20742

Abstract

In this paper we describe an adaptive block transform speech coding system based on vector quantization of LPC parameters. In order to account for the power fluctuations, the speech signal is normalized to have a unit-energy prediction residual. The temporal variations in the short-term spectrum, on the other hand, are taken into account by vector quantizing the LPC parameters associated with the vector of speech samples and transmitting the codeword index. For each block, based on the codevector associated with the input vector, an optimum bit assignment map is used to quantize the transform coefficients. We consider two types of zero-memory quantizers for encoding the transform coefficients, namely the Llyod-Max quantizer and the entropy-coded quantizer. The performance of these schemes is compared with other adaptive transform coding schemes. We show by means of simulations that the system based on entropy-coded quantizer design leads to very high performance and in most cases as much as 5 dB performance improvement in terms of segmental signal-to-noise ratio is observed over the adaptive block transform coding scheme of Noll and Zelinski [1]. The effects of the bit-rate and the size of the codebook on the performance of the systems are also studied in detail.

[†]This work was supported in part by National Science Foundation grants NSFD MIP-86-57311 and NSFD CDR-85-00108 and in part by grants from Martin Marietta Laboratories and General Electric Company.

1 Introduction

Block transform coding (BTC), when made adaptive to the changing characteristics of the speech, has proven to be a promising medium-band speech coding scheme. In a nonadaptive block transform speech coding system, each block of speech samples is transformed into a set of transform coefficients; these coefficients are quantized independently by scalar quantizers and transmitted. An inverse transform is taken at the receiver to obtain a corresponding block of reconstructed samples. Such a scheme will perform well for a source which is stationary as shown by Huang and Schulthesis [7] for the particular case of a Gaussian stationary source. However, in practical situations such as speech coding, the source is at best quasi-stationary and the BTC scheme needs to be made adaptive to the changing statistics of the incoming signal. Typical speech waveforms exhibit temporal variations in the signal power and the short-term spectrum. Various adaptive schemes for block transform coding of speech have been reported in the literature [1], [2], [3].

In the scheme described in [1], estimates of the variances of transform coefficients are made by local averaging of the transform coefficients. These estimates, which are also sent to the receiver as side information, are used to adaptively vary the bit allocation among the transform coefficients. In [2], the speech signal is modeled by an autoregressive Gaussain hidden Markov process [4]. The short-term spectrum of speech is then determined from a maximum-likelihood estimate of the state and is used for adaptive bit allocation. Again the state estimate is sent to the receiver as side information.

In this paper, we consider an alternative method to model the short-term spectrum of the speech signal based on ideas borrowed from [3]. Specifically, we vector quantize the LPC parameters (LPCVQ) associated with each speech block [5] and transmit the index of the codevector as overhead information; this codevector will determine the short-term spectrum corresponding to that block and, in turn, can be used for optimal bit allocation among the transform coefficients. In addition, we consider entropy-coded zero-memory quantization of the transform coefficients as an alternative to Lloyd-Max quantization. We will develop an adaptive BTC scheme based on LPCVQ and using entropy-coded quantizers. We will show via extensive simulations that this scheme performs considerably better than that of [1].

The rest of the paper is organized as follows. In Section 2, we develop notation and briefly discuss the basic operation of the BTC scheme. Section 3 describes the structure

of LPCVQ-based adaptive BTC speech coding system, while Section 4 discusses a similar adaptive scheme which uses entropy-coded quantization for encoding the transform coefficients. In Section 5, simulation results are presented, followed by Section 6 which provides a summary and conclusions.

2 Preliminaries and Notation

Let $\{X_n, n = 0, 1, \ldots\}$ be a zero-mean stationary source with variance σ^2. An L-dimensional *nonadaptive* BTC scheme operates as follows. A typical vector of length L denoted by $\mathbf{X} = (X_0, X_1, \ldots, X_{L-1})^{\mathrm{T}}$ is operated upon by a linear unitary $L \times L$ transformation $\mathbf{T}$ to obtain the transformed vector $\mathbf{Y}$ described by

$$\mathbf{Y} = (Y_0, Y_1, \ldots, Y_{L-1})^T = \mathbf{TX} \ .$$

The components of $\mathbf{Y}$, called the transform coefficients, are subsequently quantized independently by zero-memory Llyod-Max quantizers to give the quantized version $\hat{\mathbf{Y}} = (\hat{Y}_0, \hat{Y}_1, \ldots, \hat{Y}_{L-1})^{\mathrm{T}}$ of the transformed vector. The quantized transform coefficients are subsequently encoded and transmitted. At the receiver (assuming a noiseless channel), a replica of $\mathbf{X}$, say $\hat{\mathbf{X}}$, is obtained by an inverse transformation on $\hat{\mathbf{Y}}$ according to

$$\hat{\mathbf{X}} = \mathbf{T}^{-1}\hat{\mathbf{Y}} \ .$$

For the squared-error distortion criterion, the average per-sample distortion is given by

$$D = \frac{1}{L} E\{\|\mathbf{X} - \hat{\mathbf{X}}\|^2\} \ ,$$

which, under the assumption of unitary transformation $\mathbf{T}$, reduces to

$$D = \frac{1}{L} E\{\|\mathbf{Y} - \hat{\mathbf{Y}}\|^2\} = \frac{1}{L} \sum_{i=0}^{L-1} E\{(Y_i - \hat{Y}_i)^2\} \ .$$

Each term in the summation above is the mean squared-error (MSE) caused by zero-memory quantization of a transform coefficient. Adopting the notation of [2], let us assume that λ_i is the variance of the i^{th} transform coefficient and let $d_i(b_i)$ denote the MSE associated with the b_i-bit Lloyd-Max quantization of the i^{th} transform coefficient after normalization to unit-variance. Using this notation, the per-sample MSE for the overall system can be written as

$$D(\mathbf{b}) = \frac{1}{L} \sum_{i=0}^{L-1} \lambda_i d_i(b_i) \ , \qquad (1.a)$$

where $\mathbf{b} = (b_0, b_1, \ldots, b_{L-1})$ is called the bit assignment map in which b_i is the number of bits assigned to the i^{th} transform coefficient. For a prescribed average number of bits per sample, say b_{av}, the optimum bit assignment map $\mathbf{b}^* = (b_0^*, b_1^*, \ldots, b_{L-1}^*)$, is that which minimizes the per-sample MSE given by (1.a) subject to a constraint on the average number of bits per sample given by

$$R(\mathbf{b}) = \frac{1}{L} \sum_{i=0}^{L-1} b_i \leq b_{av}. \tag{1.b}$$

Additional details about the BTC scheme can be found in [6]. Various algorithms for bit assignment can be found in [6], [7] and [8].

The above analysis is carried out under the assumption of a stationary source. In practical situations, such as speech coding, this assumption does not hold. In fact, both the signal variance σ^2 and power spectral density (hence, the variances of transform coefficients) vary with time. In such cases, some type of adaptation of the BTC scheme to the changing statistics of the signal is needed. To do this, it is assumed that the signal is quasi-stationary in the sense that its statistical characteristics do not change within a block - hence the adaptation takes place on a block-to-block basis.

A well-known scheme for adaptive block transform coding of speech is described in [1]. In [1], to account for the variability in the input variance, the signal variance is computed for each block and then used to normalize the source output vectors. The variance normalization can be thought of as some type of *adaptive quantization* [6]. To account for the variability in the speech spectrum, an estimate of the variance of the transform coefficients is obtained for each block. These estimates are then used to adapt the allocation of bits among the quantizers used for the transform coefficients. Both the variance of the input block and the estimates of the variances of the transform coefficients are transmitted to the receiver through a side channel. As a consequence of the adaptive quantization and adaptive bit allocation, an improvement of about 6 dB in segmental signal-to-noise ratio is observed over the nonadaptive block transform speech coding system [1].

In [1], the variances of the transform coefficients are estimated in an "ad hoc" fashion and the overhead information accounts for 2 Kbits/sec (0.25 bits/sample). In what follows, we describe an alternative means to estimate the variances of the transform coefficients which not only results in improved system performance but also requires lower overhead information.

3 Adaptive BTC Scheme Based on LPCVQ

The basic idea in the adaptive scheme to be developed revolves around the LPC model of speech. Specifically, it is assumed that during short time intervals (10-30 msec), the speech signal can be described by the linear predictive model described by

$$X_n = -\sum_{i=1}^{p} a_i X_{n-i} + gW_n,$$

in which p is the order of the model and $\{gW_n\}$ is the prediction residual [9]. Here it is assumed that $\{W_n\}$ is a unit-energy signal; the parameter g is referred to as the *gain factor* and the vector $\mathbf{a} = (a_1, a_2, \ldots, a_p)$ is called the LPC vector.

Due to the quasi-stationary nature of speech, the LPC vectors need to be updated for every frame of speech. There are various algorithms in the literature for computing the LPC parameters [9] and therefore we will not elaborate on this issue. From now on, the LPC vector and the gain factor associated with the t^{th} frame of speech $\mathbf{x}_t = (x_{tL}, x_{tL+1}, \ldots, x_{tL+L-1})^{\mathrm{T}}$, will be denoted by $\mathbf{a}_t = (a_1^t, a_2^t, \ldots, a_p^t)$ and g_t, respectively.

Notice that the knowledge of g_t and $\mathbf{a}_t$ enables us to determine the energy and spectrum of the t^{th} speech frame. Therefore, this information can be used to adaptively vary the parameters of the BTC scheme. However, these parameters need to be transmitted to the receiver as well which amounts to some additional overhead information. To reduce the overhead information, we vector quantize the LPC vectors [10]. That is, instead of transmitting the vector $\mathbf{a}_t$, we transmit an index corresponding to the closest codevector in the vector quantizer (VQ) codebook. In what follows, we will describe the details of the system implementation.

Let us assume that for the gain-normalized Itakura-Saito (GNIS) distortion measure [10] an M-level LPC vector quantizer (LPCVQ) has been designed using the algorithm described in [10], [11]. We denote the resulting codebook by $\mathcal{C} = \{\mathbf{c}_1, \mathbf{c}_2, \ldots, \mathbf{c}_M\}$, where

$$\mathbf{c}_j = (a_{j,1}, a_{j,2}, \ldots, a_{j,p}),\ j = 1, 2, \ldots, M,$$

and $a_{j,i}$, $i = 1, 2, \ldots, p$, are the LPC coefficients corresponding to the j^{th} codevector.

For convenience of presentation, we introduce the notion of "state" in the following sense: we say that the system is in state j at time t if the codevector resulting from vector quantization of the frame $\mathbf{x}_t$ is $\mathbf{c}_j$ (i.e., the codevector index is j).

An algorithm which uses LPCVQ to adapt the BTC scheme is described below.

3.1 The Coding Algorithm

The adaptive coding scheme can be described in the following steps:

1. For the input vector $\mathbf{x}_t$, the LPC vector $\mathbf{a}_t$ and the gain factor g_t are computed (using standard LPC analysis techniques [9]). A quantized version of g_t, say $\hat{g}_t$, is transmitted over a side channel.

2. The input vector is normalized to unit residual energy. Specifically, for each input vector $\mathbf{x}_t$, the normalized vector $\tilde{\mathbf{x}}_t$ is obtained according to $\tilde{\mathbf{x}}_t = \mathbf{x}_t/g_t$. Therefore, as a result of this normalization, the vector $\tilde{\mathbf{x}}_t$ will always have a unit-variance prediction residual.

3. The codebook $\mathcal{C}$ is searched to obtain the codevector which is closest to $\tilde{\mathbf{x}}_t$ in the GNIS sense [10]. Specifically, the codevector index (or, equivalently, state) at time t is given by

$$s_t = \underset{s\in\{1,2,\ldots,\ M\}}{\arg} \min\{r_{a_s}(0)r_t(0) + 2\sum_{m=1}^{p} r_{a_s}(m)r_t(m)\}\ , \tag{2}$$

where

$$r_{a_s}(m) = \sum_{i=0}^{p-m} a_{s,i}a_{s,m+i}\ ,$$

and

$$r_t(m) = \sum_{i=tL}^{tL+L-1} \tilde{x}_i\tilde{x}_{i+m}\ .$$

Using the above classification technique (nearest-neighbor rule), a codevector can be associated with each input vector $\tilde{\mathbf{x}}_t$, $t = 1, 2, \ldots$, resulting in a state sequence $\mathbf{s} = \{s_1, s_2, \ldots\}$ of the system. The state sequence is transmitted over a side channel.

4. The linear transformation $\mathbf{T}$ is performed on $\tilde{\mathbf{x}}_t$ to obtain $\mathbf{y}_t$, the vector of transform coefficients.

5. The components of $\mathbf{y}_t$ are quantized and encoded separately. The bit assignment map used in encoding this vector is given by a vector $\mathbf{b}^*_{s_t}$, optimized for the state s_t, as described in the next subsection.

The decoding process is essentially the reverse of the encoding process. Upon receiving the index s_t and the quantized version of g_t through the side channel, the vector $\hat{\mathbf{y}}_t$, the quantized version of $\mathbf{y}_t$, is reconstructed based on s_t and $\mathbf{b}^*_{s_t}$. This vector is then inverse

transformed to obtain $\hat{\tilde{\mathbf{x}}}_t$ which is subsequently denormalized to give a replica of $\mathbf{x}_t$ given by $\hat{\mathbf{x}}_t = \hat{g}_t \hat{\tilde{\mathbf{x}}}_t$. We shall refer to this coding scheme by LPCVQ-LMBTC.

3.2 Variance Computation and Bit Assignment

In this paper, the linear transformation used is the discrete cosine transformation (DCT). The rationale behind using the DCT is that it is a signal independent, computationally efficient transformation with *good* performance [1], [6]. Moreover, to make fair comparisons against the results in [1], we had to use the same transformation.

The L-point DCT of a vector $\tilde{\mathbf{x}} = (\tilde{x}_0, \tilde{x}_1, \ldots, \tilde{x}_{L-1})$ is a vector $\mathbf{y}$ whose components are given by [6]

$$y_k = \frac{2C_k}{L} \sum_{n=0}^{L-1} \tilde{x}_n \cos \frac{(2n+1)k\pi}{2L}, \quad k = 0, 1, \ldots, L-1, \tag{3}$$

where $C_0 = \frac{1}{\sqrt{2}}$, $C_k = 1$, $k = 1, 2, \ldots, L-1$.

Given the LPC model of speech, the variances of the transform coefficients can be computed explicitly. Specifically, given that $s_t = s$, (i.e., system is in state s), it is easily shown that the variance of the k^{th} transform coefficient is given by

$$\lambda_{s,k} = \frac{4C_k^2}{L^2} \sum_{i=0}^{L-1} \sum_{j=0}^{L-1} r_{a_s}(|i-j|) \cos \frac{(2i+1)k\pi}{2L} \cos \frac{(2j+1)k\pi}{2L}, \quad k = 0, 1, \ldots, L-1. \tag{4}$$

Empirical evidence has shown that the transform coefficients have a distribution close to Gaussian [1]. With the above assumption and noting that the DCT is a linear transformation, the transform coefficients will also be Gaussian. Therefore, the MSE associated with state s, denoted by D_s is given by

$$D_s(\mathbf{b}_s) = \frac{1}{L} \sum_{k=0}^{L-1} \lambda_{s,k} d(b_{s,k}) \ , \tag{5.a}$$

where $b_{s,k}$ is the number of bits used for encoding the k^{th} transform coefficient when the system is in state s, and $d(b)$ denotes the variance-normalized MSE associated with b-bit Lloyd-Max quantization of a Gaussian source.

Assuming the same average bit rate for all states, † the bit assignment problem is that

†Note that the imposition of assigning equal average bit rates to each state leads to a suboptimal bit allocation and hence results in some degradation in the performance as compared to the system with an optimal bit allocation. However, if the state average bit rates are allowed to vary from state to state, the overall average bit rate becomes a function of state probabilities and hence the overall average bit rate will be data dependent - an undesirable property. In this work we maintain the assumption that the state average bit rates are the same.

of determining the bit assignment map vector $\mathbf{b}_s^* = (b_{s,0}^*, b_{s,1}^*, \ldots, b_{s,L-1}^*)$ which minimizes $D_s(\mathbf{b}_s)$ described by (5.a) subject to

$$\frac{1}{L}\sum_{k=0}^{L-1} b_{s,k} \leq b_{av} \ . \tag{5.b}$$

This constrained optimization problem is solved using a steepest-descent, integer programming algorithm developed by Trushkin [8]. The bit assignment maps $\mathbf{b}_s^*$, $s = 1, 2, \ldots, M$, are used in encoding the transform coefficients in step (5) of the coding algorithm.

The scheme described in this section is similar to that of [2], in which the speech signal is modeled as an autoregressive Gaussian hidden Markov process. A brief description of this system is provided below.

3.3 HMM-Based Adaptive Scheme

In [2], the speech process is modeled by an autoregressive Gaussian hidden Markov model (AGHMM). This process is described by a finite-state homogeneous Markov chain associated with each state of which is an L-dimensional vector source corresponding to the output of a p^{th}-order stationary Gaussian source [4]. In the AGHMM-based block transform coding scheme, the state of the underlying Markov chain is estimated for each frame based on a maximum likelihood state estimation procedure. Then an optimum bit assignment map which depends on the state is used to quantize the transform coefficients associated with the frame.

Subsequently, we shall refer to the scheme described in [2] by AGHMM-LMBTC; LMBTC will jointly refer to LPCVQ-LMBTC and AGHMM-LMBTC. The basic difference between AGHMM-LMBTC and LPCVQ-LMBTC is that in the latter we use an LPCVQ in place of an AGHMM to model the short-term spectrum of speech. The LPCVQ-based scheme was motivated mainly due to numerical problems associated with the design of the AGHMM for large M and long training sequence and also, because the design of AGHMM-LMBTC was found to be computationally more expensive than LPCVQ-LMBTC.

4 Entropy-Coded Adaptive BTC Scheme

In the LMBTC scheme described in the previous section, the transform coefficients are quantized using zero-memory Lloyd-Max quantizers. In this section, we consider entropy-coded zero-memory quantization of the transform coefficients. The entropy-coded BTC

system (ECBTC), as is confirmed by the simulation results of Section 5, results in an improved performance in the rate-distortion theoretic sense in comparison with LMBTC. The reason for the improvement is twofold. First, the optimum zero-memory entropy-constrained quantizer performs better than the zero-memory Lloyd-Max quantizer since it is optimal in the rate-distortion theoretic sense. Second, the bit-rate associated with each scalar quantizer in ECBTC is not constrained to be an integer as in the case of LMBTC; the removal of this constraint leads to additional performance improvements. In what follows, we provide the details of the adaptive ECBTC scheme.

Let us assume that when the system is in state s, the k^{th} transform coefficient is quantized by means of an optimum entropy-constrained zero-memory quantizer with output entropy $H_{s,k}$ bits/sample. Thus, according to Shannon's noiseless source coding theorem [12], the average number of bits necessary to represent the output of the k^{th} quantizer in state s is $H_{s,k}$ bits/sample. Also, let $\tilde{d}_{s,k}(H_{s,k})$ be the variance-normalized MSE associated with quantizing the k^{th} transform coefficient when the system is in state s. Then our problem is to find the optimum vector $\mathbf{H}_s^* = (H_{s,0}^*, H_{s,1}^*, \ldots, H_{s,L-1}^*)^{\mathrm{T}}$, $s = 1, 2, \ldots, M$, that minimizes the average distortion given by

$$\tilde{D}(\mathbf{H}) = \frac{1}{L} \sum_{s=1}^{M} \sum_{k=0}^{L-1} P_s \lambda_{s,k} \tilde{d}_{s,k}(H_{s,k}), \tag{6.a}$$

subject to

$$\frac{1}{L} \sum_{s=1}^{M} \sum_{k=0}^{L-1} P_s H_{s,k} \leq R_{av}, \tag{6.b}$$

where R_{av} is the constraint on the overall average output entropy, $\mathbf{H}$ is the matrix $(\mathbf{H}_1, \mathbf{H}_2, \ldots, \mathbf{H}_M)$ and P_s is the probability that the system is in state s.

Here, $\tilde{d}_{s,k}(H_{s,k})$ and $H_{s,k}$ are related to one another through the quantization thresholds $T_\ell^{(s,k)}$, $\ell = 1, 2, \ldots, N_{s,k} - 1$ and the quantization levels $Q_\ell^{(s,k)}$, $\ell = 1, 2, \ldots, N_{s,k}$, where $N_{s,k}$ is the number of levels associated with the k^{th} quantizer in state s. Solving the problem described by (6) involves the determination of $\sum_{s=1}^{M} \sum_{k=0}^{L-1} (2N_{s,k} - 1)$ threshold levels and quantization levels, which in general is very difficult.

It is shown by Gish and Pierce [13] that at high bit rates and for a large number of quantization levels, the optimum quantizer has uniformly spaced levels. Furthermore, the experimental results of Farvardin and Modestino [14] have revealed that, for a wide class of memoryless sources, even at low bit rates, uniform-threshold quantizers

(UTQ's)[†] perform very close to the optimum performance for large N. In view of these observations, we have used UTQ's to quantize the transform coefficients. Let us use $\hat{d}_{s,k}(\Delta_{s,k})$ and $\hat{H}_{s,k}(\Delta_{s,k})$ to denote, respectively, the MSE and output entropy of the UTQ with stepsize $\Delta_{s,k}$ for the k^{th} transform coefficient and state s. Then the problem posed by (6) is reformulated as that of determining $\boldsymbol{\Delta}^*$ to minimize

$$\hat{D}(\boldsymbol{\Delta}) = \frac{1}{L}\sum_{s=1}^{M}\sum_{k=0}^{L-1} P_s \lambda_{s,k} \hat{d}_{s,k}(\Delta_{s,k}) \ , \tag{7.a}$$

subject to

$$\frac{1}{L}\sum_{s=1}^{M}\sum_{k=0}^{L-1} P_s \hat{H}_{s,k}(\Delta_{s,k}) \leq R_{av}, \tag{7.b}$$

where $\boldsymbol{\Delta} = (\boldsymbol{\Delta_1}, \boldsymbol{\Delta_2}, \ldots, \boldsymbol{\Delta}_M)$ and $\boldsymbol{\Delta}_s = (\Delta_{s,0}, \Delta_{s,1}, \ldots, \Delta_{s,L-1})^{\mathrm{T}}$, $s = 1, 2, \ldots, M$.

It is also shown by Gish and Pierce [13] that in zero-memory quantization of memoryless sources, when the number of quantization levels is large and when the output entropy is high, under certain mild conditions on the source p.d.f, the variance-normalized MSE and output entropy can be expressed as a function of quantizer stepsize according to

$$\hat{d}(\Delta_{s,k}) = \frac{\Delta_{s,k}^2}{12\lambda_{s,k}}, \tag{8.a}$$

and

$$\hat{H}(\Delta_{s,k}) = h_{s,k} - \log_2 \Delta_{s,k}, \text{ bits/sample}, \tag{8.b}$$

where $h_{s,k}$ is the *differential entropy* associated with the k^{th} transform coefficient in state s. Invoking Kuhn-Tucker theorem [15] and using the approximate expressions of (8) we obtain

$$\Delta_{s,k}^* = \frac{1}{\sqrt{\lambda_{s,k}}} 2^{[h - R_{av}]}, \ s = 1, 2, \ldots, M, \ k = 0, 1, \ldots, L-1, \tag{9.a}$$

where

$$h = \frac{1}{L}\sum_{s=1}^{M}\sum_{k=0}^{L-1} P_s h_{s,k}. \tag{9.b}$$

For reasons discussed before, the transform coefficients will be Gaussian and the differential entropy $h_{s,k}$ is given by

$$h_{s,k} = \frac{1}{2}\log_2 2\pi e \lambda_{s,k}, \text{ bits/sample.}$$

[†]An N-level UTQ is a symmetric quantizer with $T_\ell - T_{\ell-1} = \Delta$, $\ell = 2, \ldots, N-1$, and Q_ℓ's as the center of probability mass of their respective quantization interval.

The operation of the ECBTC system is the same as the LMBTC scheme with only one difference that we use entropy-coded zero-memory quantizers for encoding the transform coefficients in ECBTC in place of zero-memory Lloyd-Max quantizer as in the LMBTC system.

Note that for any state s, $\mathbf{\Delta}_s^*$ in ECBTC will determine the quantizers' operating rate similar to $\mathbf{b}_s^*$ in LMBTC. Our simulation results revealed that for $\Delta_{s,k} \leq 1.0$ the actual MSE and output entropy are well approximated by the expressions in (8); for larger values of $\Delta_{s,k}$, the differences between the actual results and those suggested by (8) were more noticeable. However, the transform coefficients with large values of $\Delta_{s,k}$ have small variances and their contribution to the overall system performance is masked by that of coefficients with large variance (small stepsize). Therefore, the exact expression for $\hat{d}(\Delta_{s,k})$ and $\hat{H}(\Delta_{s,k})$ for large $\Delta_{s,k}$ is not very critical as far as the overall system performance is concerned. For simplicity of analysis, therefore, we have used expressions (8) for all values of $\Delta_{s,k}$.

The above analysis holds for AGHMM-based BTC scheme of [2] as well as the LPCVQ-based BTC described in Section 3. Hereafter, we refer to ECBTC system based on LPCVQ by LPCVQ-ECBTC, while AGHMM-ECBTC refers to ECBTC based on AGHMM. The simulation results for the two schemes are provided in the next section.

5 Simulation Results

We performed extensive simulations to compare the LPCVQ-based BTC systems of Sections 3 and 4 with the schemes described in [1] and [2]. The performance measures used are signal-to-noise ratio (SNR) and segmental signal-to-noise ratio (SNRSEG). We denote by b_{tot} the total average bit rate including the overhead information (b_{oh}), i.e., $b_{tot} = b_{av} + b_{oh}$.

5.1 Performance on the in-training test sequence

The AGHMM- and LPCVQ-based BTC systems were designed for $p = 10$, $L = 128$ and $M = 1, 2, 4, 8, 16$ and 32. The database used for these designs was a sequence of speech samples consisting of 60 seconds of speech sampled at 8 KHz and uttered by two male speakers.

The performances of the LPCVQ-LMBTC and AGHMM-LMBTC on the training sequence are tabulated in Table 1. We also include the performance results obtained using the

scheme of [1] (denoted by Z-N) for comparison purposes. Study of Table 1 indicates that at $b_{tot} = 1.25$ bits/sample AGHMM-LMBTC performs better than LPCVQ-LMBTC in terms of SNRSEG by about 1 dB on the average, while in terms of SNR, LPCVQ-LMBTC outperforms AGHMM-LMBTC by 2.5 dB on the average. Subjectively, the two schemes are comparable. However, the design of the LPCVQ-LMBTC is found to be computationally less expensive than the AGHMM-LMBTC. Both LPCVQ-LMBTC and AGHMM-LMBTC with $M = 32$ perform better than the Z-N scheme at $b_{tot} = 1.25$ bits/sample . Under these conditions, LPCVQ-LMBTC is 1.36 dB better than Z-N in terms of SNRSEG, while AGHMM-LMBTC is better by over 2 dB. The SNR for LPCVQ-LMBTC is about 5 dB higher than Z-N scheme, while AGHMM-LMBTC performs better by over 3 dB. The study of tabulated results in Table 1 shows a similar trend at $b_{tot} = 2.25$ bits/sample.

The performance of LPCVQ-ECBTC, AGHMM-ECBTC and Z-N schemes for $b_{tot} = 1.25$ and 2.25 bits/sample are illustrated in Tables 2. In terms of SNRSEG, while LPCVQ-ECBTC and AGHMM-ECBTC perform closely, the two schemes outperform the Z-N scheme by over 3.5 dB when $M = 32$. On the average, LPCVQ-ECBTC performs better than AGHMM-ECBTC and Z-N schemes by about 1 dB and 5 dB respectively, in terms of the SNR. Subjectively, LPCVQ-ECBTC and AGHMM-ECBTC schemes are comparable and both are noticeably superior to the Z-N scheme.

In order to assess the efficacy of LPCVQ-ECBTC [†] and LPCVQ-LMBTC for larger values of M, we designed LPCVQ's for $p = 10$, $L = 128$ and $M = 1, 2, 4, 8, 16, 32, 64, 128, 256, 512, 1024$. In this case the training sequence consisted of several sentences uttered by 12 male and 10 female speakers. The database consisted of approximately 14 minutes of speech sampled at 8 KHz.

Table 3 shows the performance of LPCVQ-LMBTC in terms of SNRSEG and SNR on a test sequence of duration 60 seconds taken from the training sequence for $b_{tot} = 1.25$ bits/sample and 2.25 bits/sample. While AGHMM could not be trained for this long database due to numerical problems encountered, we provide the performance of the Z-N scheme for the sake of comparison. Actually the test sequence used is the same as the training sequence described in the beginning of this section. From Table 3, it is observed that for $b_{tot} = 1.25$ bit/sample, LPCVQ-LMBTC outperforms Z-N scheme when $M = 128$; for $M = 1024$, LPCVQ-LMBTC is better than Z-N scheme by 2 dB in SNRSEG and by

[†]We could not obtain results for LPCVQ-ECBTC system with M greater than 256 due to limitation on computing facility.

6 dB in SNR. The trend is similar for $b_{tot} = 2.25$ bits/sample. For $M = 1024$, SNRSEG difference is 1.75 dB, while SNR differs by 10 dB.

The performance of LPCVQ-ECBTC system on the training sequence is tabulated in Table 4. LPCVQ-ECBTC performs considerably better than the Z-N scheme in both subjective and objective sense. For $M = 128$ and $b_{tot} = 2.25$ bits/sample, improvements in terms of SNRSEG and SNR are of the order of 5.5 dB and 11 dB, respectively. An interesting observation is that even with a one-codevector codebook (no adaptation of bit assignment maps), LPCVQ-ECBTC performs better than the Z-N scheme at $b_{tot} = 1.25$ and 2.25 bits/sample.

5.2 Performance on the out-of-training test sequence

The performance of LPCVQ-LMBTC on a test sequence chosen from outside the training sequence is summarized in Table 5. The test sequence was 55 seconds of speech sampled at 8 KHz. It consisted of three different sentences spoken by six male and three female speakers. The performance results of the Z-N scheme on the same test sequence is also included for comparison purposes. It is observed that for $b_{tot} = 1.25$ and 2.25 bits/sample, the SNRSEG of LPCVQ-LMBTC with $M = 1024$ is close to that of Z-N scheme, while the SNR is better by about 3 dB at $b_{tot} = 1.25$ bits/sample and 4 dB at $b_{tot} = 2.25$ bits/sample. Subjectively, LPCVQ-LMBTC is slightly better than Z-N.

Table 6 shows the performance of LPCVQ-ECBTC on the aforementioned test sequence. These results clearly highlight the superiority of the LPCVQ-ECBTC system. Even with $M = 1$, LPCVQ-ECBTC system performs better than the Z-N scheme and when $M = 128$, increase in SNRSEG over Z-N system is 4 dB at $b_{tot} = 1.25$ and 5.5 dB at $b_{tot} = 2.25$ bits/sample. The corresponding improvements in SNR are 8 dB and 11 dB, respectively.

Also comparisons of Table 5 with Table 6 and Table 1 with Table 2 support our earlier contention that ECBTC scheme is better than LMBTC. For example, study of Table 5 and 6 shows that at $b_{tot} = 1.25$ bits/sample, LPCVQ-ECBTC with $M = 256$ outperforms LPCVQ-LMBTC with $M = 1024$ by 4.5 dB in SNRSEG and 5.5 dB in SNR and when $M = 256$ for both systems (for fair comparison), the corresponding figures are 5.5 dB and 6 dB, respectively.

6 Summary and Conclusions

We have used LPCVQ to design an adaptive block transform speech coding system. The basic operation of this scheme is similar to the scheme described in [2] where an AGHMM was used to model the source. The LPCVQ-based system was motivated mainly due to certain design problems with AGHMM for large values of M. We have also developed an entropy-coded version of this adaptive BTC scheme. The simulation results indicate that the LPCVQ-ECBTC performed better than LPCVQ-LMBTC by at least 3 dB in both SNR and SNRSEG; similar behavior is observed for the AGHMM-based scheme. Furthermore, both AGHMM- and LPCVQ-based BTC schemes performed better than the Z-N scheme. The most interesting results are obtained for LPCVQ-ECBTC (with large M). This scheme offers significant objective and subjective improvements over Z-N for test data from inside and outside the training sequence.

References

[1] R. Zelinski and P. Noll, "Adaptive Transform Coding of Speech Signals," *IEEE Trans. Acoust. Speech Signal Process.*, vol. ASSP-25, pp. 299-309, Aug. 1977.

[2] N. Farvardin and Y. Hussain, "Adaptive Block Transform Coding of Speech Based on the Hidden Markov Model," *Proc. EUSIPCO*, Grenoble, France, pp. 883-886, Sept. 1988.

[3] J.M. Tribolet and R.E. Crochiere, "Frequency Domain Coding of Speech," *IEEE Trans. Acoust. Speech Signal Process.*, vol. ASSP-27, pp. 512-530, Oct. 1979.

[4] B.H. Juang and L.R. Rabiner, "Mixture Autoregressive Hidden Markov Models for Speech Signals," *IEEE Trans. Acoust. Speech Signal Process.*, vol. ASSP-27, pp. 512-530, Oct. 1979.

[5] R.M. Gray, "Vector Quantization," *IEEE ASSP Mag.*, vol. 1, pp. 4-29, Apr. 1984.

[6] N.S. Jayant and P. Noll, *Digital Coding of Waveforms: Principles and Application to Speech and Video*, Prentice Hall, Inc., Englewood Cliffs, New Jersey, 1984.

[7] J. Huang and P.M. Schultheiss, "Block Quantization of Correlated Gaussian Random Variables," *IEEE Trans. Commun. Syst.*, vol. CS-11, pp. 289-296, Sept. 1963.

[8] A.V. Trushkin, " Optimal Bit Allocation Algorithm for Quantizing a Random Vector," *Problems on Information Transmission*, pp. 156-161, Jan. 1982.

[9] J.D. Markel and A.H. Gray, *Linear Prediction of Speech*, Springer-Verlag, New York, 1976.

[10] A. Buzo, A.H. Gray, Jr., R.M. Gray and J.D. Markel, "Speech Coding Based upon Vector Quantization," *IEEE Trans. Acoust. Speech Signal Process.*, vol. ASSP-28, pp. 562-574, Oct. 1980.

[11] Y. Linde, A. Buzo and R.M. Gray, "An Algorithm for Vector Quantization Design," *IEEE Trans. Commun.*, vol. COM-28, pp. 84- 95, Jan. 1980.

[12] R.G. Gallager, *Information Theory and Reliable Communication*, Wiley, New York, 1968.

[13] H. Gish and J. N. Pierce, " Asymptotically Efficient Quantizing, " *IEEE Trans. Inform. Theory*, vol. IT-14, pp. 676-683, Sept. 1968.

[14] N. Farvardin and J. W. Modestino, " Optimum Quantizer Performance for a Class of Non-Gaussian Memoryless Sources, " *IEEE Trans. Inform. Theory*, vol. IT-30, pp. 485-497, May 1984.

[15] S.M. Bazarra and C. M. Shetty, *Nonlinear Programming*, Wiley, New York, 1979.

	$b_{tot} = 1.25$				$b_{tot} = 2.25$			
	AGHMM-LMBTC		LPCVQ-LMBTC		AGHMM-LMBTC		LPCVQ-LMBTC	
M	SNRSEG	SNR	SNRSEG	SNR	SNRSEG	SNR	SNRSEG	SNR
1	15.17	11.97	13.21	14.59	18.57	14.33	18.23	18.38
2	14.91	12.06	13.14	14.73	19.85	15.37	19.42	19.69
4	16.46	13.57	15.27	15.88	21.80	17.18	21.12	21.58
8	16.80	13.73	15.22	16.81	22.47	17.37	21.55	23.18
16	17.47	14.76	16.27	17.54	23.62	18.76	22.46	24.07
32	17.85	16.04	17.13	17.78	24.00	20.27	23.28	24.63
Z-N	15.77	12.86			22.56	17.63		

Table 1: SNRSEG and SNR Performance of Different Adaptive BTC Schemes at $b_{tot} =$ 1.25 and 2.25 bits/sample on the Training Sequence.

	$b_{tot} = 1.25$				$b_{tot} = 2.25$			
	AGHMM-ECBTC		LPCVQ-ECBTC		AGHMM-ECBTC		LPCVQ-ECBTC	
M	SNRSEG	SNR	SNRSEG	SNR	SNRSEG	SNR	SNRSEG	SNR
1	16.25	16.77	16.54	17.91	22.24	20.70	22.71	23.75
2	17.28	17.36	18.01	18.80	23.60	21.31	24.63	25.36
4	18.26	18.01	18.37	19.22	25.16	21.95	25.10	25.99
8	18.64	18.23	18.57	19.41	25.60	22.43	25.58	26.53
16	18.86	18.44	19.04	19.84	26.41	22.51	26.20	26.91
32	19.20	18.78	19.35	20.12	26.48	22.67	26.61	26.93
Z-N	15.77	12.86			22.56	17.63		

Table 2: SNRSEG and SNR Performance of ECBTC and Z-N Schemes at $b_{tot} = 1.25$ and 2.25 bits/sample on the Training Sequence.

	LPCVQ-LMBTC			
	$b_{tot} = 1.25$		$b_{tot} = 2.25$	
M	SNRSEG	SNR	SNRSEG	SNR
1	10.01	13.06	14.94	17.63
2	10.54	13.64	16.63	19.59
4	12.89	15.06	18.67	20.20
8	14.33	15.86	20.39	22.04
16	14.80	16.60	20.41	23.35
32	15.28	16.95	21.38	24.04
64	15.66	16.99	21.98	24.11
128	16.40	17.53	22.65	24.35
256	17.03	17.85	23.22	24.83
512	17.36	18.27	23.98	25.82
1024	17.70	18.63	24.28	27.39
Z-N	15.77	12.86	22.56	17.63

Table 3: Performance of LMBTC and Z-N Schemes at $b_{tot} = 1.25$ and 2.25 bits/sample on a Test Sequence Chosen from the Training Sequence.

	LPCVQ-ECBTC			
	$b_{tot} = 1.25$		$b_{tot} = 2.25$	
M	SNRSEG	SNR	SNRSEG	SNR
1	16.30	17.65	22.76	23.95
2	17.67	18.70	24.52	25.65
4	17.91	18.69	25.28	26.24
8	18.18	18.94	25.53	26.33
16	18.65	19.42	26.28	27.00
32	18.95	19.71	26.58	27.28
64	19.19	19.93	27.44	28.14
128	19.76	20.50	27.92	28.57
256	20.23	20.98		
Z-N	15.77	12.86	22.56	17.63

Table 4: Performance of ECBTC and Z-N Schemes at $b_{tot} = 1.25$ and 2.25 bits/sample on a Test Sequence Chosen from the Training Sequence.

	LPCVQ-LMBTC			
	$b_{tot} = 1.25$		$b_{tot} = 2.25$	
M	SNRSEG	SNR	SNRSEG	SNR
1	7.81	10.13	11.54	13.72
2	7.82	10.04	12.00	14.16
4	8.94	11.01	13.01	15.40
8	10.28	11.72	14.59	16.06
16	10.58	11.97	15.08	17.06
32	11.38	12.29	16.21	17.32
64	11.71	12.65	17.08	18.07
128	12.76	12.96	18.04	18.38
256	13.44	13.03	18.37	18.68
512	13.90	13.29	19.57	19.25
1024	14.66	13.39	20.21	19.35
Z-N	14.23	10.21	20.16	15.03

Table 5: Performance of LMBTC and Z-N Schemes at $b_{tot} = 1.25$ and 2.25 bits/sample on a Test Sequence Chosen from Outside the Training Sequence.

	LPCVQ-ECBTC			
	$b_{tot} = 1.25$		$b_{tot} = 2.25$	
M	SNRSEG	SNR	SNRSEG	SNR
1	14.79	14.83	20.25	20.68
2	16.10	15.92	21.60	22.28
4	16.55	16.28	22.47	22.72
8	17.08	16.85	23.23	23.36
16	17.23	17.03	23.43	23.69
32	17.52	17.33	24.26	24.47
64	17.95	17.96	24.48	25.01
128	18.25	18.36	24.86	25.43
256	19.11	19.13		
Z-N	14.23	10.21	20.16	15.03

Table 6: Performance of ECBTC and Z-N Schemes at $b_{tot} = 1.25$ and 2.25 bits/sample on a Test Sequence Chosen from Outside the Training Sequence.